

REFERENCE MANUAL OF ONLINE TRAINING PROGRAMME ON DIGITAL COMPETENCY, NEW TOOLS AND SOFTWARE FOR EFFICIENT COMPUTER APPLICATIONS

Mode: Online

3-9 January, 2024

Course Coordinator :
Dr. Shashi Dahiya

Course Co-Coordination :
Dr. Sanchita Naha
Mr. Akshay Dheeraj



Division of Computer Applications
ICAR - Indian Agricultural Statistics Research Institute (ICAR-IASRI)
Library Avenue, PUSA, New Delhi - 110012
<https://iasri.icar.gov.in>



कुशल कंप्यूटर अनुप्रयोगों के लिए डिजिटल योग्यता, नए उपकरण और सॉफ्टवेयर

Digital Competency, New Tools and Software for Efficient Computer Applications

Under the aegis of
HRM Unit, ICAR

3 – 9 January 2024

Reference Manual

Course Coordinators:
Dr. Shashi Dahiya, Dr. Sanchita Naha, Mr. Akshay Dheeraj

**ICAR- Indian Agricultural Statistics Research Institute
Library Avenue, New Delhi – 110012**

Editors:

Dr. Shashi Dahiya

Dr. Sanchita Naha

Mr. Akshay Dheeraj

Disclaimer: The information contained in this reference manual has been taken from various web resources. Respective URLs are mentioned in the content.

Citation:

Dahiya, S., Naha, S., Dheeraj, A. (2024). **Digital Competency, New Tools and Software for Efficient Computer Applications**. Reference Manual, ICAR-Indian Agricultural Statistics Research Institute, New Delhi.

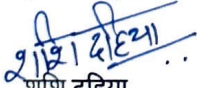
प्रस्तावना

भा.कृ.अनु.प.-भारतीय कृषि सांख्यिकी अनुसंधान संस्थान (भा.कृ.अनु.प - आई ए एस आर आई) सांख्यिकीय विज्ञान (सांख्यिकी, कंप्यूटर अनुप्रयोग और जैव सूचना विज्ञान) में प्रासंगिकता और कृषि अनुसंधान की गुणवत्ता को समृद्ध करने और सूचित नीति निर्णय लेने के लिए कृषि विज्ञान में उनके विवेकपूर्ण संलयन का एक प्रमुख संस्थान है। संस्थान ने कृषि अनुसंधान के लिए उपयोगी विभिन्न सॉफ्टवेयर, वेब एप्लिकेशन, मोबाइल ऐप और अब आर्टिफिशियल इंटेलिजेंस (एआई) आधारित टूल, तकनीक और कार्यप्रणाली विकसित करने में अग्रणी भूमिका निभाई है।

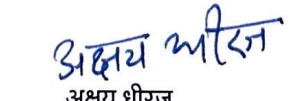
हाल ही में कृषि में डिजिटलीकरण की लहर के बाद से, परिषद के वैज्ञानिक और तकनीकी कर्मचारियों को कुशल कंप्यूटर अनुप्रयोगों के लिए नए उपकरणों और सॉफ्टवेयर के साथ अद्यतन रखने के लिए डिजिटल योग्यता वृद्धि पर समर्पित प्रशिक्षण कार्यक्रमों की आवश्यकता है। संस्थान का एक मुख्य कार्य मानव संसाधन विकास के लिए कृषि में कंप्यूटर अनुप्रयोगों के विभिन्न क्षेत्रों में प्रशिक्षण प्रदान करना है। इसके अलावा, माइक्रोसॉफ्ट एक्सेल, आर, पायथन, डीबीएमएस आदि सॉफ्टवेयर विभिन्न कृषि प्रयोगों से निकले आंकड़ों के विश्लेषण में प्रमुख भूमिका निभा रहे हैं। भा.कृ.अनु.प -आई ए एस आर आई 03-09 जनवरी, 2024 के दौरान परिषद के वैज्ञानिक और तकनीकी कर्मचारियों के लिए "डिजिटल योग्यता, कुशल कंप्यूटर अनुप्रयोगों के लिए नए उपकरण और सॉफ्टवेयर" पर 7 दिवसीय एचआरएम अनुमोदित प्रशिक्षण कार्यक्रम का आयोजन कर रहा है। इस प्रशिक्षण कार्यक्रम का उद्देश्य प्रतिभागियों को कृषि के लिए अत्याधुनिक डिजिटल समाधान तैयार करने में व्यापक रूप से उपयोग किए जाने वाले उपकरणों का व्यापक अवलोकन प्रदान करना है। व्यावहारिक सत्रों के माध्यम से, प्रतिभागी इन उपकरणों और सॉफ्टवेयर को नेविगेट करने और उपयोग करने में अपनी दक्षता बढ़ाएंगे। इसके अतिरिक्त, कार्यक्रम में भा.कृ.अनु.प की लोकप्रिय ई-गवर्नेंस पोर्टल, एनएआरईएस के लिए आईटी अनुप्रयोग, KRISHI (नवाचार के लिए कृषि ज्ञान संसाधन और सूचना प्रणाली हब), फसलों के लिए कृत्रिम बुद्धिमत्ता आधारित रोग पहचान प्रणाली, एनएआरईएस-बीएलपी (ब्लेंडेड लर्निंग प्लेटफॉर्म) का प्रदर्शन शामिल होगा।

हमने इस पाठ्यक्रम को कई डिजिटल टूल और सॉफ्टवेयर पर सिद्धांत और व्यावहारिक दोनों कक्षाएं प्रदान करने के लिए डिज़ाइन किया है। इस पाठ्यक्रम के अंतर्गत शामिल विषयों में Google फॉर्म विकास, पायथन मूल बातें, डेटा प्रबंधन और विज़ुअलाइज़ेशन तकनीक और एमएस-एक्सेल, आर का उपयोग करके सांख्यिकीय डेटा विश्लेषण शामिल हैं; आर का उपयोग करके कृषि प्रयोगों की योजना बनाना और डिज़ाइन करना, वेब एप्लिकेशन और मोबाइल एप्लिकेशन का विकास, डेटाबेस प्रबंधन प्रणाली और साइबर सुरक्षा।

हम इस अवसर पर संस्थान के संकाय को धन्यवाद देना चाहते हैं जिन्होंने इस पाठ्यक्रम को सफल बनाने में अपना बहुमूल्य समय दिया। उनके सहयोग के बिना इस मैनुअल को समय पर पूरा करना संभव नहीं होता। हम इस प्रशिक्षण कार्यक्रम में अपने कर्मचारियों को तैनात करने के लिए विभिन्न आईसीएआर संस्थानों के भी आभारी हैं। हम डॉ. राजेंद्र प्रसाद, निदेशक, भा.कृ.अनु.प - भा.कृ.सां.अनु.सं. के बहुमूल्य मार्गदर्शन और पाठ्यक्रम के सुचारू संचालन के लिए सभी आवश्यक सुविधाएं उपलब्ध कराने के लिए आभारी हैं। हम उन सभी के आभारी हैं जिन्होंने इस प्रशिक्षण मैनुअल को तैयार करने के लिए प्रत्यक्ष या अप्रत्यक्ष रूप से हमारा समर्थन किया है।


शाशि दहिया
प्रशिक्षण समन्वयक


संचिता नाहा
प्रशिक्षण सह समन्वयक


अक्षय धीरज
प्रशिक्षण सह समन्वयक

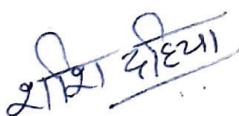
PREFACE

ICAR-Indian Agricultural Statistics Research Institute (ICAR-IASRI) is a premier Institute of relevance in Statistical Sciences (Statistics, Computer Applications and Bioinformatics) and their judicious fusion in agricultural sciences for enriching quality of agricultural research and informed policy decision making. The Institute has taken a lead in developing various Software, Web Applications, Mobile apps and now Artificial Intelligence (AI) based tools, techniques, and methodologies useful for Agricultural Research.

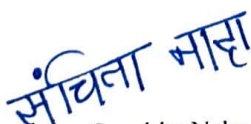
Since the recent wave of digitization in agriculture, scientific and technical staff of the council need dedicated training programmes on digital competency enhancement to keep them up to date with new tools and software for efficient computer applications. One of the institute's main mandates is to impart training in various areas of computer applications in agriculture for human resource development. Moreover, software like Microsoft Excel, R, Python, DBMS etc. are playing major roles in analysing data emerged from various agricultural experiments. ICAR-IASRI is organizing a 7-days HRM approved training programme on "Digital Competency, New Tools and Software for Efficient Computer Applications" for the scientific and technical staff of ICAR during January 03-09, 2024. This training program aims to provide participants with a comprehensive overview of the tools widely employed in crafting cutting-edge digital solutions for agriculture. Through hands-on practical sessions, participants will enhance their proficiency in navigating and utilizing these tools and software. Additionally, the program will feature demonstrations of popular e-governance initiatives of ICAR, IT applications for NARES, KRISHI (Agricultural Knowledge Resources and Information System Hub for Innovations), AI-DISC (Artificial Intelligence based Disease Identification System for Crops), NARES-BLP (Blended Learning Platform) commonly used by Scientific and Technical staff associated with ICAR.

We have designed this course to offer both theory and practical classes on several digital tools and software. The topics covered under this course include Google forms development, Python basics, data handling and visualization techniques and statistical data analysis using MS-Excel, R; planning and designing of agricultural experiments using R, development of web applications and mobile applications, database management systems and Cyber security.

We would like to take this opportunity to thank the faculty of the institute who have spared their valuable time in making this course successful. Without their cooperation timely completion of this manual would not have been possible. We are also thankful to the various ICAR Institutes for deputing their employees in this training programme. We are grateful to Dr. Rajender Parsad, Director, ICAR-IASRI for his valuable guidance and making all necessary facilities available for smooth conduct of the course. We are thankful to everyone who has supported us directly or indirectly for preparing this training manual.



Shashi Dahiya
Training Coordinator



Sanchita Naha
Training Co-coordinator



Akshay Dheeraj
Training Co-coordinator

Lecture Schedule for Training Program on
“Digital Competency, New Tools and Software for Efficient Computer Applications”
3-9 January 2024
(ONLINE Mode)
Division of Computer Applications, ICAR-IASRI, New Delhi

Date	09:30-10:00	10:00-11:15	11:30-1:00	L U N C H B R E A K	2:15-3:30	3:45-5:00
(03/01/2024) Wednesday	Inaugural Session	Google Forms Development (Sh. Akshay Dheeraj)			Statistical Data Analysis using MS Excel (Dr. Sanchita Naha)	
Date	09:15-11:15	11:30-01:00			2:15-3:30	3:45-5:00
(04/01/2024) Thursday	Data Analysis using R Software (Dr. Soumen Pal)	Descriptive Statistics and Exploratory Data Analysis: Hands on Session (Dr. Achal Lama)			Planning and Designing of Experiments using R (Dr. Md. Harun)	Database Management System (Dr. Shashi Dahiya)
Date	09:15-10:30	10:30-11:45	12:00-01:15		2:15-3:30	3:45-5:00
(05/01/2024) Friday	Web Application Development using HTML (Dr. Alka Arora)		Basics of Programming using Python (Dr. Madhu Dahiya)		Hands on Session on Python (Dr. Madhu Dahiya)	Data Handling and Visualization using Python (Dr. Samarth Godara)
(08/01/2024) Monday	e-governance initiatives in ICAR (Dr. Mukesh Kumar)	Mobile Application Development (Dr. Md Ashraful Haque)			IASRI - IT Applications for NARES (Dr. Sudeep Marwaha)	KRISHI (Dr. Anshu Bharadwaj)
(09/01/2024) Tuesday	AI-DISC: Artificial Intelligence based Disease Identification System for Crops (Dr. Chandan Kumar Deb)	Feedback Session	NARES-BLP (Dr. Sapna Nigam)		Networking basics and Cyber security (Subhash Chand)	Valedictory Session

CONTENTS

S.No.	Topic	Author	Page No.
1.	Google Forms Development	Akshay Dheeraj	1-12
2.	Statistical Data Analysis using MS Excel	Sanchita Naha	13-29
3.	Data Analysis using R Software	Soumen Pal	30-49
4.	Descriptive Statistics and Exploratory Data Analysis	Achal Lama	50-65
5.	Planning and Designing of Experiments using R	Md. Harun	66-90
6.	Database Management System	Shashi Dahiya	91-107
7.	Web Application Development using HTML	Alka Arora	108-141
8.	Basics of Programming using Python	Madhu Dahiya	142-184
9.	Data Handling and Visualization using Python	Samarth Godara	185-190
10.	E-Governance initiatives in ICAR	Mukesh Kumar	191-203
11.	Mobile Application Development	Md Ashraful Haque	204-223
12.	IASRI - IT Applications for NARES	Sudeep Marwaha	224-227
13.	Knowledge Resources and Information System Hub for Innovations (KRISHI)	Anshu Bharadwaj	228-242
14.	AI-DISC: Artificial Intelligence based Disease Identification System for Crops	Chandan Kumar Deb	243-247
15.	National Agricultural Research and Education System- Blended Learning Platform (NARES-BLP)	Sapna Nigam	248-251
16.	Networking basics and Cyber security	Subhash Chand	252-259

Google Forms Development

Akshay Dheeraj

ICAR-Indian Agricultural Statistics Research Institute, New Delhi - 110 012

akshay.dheeraj@icar.gov.in

Topics covered in this lecture:

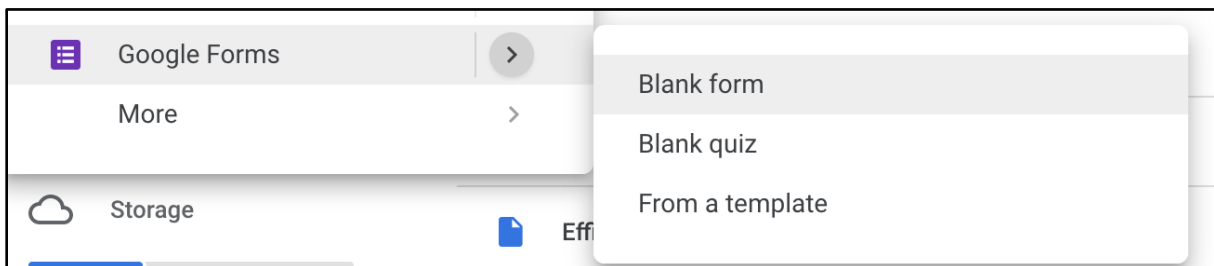
- Creating a form
- Question types and options
- Adding and moving questions
- Adding sections
- Sharing a form
- Adding photos
- Changing the theme
- Creating a self-grading quiz
- Previewing a form
- Reviewing results

Introduction

Google Forms is one of the programs available in Google Drive, along with Docs, Sheets, and Slides. You can use Google Forms to create a survey and gather responses. The answers are stored on a Google Sheet spreadsheet, so it's helpful to be somewhat familiar with Sheets. Some uses of Forms beyond the simple survey are self-grading quizzes.

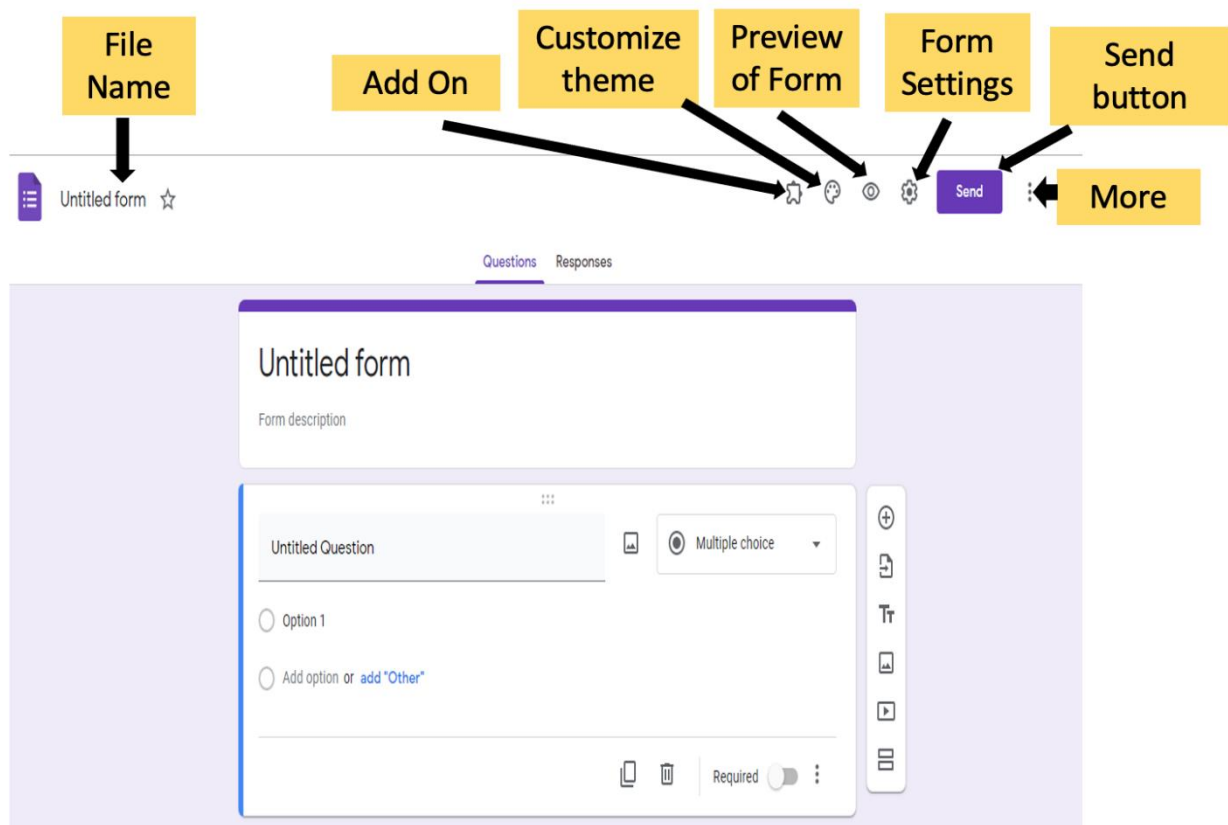
Creating a New Form

Log in to your Google Drive account (drive.google.com). Click on **New**, then mouse down to Google Forms and hover over the arrow next to it. (If you haven't used it before, it might be hiding under **More**.) The arrow gives you two options: **Blank form** and **from a template**. If you just click on Google Forms, it will open a new blank form.



Click on **From a template** to see all of the pre-created template options you can use. The templates include a party invitation, a form for deciding on the best time to meet, a feedback form, an order form, and more.

Screen Layout



You can rename the file in the upper left by clicking on the name. You can move it by clicking on the folder icon, or you can start it, to make it easier to find later. In the upper right are more detailed controls. The paint palette icon is **Customize Theme**. You can change the header image, pick new theme colors, or change the font. The eye icon is **Preview of Form**, which lets you see what the live version of the form looks like. It opens in a new browser tab. The gear is **Form Settings**, where you can control how people will interact with the form. Use the **Send** button when you want to email your completed form to people to invite them to fill it out. The three vertical dots are **More**, where you can find **Undo**, **make a copy**, **Add-Ons** and other commands.

Note: Most actions can be undone using **Undo**. You can also use CTRL-Z on your keyboard. CTRL-Y is **Redo**.

Question Types

The essence of Google Forms is creating questions and providing a way for people to answer them. There are ten (10) question types that dictate how a person can answer. Most types have additional options under the ⋮ icon in the question box.

Short answer – A very short text answer. Will display a small textbox. You can control what type of answer is valid with **Response validation**.

Paragraph – A longer text answer. Will display a longer textbox that will line-wrap if necessary. You can control what's valid with **Response validation**.

Multiple choice – A list of options, only one of which can be selected. Displays as radio buttons (circles). 'Other' can be an option, which includes a textbox next to it. You can **Shuffle option order** or **Go to a section based on the answer**.

Checkboxes – A list of options with the potential to select more than one. Displays as checkboxes (squares). 'Other' can be an option, which includes a textbox. **Response validation** here controls how many boxes a respondent can select. You can also **Shuffle option order**.

Dropdown – Displays options as a dropdown menu. Only one option can be selected. Useful if you have a lot of items (such as 50 states). While you can add 'Other' as a choice, it will not create a textbox. You can **Shuffle option order** or **Go to a section based on the answer**.

File upload – Allows respondents to upload a file to your Google Drive. You can control what file types are allowed, how many files they can upload, and what their maximum size can be.

Linear scale – Displays a row of radio buttons from 0 or 1 to X, with labels on either side. (For example, Not Satisfied->Satisfied on a scale of 1-10.)

Multiple choice grid – Displays a grid of radio buttons. Only one option can be selected from each row. You can **Shuffle option order** or **Limit to one response per column**.

Checkbox grid – Displays a grid of checkboxes. Any number of boxes can be checked. You can **Shuffle option order** or **Limit to one response per column**.

Date – Displays mm/dd/yyyy which can be filled in by typing or clicking on to display a calendar that a date can be selected. You can choose to include the time or the year.

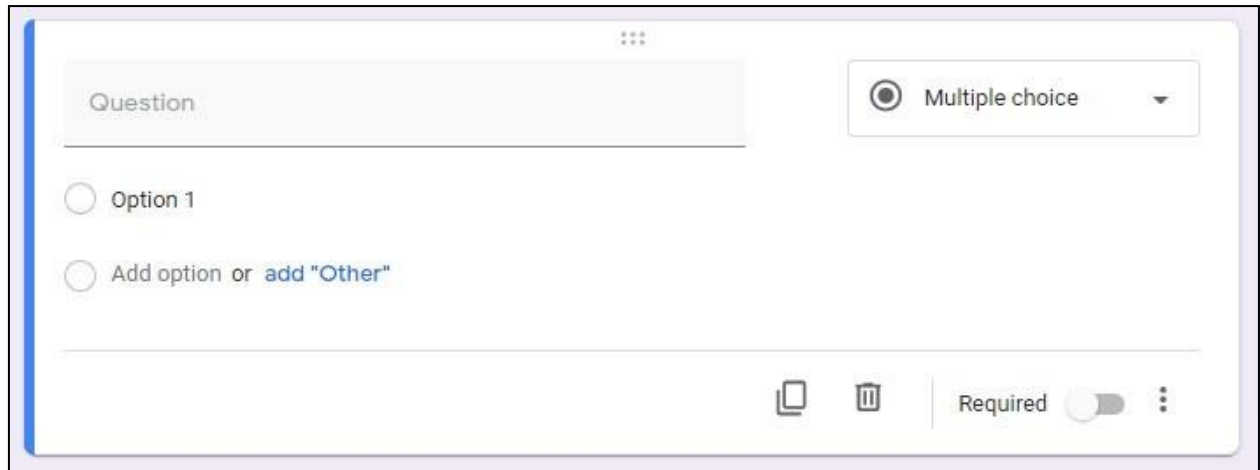
Time – By default, displays an empty time to be filled in. “:” and AM/PM dropdown menu. You can change this to **Duration**, which asks HRS: MIN: SEC.

Title and Description

On a blank form, it says **Untitled form**. Click here and you can change the title of your form. Click on the **Form description** to describe what the form is to your future respondents. If you choose to write nothing, it will display as blank.

The Question Box

Google Forms starts you off with one question box. No matter what type of question you choose, some of the controls remain the same.

A screenshot of a Google Form question box. At the top, there is a text input field labeled "Question". To its right is a dropdown menu currently set to "Multiple choice". Below the question field, there are two radio button options: "Option 1" and "Add option or add 'Other'". At the bottom right of the question box, there are icons for duplicating (two overlapping squares) and deleting (a trash can) the question, followed by a "Required" toggle switch which is currently turned off, and a vertical ellipsis menu icon.

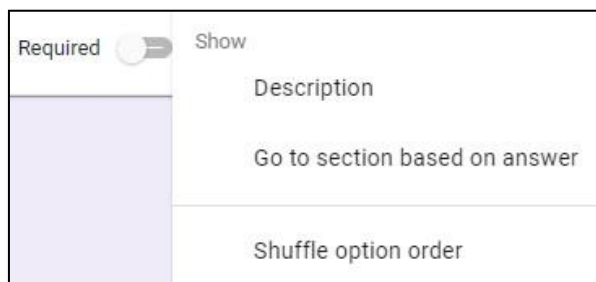
Click on **Untitled Question** or **Question** to put in the text of your question. It can be something as simple as asking for "First Name" or an entire paragraph detailing a math problem.

When you hover over the Question text box, a **photo icon** appears. Click on it to add a photo to that question. You can upload from your computer or Google Drive, or do a Google Imagesearch. Be mindful of copyright if you use a picture from the internet.

The drop-down box in the upper right is where you select your question type.

At the bottom of the question box are icons to **Duplicate** or **Delete** the question.

The **Required** toggle is for making a question optional or required. Click the toggle to require it. Click it again to make it not required.

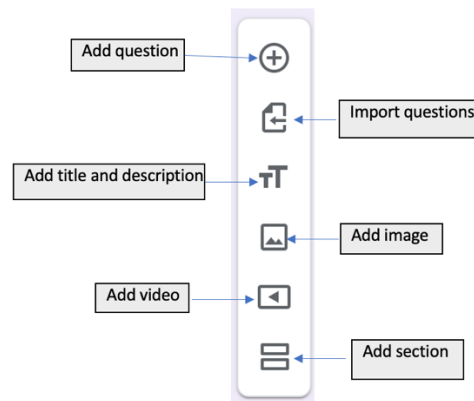
A screenshot of the expanded menu for the question box. On the left, there is a "Required" toggle switch which is currently turned on. To its right is a "Show" button. Below these, there are three menu items: "Description", "Go to section based on answer", and "Shuffle option order".

The vertical ellipsis, :, has more options. You can add a **Description** box to your question. Other options that appear on this menu vary based on the type of question you've selected. **Go to a section based on the answer** can be used in more advanced surveys when you want to direct people to different questions

based on their answer to this one. The shuffle **option order** will display the options in a different order each time the form is loaded. Useful if you want to avoid biases based on the order. Less useful if there's a logical order to your answers (such as age ranges).

The Floating Bar

When you click on a question to edit or add a new one, a bar appears floating to the right of it.



Add question – Add a new question.

Import questions – Import questions from another Google Form.

Add title and description – Create another title and description box like the one at the top.

Add Image – Add an image between questions (rather than in a question or answer itself). **Add video** – Add a video. Has to be a video on YouTube.

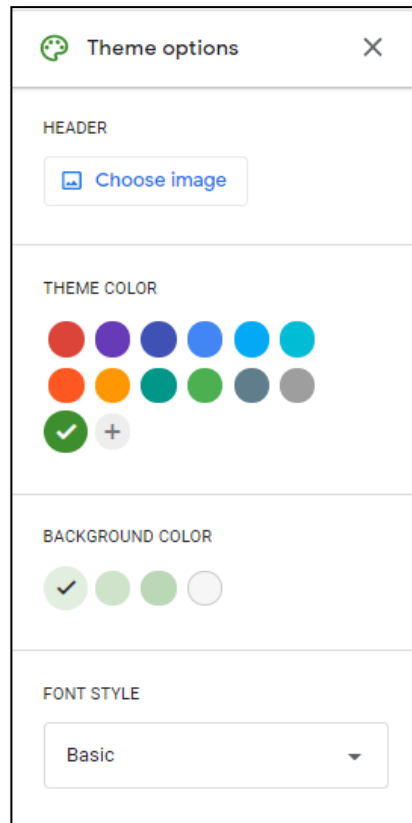
Add section – Add a new section. This adds the **Next** button and takes people to a new page of your form.

If you'd like to rearrange the order of your questions or other elements, hover over one of the boxes until a little grid-shaped icon appears. Click on it and you can drag that element up or down the page and drop it where you want it

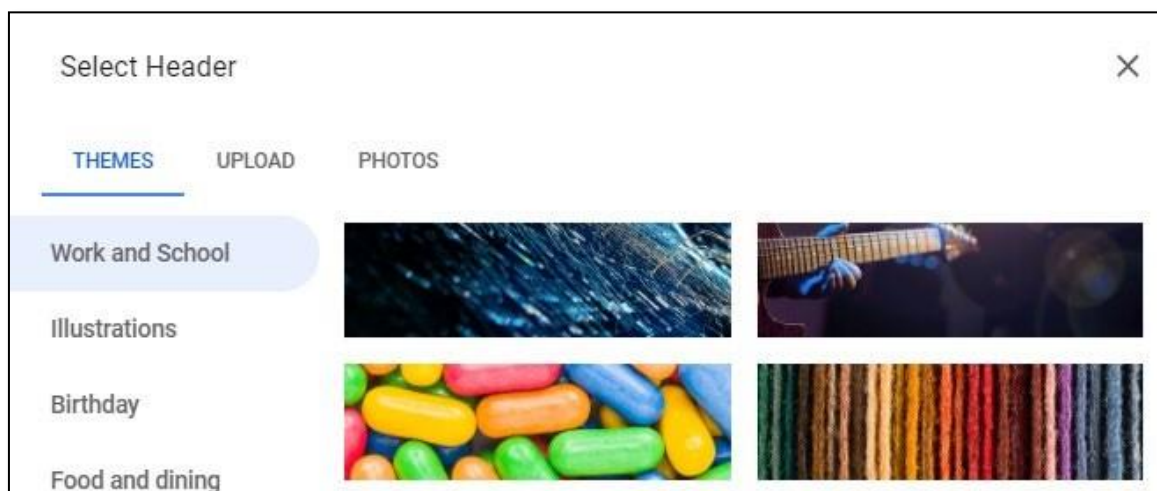


Customizing Theme

You can change (or add) an image at the top of your form and customize the colours and fonts by clicking the **Customize Theme** icon in the upper right that looks like a paint palette.



For images, you have the option of selecting an image from one of the **Themes**, **uploading** your own, or choosing from **Photos** you may already have in your Google account.



Settings

The gear icon in the upper right is **Settings**. Here you can control whether you're going to collect email addresses and provide receipts, whether you want respondents to be able to fill out the form more than once, whether they can edit their answers after they submit them, and whether they can see how others have responded.

Settings

GeneralPresentationQuizzes

☒ Collect email addresses

☒ Response receipts ?

☒ If respondent requests it

☐ Always

Requires sign in:

☒ Limit to 1 response

Respondents will be required to sign in to Google.

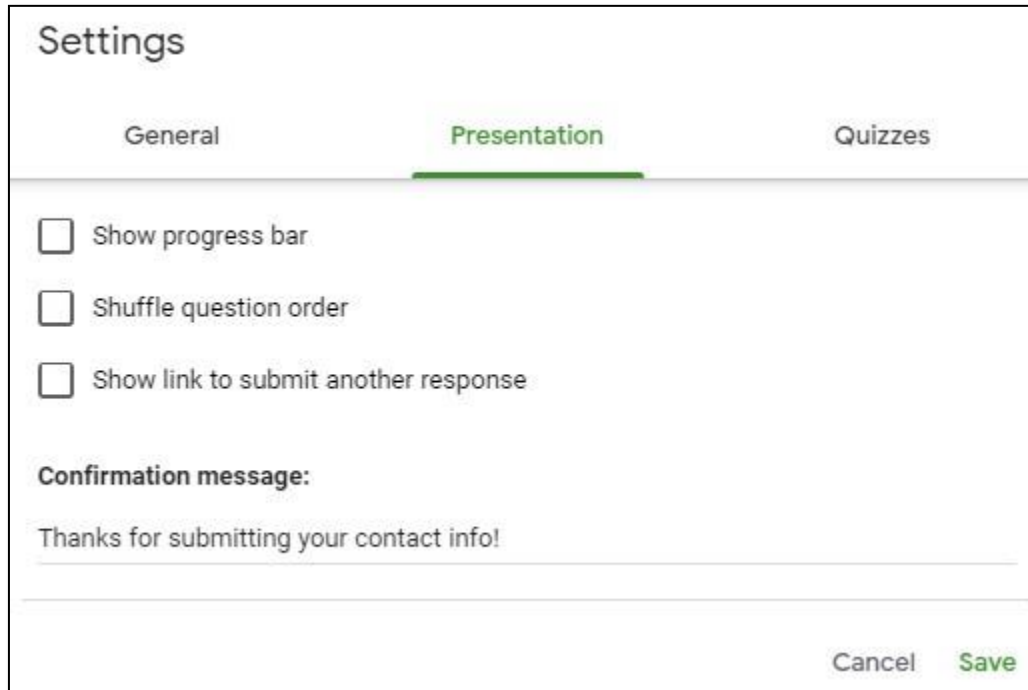
Respondents can:

☒ Edit after submit

☐ See summary charts and text responses

CancelSave

On the **Presentations** tab of the **Settings** box, you can decide if respondents will see a progress bar as they complete your form, whether you want to shuffle the order of the questions, and if you allow multiple submissions, a link to where they can fill out the form again. You can also customize your **Confirmation message**.



The screenshot shows a 'Settings' dialog box with three tabs: 'General', 'Presentation' (which is selected and highlighted with a green underline), and 'Quizzes'. Under the 'Presentation' tab, there are three unchecked checkboxes: 'Show progress bar', 'Shuffle question order', and 'Show link to submit another response'. Below these is a section labeled 'Confirmation message:' followed by a text input field containing the text 'Thanks for submitting your contact info!'. At the bottom right of the dialog box are two buttons: 'Cancel' and 'Save'.

Creating a Quiz

To turn a form into a quiz, click the **Settings** gear icon in the upper right. Click on the **Quizzes** tab. Toggle **Make this a quiz**. You're given the option that people answering the quiz can see the results **Immediately after each submission** or, if you'd like to manually grade them yourself, or give a group of people their results all at once, choose **Later, after manual review**.

You can also choose whether people taking your quiz will be able to see **Missed questions**, **Correct answers**, or **Point values**. If it's just a fun quiz, you probably want them all checked. If it's for a class, you might choose to keep some or all of those hidden to reduce cheating amongst classmates. Or if you'd like to give someone a chance to keep trying, maybe show them which questions they missed, but not the correct answers.

Settings

General

Presentation

Quizzes

☒

Make this a quiz
Assign point values to questions and allow auto-grading.

Quiz options

Release grade:

☒

Immediately after each submission

☐

Later, after manual review
Turns on email collection

Respondent can see:

☒

Missed questions ⓘ

☒

Correct answers ⓘ

☒

Point values ⓘ

Cancel

Save

Quiz Questions and Answers

Create sections and questions as you would for a survey, but now your question box will have a place for you to put the correct answer(s).

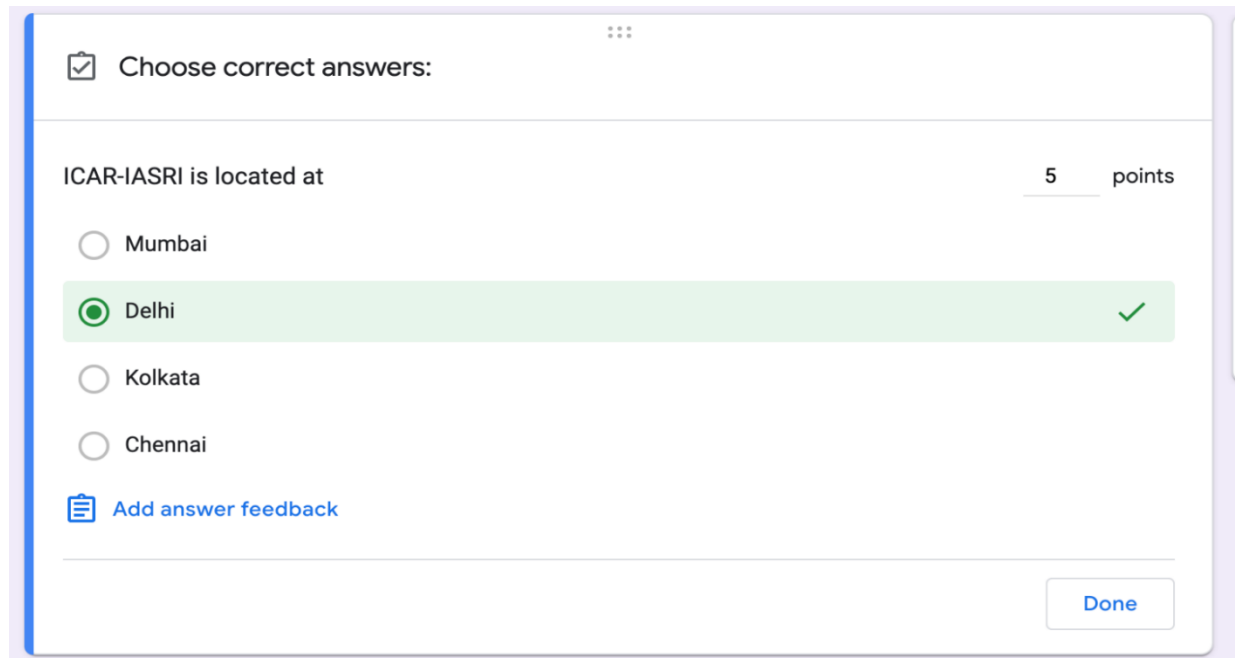
☒

Answer key (5 points)

Required

☐

Clicking on the Answer key will change the box to look something like this:



The screenshot shows a quiz question interface. At the top, there is a checkbox icon and the text "Choose correct answers:". Below this, the question "ICAR-IASRI is located at" is displayed on the left, and "5 points" is on the right. There are four radio button options: "Mumbai", "Delhi", "Kolkata", and "Chennai". The "Delhi" option is selected, indicated by a green circle and a green highlight bar. To the right of the "Delhi" option is a green checkmark icon. At the bottom left, there is a link that says "Add answer feedback" with a document icon. At the bottom right, there is a "Done" button.

As this example is Multiple Choice, only one answer can be the correct one. (Use Checkboxes if you want a multiple-choice question with multiple correct answers.) Select the correct answer.

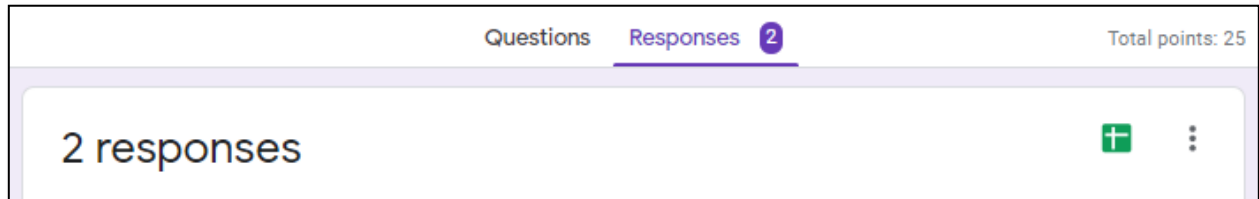
For Short Answer questions, you can add as many correct answers to your Answer key as necessary. This is case-sensitive, so you'll want to provide a few options.

Next, in the upper right, you can assign a point value to this question. Click Add answer feedback if you'd like to explain why an answer is correct, or why other answers are incorrect. You can add a picture or a link to your feedback.

Preview and Responses

Click the **Preview** eye icon in the upper left of the form to preview the quiz and take it yourself. It's always a good idea to test it out.

Once your quiz is live and some people have taken it (even if it's only you!), in the top middle of the screen next to **Questions** is **Responses**. It will have a number next to it telling you how many responses you've had. Click on **Responses** to get a summary of how your respondents are doing.



Click on the green **Create spreadsheet** icon to do just that. You can create a new Sheets spreadsheet or add to an existing one.

Click on the vertical ellipsis for more options. You can **Get email notifications for new responses**. You can **Download responses** to a CSV file (to import into Excel, for example), you can **Print all responses**, and can **Delete all responses**. You might want to **Delete all responses** after you've done your testing and are ready to share the quiz with others. If you have already created a spreadsheet, the responses will not be deleted from it.

Sending the Form to Potential Respondents

Now that you have a survey, a quiz, or another form, you probably want to share it with other people so they can fill it out. You have many options of ways to share it. First, click on the large **Send** button in the upper right.

Your main options are in the three tabs and the two social media buttons.

Send via Email will email invitations to the people or mailing lists you send it to.

Send via the link will give you a URL you can copy and paste.

Send via embed HTML will give you code to put on your website if you'd like the form to appear there.

Facebook or **Twitter** will start a new post on those platforms under whatever account you're logged in there as.

You can use as many of these options as you want.

Add collaborators is for adding people you want to give permission to view and modify the form.

Statistical Data Analysis using Microsoft Excel

Dr. Sanchita Naha

ICAR-Indian Agricultural Statistics Research Institute, New Delhi - 110 012

sanchita.naha@icar.gov.in

Statistics is the study of collection, analysis, interpretation, presentation, and organization of data. Broadly, two statistical methodologies are used for data analysis, descriptive statistics, and inferential statistics. Statistical analysis can be done using software like MS Excel, SPSS, R but this tutorial is restricted to major statistical analysis methods using Microsoft Excel. Statistical analysis mainly encompasses descriptive statistics and inferential statistics.

1. Descriptive Statistics: Descriptive statistics is used to describe or summarize data in a meaningful way. Descriptive statistics provides us with tools, tables, graphs, averages, ranges, correlations for organizing and summarizing data. In descriptive statistics data is summarized with the following major numerical descriptors like

- **Arithmetic Mean:** It is defined as the average of the data values. For mean computation, data must be in numeric form.

$$Mean = \frac{\sum_{i=1}^n x_i}{n}$$

n = number of observations

Steps to compute mean in Excel:

Select data points > Click formulas > Expand auto-sum drop down menu > select Average > Enter.

The screenshot shows the Microsoft Excel interface with the 'Formulas' tab selected. The formula bar displays '=AVERAGE(B2:B11)'. The worksheet contains a table with names in column A and ages in column B. The average age is calculated in cell B12.

	A	B	C	D	E	F	G	H
1	Name	Age						
2	Alice	45						
3	Bob	56						
4	Carol	23						
5	Dave	60						
6	Eve	65						
7	Mallory	11						
8	Walter	40						
9	Trent	65						
10	Peggy	79						
11	Victor	34						
12		=AVERAGE(B2:B11)						
13								

- **Geometric Mean:** It is the n^{th} root of the product of individual data points. Let X_1, X_2, \dots, X_n be the n^{th} observation of a data set. The geometric mean is defined as

$$GM = (x_1 * x_2 * \dots * x_n)^{1/n}$$

Steps to compute geometric mean in Excel:

Select data points > Click formulas > Expand auto-sum drop down menu > Find function

‘GEOMEAN’ > click ‘Insert Function’ > Enter.

e.g., GEOMEAN (B2:B11)

	A	B	C	D	E	F	G
1	Name	Age					
2	Alice	45					
3	Bob	56					
4	Carol	23					
5	Dave	60					
6	Eve	65					
7	Mallory	11					
8	Walter	40					
9	Trent	65					
10	Peggy	79					
11	Victor	34					
12							
13	Geometric Mean	=GEOMEAN(B2:B11)					
14							

The geometric mean is used in finance to calculate average growth rates and is referred to as the compounded annual growth rate.

- **Harmonic Mean:** Harmonic mean is the reciprocal of the arithmetic mean of the reciprocals of the observations of the datasets.

$$HM = \frac{n}{\sum_{i=1}^n 1/x_i}$$

Steps to compute harmonic mean in Excel:

Select data points > Click formulas > Expand auto-sum drop down menu > Find function 'HARMEAN' > click 'Insert Function' > Enter.

e.g., HARMEAN (B2:B11)

The screenshot shows the Microsoft Excel interface with the 'Formulas' tab selected. The formula bar displays '=HARMEAN(B2:B11)'. The spreadsheet contains the following data:

	A	B	C	D	E	F	G
1	Name	Age					
2	Alice	45					
3	Bob	56					
4	Carol	23					
5	Dave	60					
6	Eve	65					
7	Mallory	11					
8	Walter	40					
9	Trent	65					
10	Peggy	79					
11	Victor	34					
12							
13	Harmonic Mean	B2:B11)					
14							

- **Median:** Median is the value in the middlemost position of all the observations when arranged in an ascending/descending order. The median is the central value of an ordered data series. It divides the data sets exactly into two parts. Fifty percent of observations are below the median value and 50% are above the median. Median is also known as 'positional average'.

Steps to compute median in Excel:

Select data points > Click formulas > Expand auto-sum drop down menu > Find function 'MEDIAN' > click 'Insert Function' > Enter.

- **Mode:** Mode is defined as the value that occurs most frequently in the data. If in the data sets each observation occurs only once, then it does not have mode. When the data set has two or more values equal to the highest frequency than two or more mode are present in the datasets.

Steps to compute median in Excel:

Select data points > Click formulas > Expand auto-sum drop down menu > Find function 'MODE' > click 'Insert Function' > Enter.

- **Range:** It is defined as the difference between the highest value and lowest value of the variable.

$$\text{Range} = \text{Maximum value} - \text{Minimum value}$$

Steps to compute range in Excel:

Compute the maximum and minimum value among the data values. Then compute the difference between them to get the range of observations.

Select data points > Click formulas > Expand auto-sum drop down menu > Find function 'MAX' > click 'Insert Function' > Enter.

Select data points > Click formulas > Expand auto-sum drop down menu > Find function 'MIN' > click 'Insert Function' > Enter.

Select a cell > write “= (specify the cell where maximum value is stored - specify the cell where minimum value is stored)” in the formula bar > Enter.

The screenshot shows the Microsoft Excel interface with the 'Formulas' tab selected. The data table is as follows:

	A	B	C	D	E	F	G	H	I
1	Name	Age	xi - mean	square					
2	Alice	45	-2.8	7.84					
3	Bob	56	8.2	67.24					
4	Carol	23	-24.8	615.04					
5	Dave	60	12.2	148.84					
6	Eve	65	17.2	295.84					
7	Mallory	11	-36.8	1354.24					
8	Walter	40	-7.8	60.84					
9	Trent	65	17.2	295.84					
10	Peggy	79	31.2	973.44					
11	Victor	34	-13.8	190.44					
12									
13	Average	47.8		4009.6					
14	GM	42.069418							
15	HM	34.605356							
16	Median	50.5							
17	Mode	65							
18		79							
19	Max	79							
20	Min	11							
21	Standard Function	21.107134							
22	Kurtosis	-0.598827							
23									

- **Standard Deviation:** It is defined as the positive square-root of the arithmetic mean of the square of the deviations of the given observations from their arithmetic mean. It takes into consideration the magnitude of all the observations and gives the minimum value of dispersion possible. It is also known as Root Mean Square Deviation about mean.

Let x_1, x_2, \dots, x_n are the n observations in a data set. The standard deviation S.D. is given by,

$$SD = \sqrt{\frac{\sum_{i=1}^n (X_i - \bar{X})^2}{n}}$$

- **Variance:** It is defined as the square of the standard deviation. Unit of the variance is the square of the actual observations, whereas unit of the standard deviation is same as the actual observations.

Steps to calculate Standard Deviation in excel:

Select data points > Click formulas > Expand auto-sum drop down menu > Find function 'STDEV' > click 'Insert Function' > Enter.

The screenshot shows the Microsoft Excel interface with the 'Formulas' tab selected. The formula bar displays `=STDEV(B2:B11)`. The 'Formula Builder' pane on the right shows the 'STDEV' function with 'Number1' set to `{45;56;23;60;65;11;40;65;79;34}` and 'Number2' set to `number`. The spreadsheet data is as follows:

	A	B	C	D
1	Name	Age		
2	Alice	45		
3	Bob	56		
4	Carol	23		
5	Dave	60		
6	Eve	65		
7	Mallory	11		
8	Walter	40		
9	Trent	65		
10	Peggy	79		
11	Victor	34		
12	Standard Deviatoin	<code>B11)</code>		
13				

There are 6 versions of standard deviation formula available which are as following:

STDEV.S: This formula calculates the sample standard deviation based on numeric information alone. It ignores text and logical (TRUE or FALSE) values in the spreadsheet. The denominator in this case is $(n-1)$.

STDEV.P: This formula calculates the standard deviation for an entire population based on numeric information alone. It ignores text and logical values in the spreadsheet. The denominator in this case is n .

STDEVA: This formula calculates the sample standard deviation of a dataset but includes text and logical values in the calculation. All FALSE values are represented by 0, and TRUE values are represented by 1.

STDEVPA: This formula calculates the standard deviation for an entire population and includes text and logical values in the calculation. Like STDEVA, all FALSE valse are represented by 0, and TRUE values are represented by 1.

STDEV: This is an older version of the STDEV.S formula that Excel used to calculate sample standard

deviation before 2007. It still exists for compatibility purposes. This formula acts as the same as STDEV.S

STDEVP: This is an older version of the STDEV.P formula that still exists for compatibility.

Steps to calculate Variance in excel:

Select data points > Click formulas > Expand auto-sum drop down menu > Find function ‘STDEV’ > click ‘Insert Function’ > Enter.

- **Coefficient of Variation (CV):** The Coefficient of Variation (CV) is defined as the ratio of the standard deviation to the mean, and expressed in percentages,

$$CV = \frac{\text{Standard Deviation}}{\text{Mean}} * 100$$

CV is calculated to have an idea about the consistency/ variability of the series. Higher the CV means the series is more variable, less stable, less uniform, and less consistent. Lesser CV indicates that the series is less variable or more stable or more uniform and more consistent.

- **Skewness and Kurtosis:** Skewness is used to detect outliers in a data set. It characterizes the degree of asymmetry of a distribution around its mean. Positive skewness indicates a distribution with an asymmetric tail extending toward more positive values. Negative skewness indicates a distribution with an asymmetric tail extending toward more negative values. A data series is said to be positively skewed if the Mean of the data series is greater than Median and is greater than Mode. On the other hand data is said to be negatively skewed if $\text{Mean} < \text{Median} < \text{Mode}$. Data series is said to be symmetric if $\text{Mean} = \text{Median} = \text{Mode}$.

$$\text{Pearson's Coefficient of Skewness} = \frac{(\text{Mean} - \text{Mode})}{\text{Standard Deviation}}$$

Alternate formula for computing Skewness Coefficient,

$$\text{Coefficient of Skewness} = \frac{3 (\text{Mean} - \text{Median})}{\text{Standard Deviation}}$$

If Skewness coefficient = 0, then the distribution is normal and symmetrical.

If Skewness coefficient > 0, then the frequency distribution is positively skewed.

If Skewness coefficient < 0, then the frequency distribution is negatively skewed.

Steps to calculate Skewness Coefficient in excel:

Select data points > Click formulas > Expand auto-sum drop down menu > Find function ‘SKEW’ > click ‘Insert Function’ > Enter.

<div> <div>Home</div> <div>Insert</div> <div>Draw</div> <div>Page Layout</div> <div>Formulas</div> <div>Data</div> <div>Review</div> <div>View</div> <div>Tell me</div> </div>						
<div> <div> <div>Paste</div> <div> <div>Calibri (Body)</div> <div>12</div> <div>A⁺</div> <div>A⁻</div> </div> </div> <div> <div>B</div> <div>I</div> <div>U</div> <div> <div></div> <div></div> </div> <div> <div></div> <div></div> </div> </div> <div> <div></div> <div></div> </div> <div> <div></div> <div></div> </div> <div> <div></div> <div></div> </div> <div> <div>Wrap Text</div> <div>Merge & Centre</div> </div> </div>						
<div> <div>D2</div> <div></div> <div></div> <div></div> <div>fx</div> <div>Column1</div> </div>						
	A	B	C	D	E	F
1	Name	Age				
2	Alice	45		Column1		
3	Bob	56				
4	Carol	23		Mean	47.8	
5	Dave	60		Standard Error	6.67466187	
6	Eve	65		Median	50.5	
7	Mallory	11		Mode	65	
8	Walter	40		Standard Deviation	21.1071341	
9	Trent	65		Sample Variance	445.511111	
10	Peggy	79		Kurtosis	-0.5988268	
11	Victor	34		Skewness	-0.3736531	
12				Range	68	
13				Minimum	11	
14				Maximum	79	
15				Sum	478	
16				Count	10	
17						

- Kurtosis:** Kurtosis is a measure of the “tailedness” of the probability distribution of a real-valued random variable. It is the tailedness of a distribution relative to a normal distribution. Distributions with medium kurtosis (medium tails) are mesokurtic, with low kurtosis are called platykurtic, and distributions with high kurtosis are leptokurtic.

$$\text{Measure of kurtosis, } \gamma_2 = \frac{\mu_4}{\sigma^4} - 3$$

Kurtosis value equals to 3.0 indicates, the data distribution is mesokurtic, for kurtosis value greater than 3.0, it is called leptokurtic and for a lesser value than 3.0 the distribution is called platykurtic.

Steps to calculate Kurtosis in excel:

Select data points > Click formulas > Expand auto-sum drop down menu > Find function ‘KURT’ > click ‘Insert Function’ > Enter.

Excel provides an “*Analysis ToolPak*” add-in under the *Data* tab to generate a report of the Descriptive Statistics on the desired data.

For example, we have examination scores of 10 students in a class like the following. To generate descriptive statistics for these scores, follow the steps below.

Step 1: On the Data tab, in the Analysis group, click Data Analysis.

Step 2: Select Descriptive Statistics and click OK.

Step 3: Select the range B2:B11 as the Input Range.

Step 4: Select cell C1 as the Output Range.

Step 5: Make sure Summary statistics are checked.

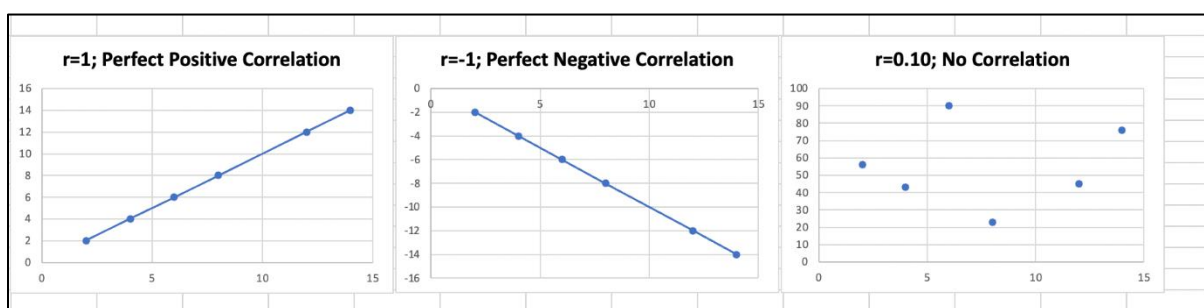
Step 6: Click ok.

1. Correlation and Regression Analysis: Correlation is the measurement of linear association between two variables. It is a measure that describes the strength and direction of a relationship between two variables. It is a commonly used measure in statistics, economics and social sciences for budgets, business plans etc. The correlation coefficient is used to measure the correlation between bivariate data which basically denotes the degree of linear association between two random variables.

In statistics, there are several types of correlation measures depending on the type of data you are working with. Here, we will focus on the most common one.

Pearson Product Moment Correlation (PPMC), popularly called Pearson Correlation, is used to evaluate linear relationships between data when a change in one variable is associated with a proportional change in the other variable.

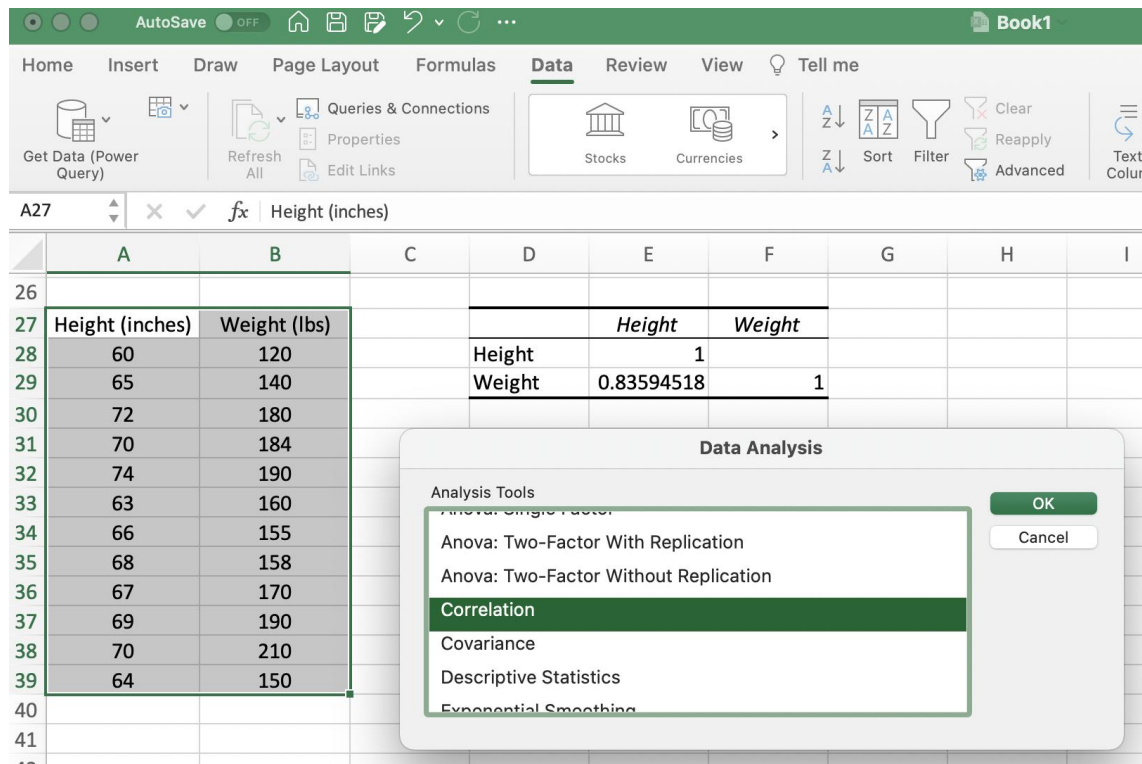
$$\text{Pearson Correlation Coefficient, } r = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 * \sum (y_i - \bar{y})^2}}$$



The correlation coefficient value always lies between -1 and 1 and it measures both the strength and direction of the linear relationship between the variables. Correlation coefficient of +1 means a perfect positive relationship, as value of one variable increases, value of other variable increases proportionally. Correlation coefficient value of -1 means a perfect negative relationship, with increase in the value of one variable, the other one decreases proportionally. A coefficient of 0 means no linear relationship between the two variables the data points are scattered all over the graph.

Steps to calculate Pearson Correlation Coefficient in Excel:

Select 'Data' tab > click 'Data Analysis' > Find Correlation from the given menus > Click ok > Select the input range > select output cell > Grouped by columns > click ok.



Regression analysis is used to estimate the relationship between two or more variables. Dependent variable is the main factor you want to study, understand, or predict. Independent variables are the factors that might influence the dependent variable. Regression analysis helps to understand how the dependent variable changes when one of the independent variables vary. Regression analysis can make it easier to predict future variable trends by analyzing the trajectory of the regression line. Simple linear regression model tries to establish a linear association between the dependent and the independent variable so that the outcome of the dependent variable can be predicted using the independent variables. The simple linear regression model uses the following equation:

$$Y = a + bX + \epsilon$$

where, Y = value of the dependent variable

X = value of the independent variable

a = intercept

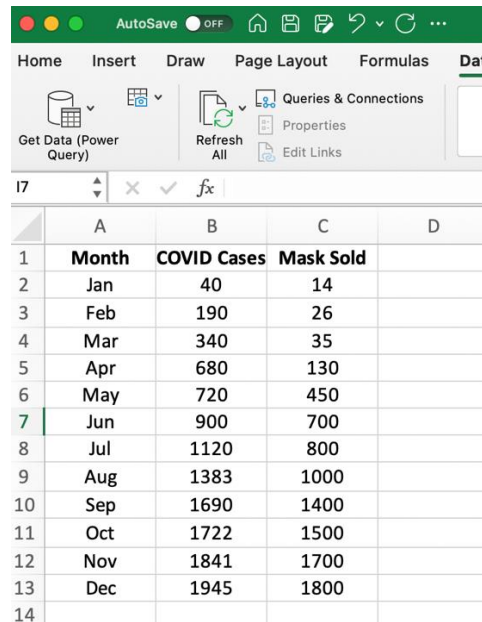
b = slope (regression line steepness)

ϵ = error component

Steps to perform Regression Analysis in Excel:

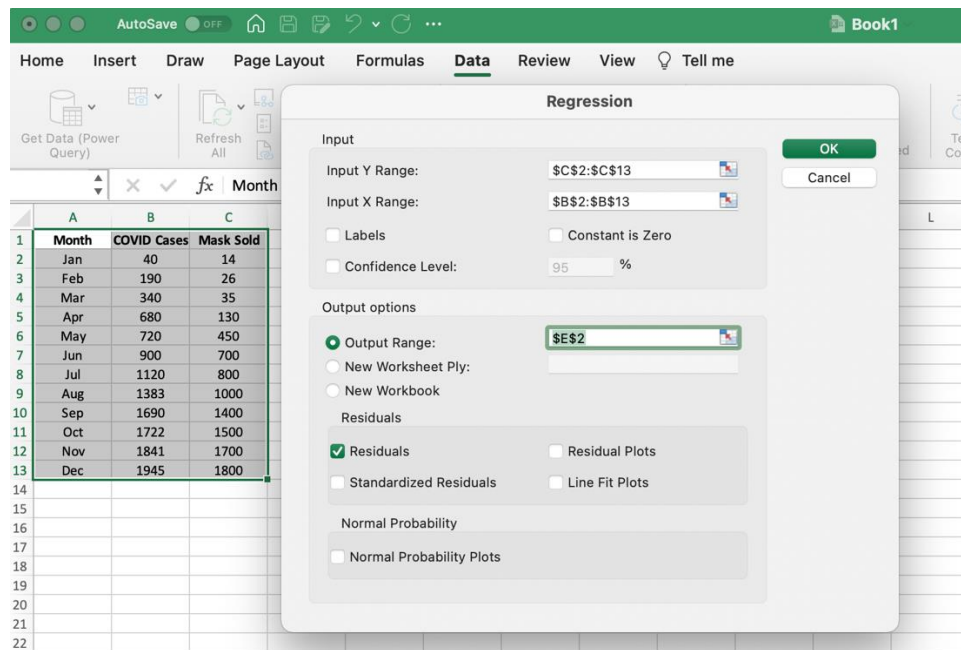
Step1: Let us consider the data values for the following two variables, COVID cases and masks sold and perform

a simple linear regression analysis in Excel considering number of Masks sold as the Y variable and number of COVID cases as X variable on which Y is dependent.



	A	B	C	D
1	Month	COVID Cases	Mask Sold	
2	Jan	40	14	
3	Feb	190	26	
4	Mar	340	35	
5	Apr	680	130	
6	May	720	450	
7	Jun	900	700	
8	Jul	1120	800	
9	Aug	1383	1000	
10	Sep	1690	1400	
11	Oct	1722	1500	
12	Nov	1841	1700	
13	Dec	1945	1800	
14				

Step2: Click on the 'Data' tab > Data Analysis > Select 'Regression' > click 'Ok'.



Step3: In the Regression dialog box select the Input Y Range, which is our dependent variable. In this case it is (C2:C13). Then select the Input X Range, independent variable. In this example, it is the number of COVID cases (B2:B13). Select the desired output range, here E2.

Click ok.

You get the following Output:

SUMMARY OUTPUT								
Regression Statistics								
Multiple R	0.980009172							
R Square	0.960417978							
Adjusted R Square	0.956459776							
Standard Error	141.8479509							
Observations	12							
ANOVA								
	<i>df</i>	<i>SS</i>	<i>MS</i>	<i>F</i>	<i>Significance F</i>			
Regression	1	4882119.838	4882119.838	242.639947	2.43153E-08			
Residual	10	201208.4118	20120.84118					
Total	11	5083328.25						
	<i>Coefficients</i>	<i>Standard Error</i>	<i>t Stat</i>	<i>P-value</i>	<i>Lower 95%</i>	<i>Upper 95%</i>	<i>Lower 95.0%</i>	<i>Upper 95.0%</i>
Intercept	-245.6307848	78.42517332	-3.13204006	0.01065345	-420.3729604	-70.88860911	-420.3729604	-70.888609
X Variable 1	0.994556473	0.063848147	15.57690428	2.4315E-08	0.852293936	1.136819009	0.852293936	1.13681901
RESIDUAL OUTPUT								
<i>Observation</i>	<i>Predicted Y</i>	<i>Residuals</i>						
1	-205.8485259	219.8485259						
2	-56.66505497	82.66505497						
3	92.51841592	-57.51841592						
4	430.6676166	-300.6676166						
5	470.4498755	-20.44987551						
6	649.4700406	50.52995942						
7	868.2724646	-68.27246456						
8	1129.840817	-129.8408169						
9	1435.169654	-35.16965395						
10	1466.995461	33.00453893						
11	1585.347681	114.6523187						
12	1688.781554	111.2184455						

- Interpreting the Output of Regression Analysis:

SUMMARY STATISTICS

Multiple R is the value of the Correlation Coefficient that measures the strength of a linear relationship between two variables. The larger the absolute value, the stronger the relationship.

R Square gives the Coefficient of Determination, which is used as an indicator of the goodness of fit. It shows how many points fall on the regression line. The R² value is calculated from the total sum of squares, more precisely, it is the sum of the squared deviations of the original data from the mean. In this example, R² is 0.96, which is very good. It means that 96% of our values fit the regression analysis model. In other words, 96% of the dependent variables (y-values) are explained by the independent variables (x-values). Generally, R Squared of 95% or more is considered a good fit.

Adjusted R Square gives the R square adjusted for the number of independent variables in the model. For multiple regression analysis, adjusted R square value is used instead of R square.

Standard Error is another goodness-of-fit measure that shows the precision of the fitted regression model. The smaller the number, the more certain one can be about the regression equation. It is an absolute measure that shows the average distance that the data points fall from the regression line.

Observations simply provides the total number of observations used to fit the model.

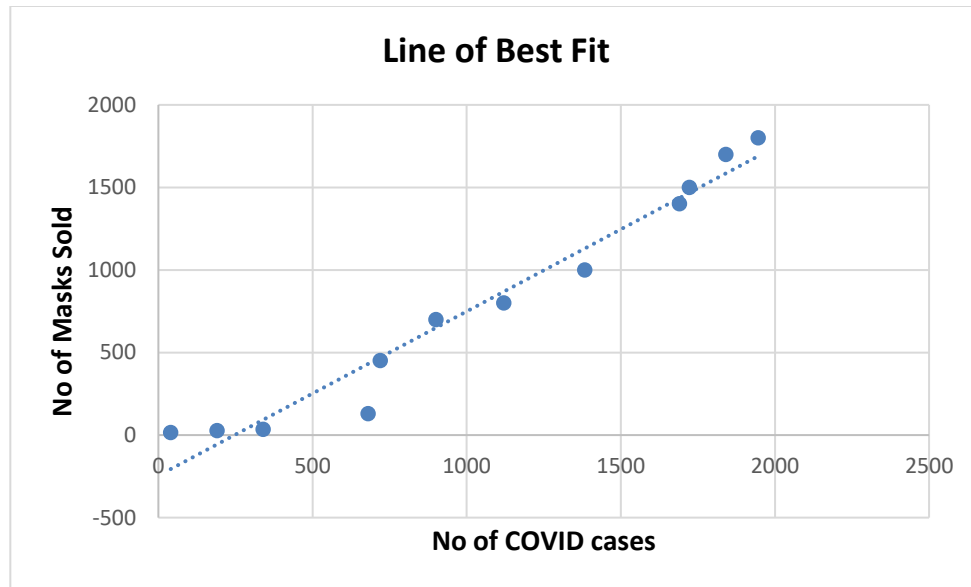
COEFFICIENTS

	Coefficients	Standard Error	t Stat	P-value	Lower 95%	Upper 95%	Lower 95.0%	Upper 95.0%
Intercept	-245.6307848	78.42517332	-3.132040063	0.01065345	-420.3729604	-70.88860911	-420.3729604	-70.888609
X Variable 1	0.994556473	0.063848147	15.57690428	2.4315E-08	0.852293936	1.136819009	0.852293936	1.13681901

Linear regression equation fitted was, $Y = b \cdot X + a$

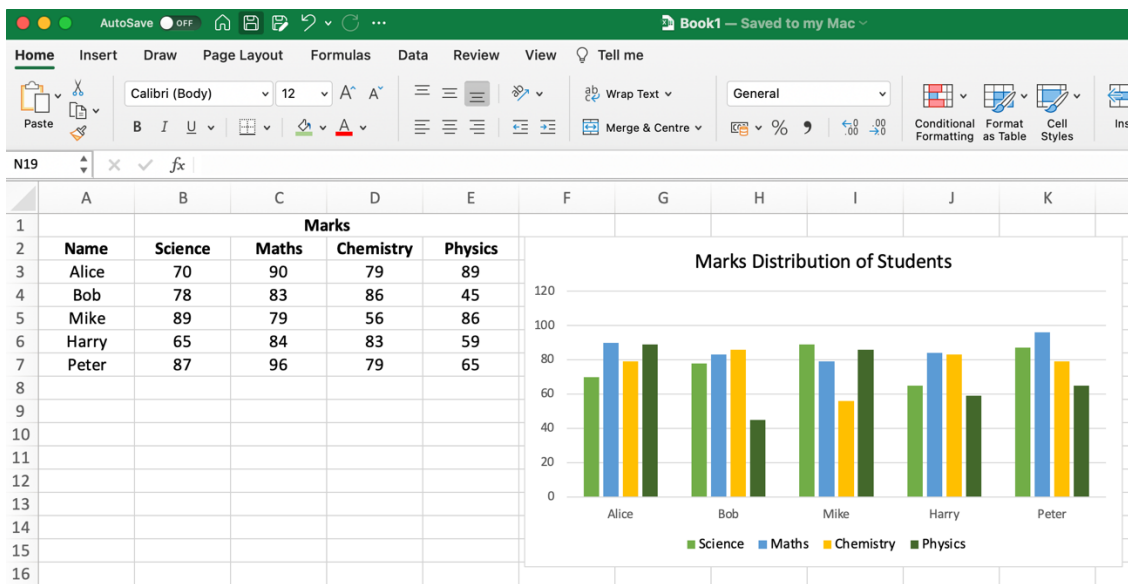
Here, Y = Mask sold; X = COVID cases; $b = 0.99$; $a = -245.63$

Therefore, $0.99 \cdot 190 - 245.63 = -57.53$



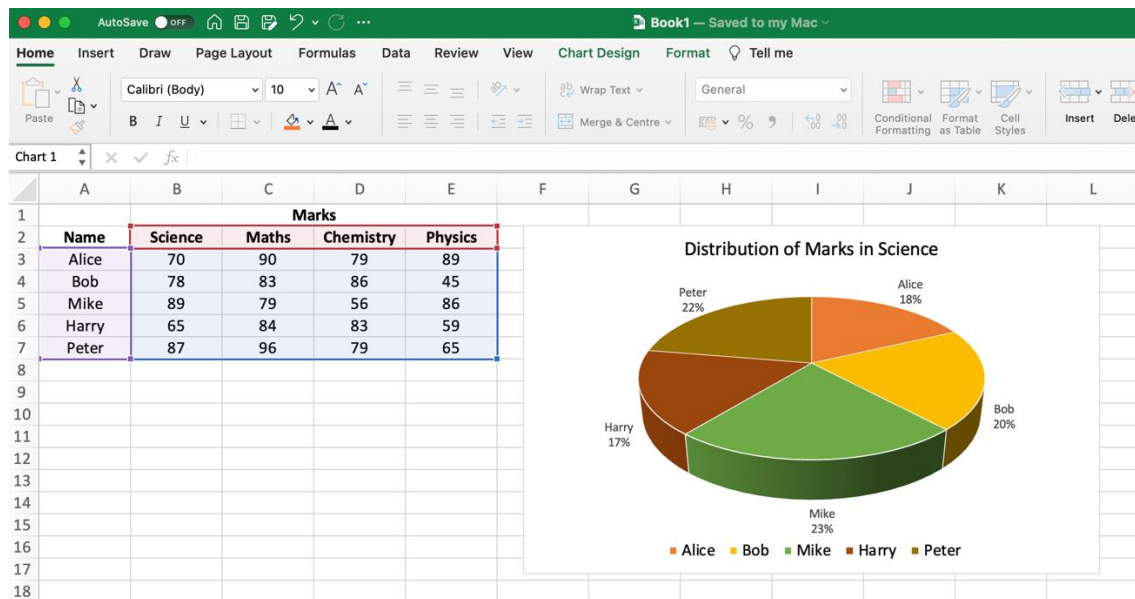
1. Create Charts/ Graphs in MS Excel:

Line Diagram: Select the data for which you want to plot the graph. Click 'Insert' tab > go to insert column chart > pick any chart of your preference. Excel will create the graphical representation as follows.



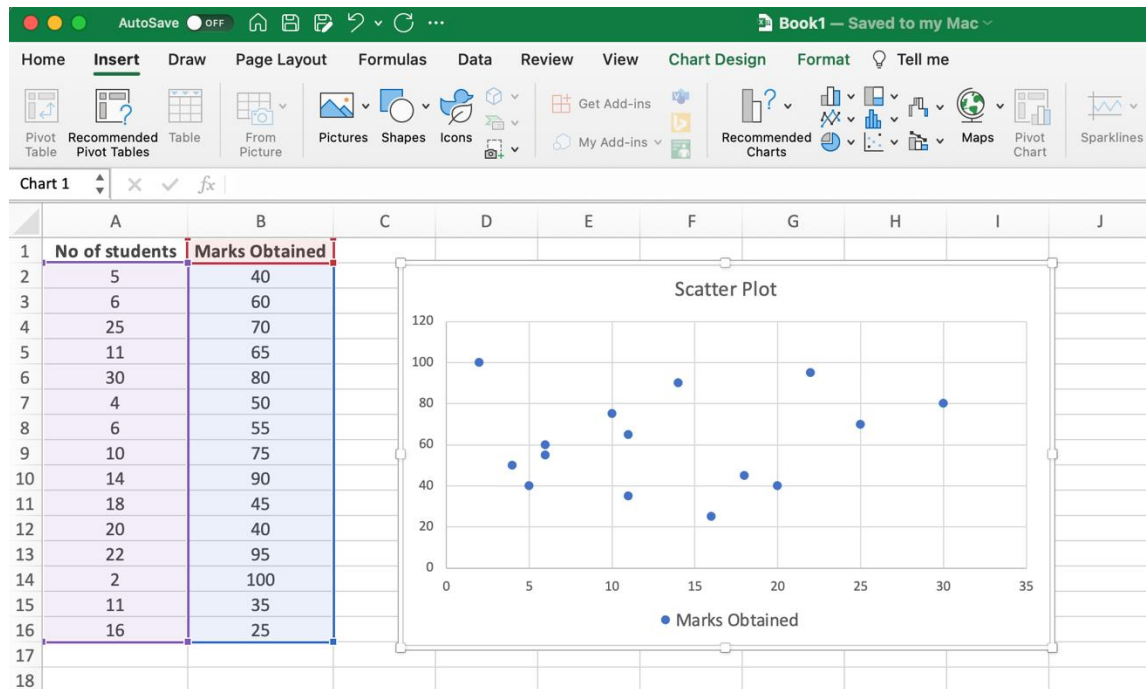
Pie chart: Pie chart represents the data in slices of a circle. Each slice represents the percentage contribution of each data section among the sum of individual data values.

Select the data for which you want to plot the pie chart. Click insert tab > go to insert pie or doughnut chart > pick any chart of your preference. Excel will create the graphical representation as following:

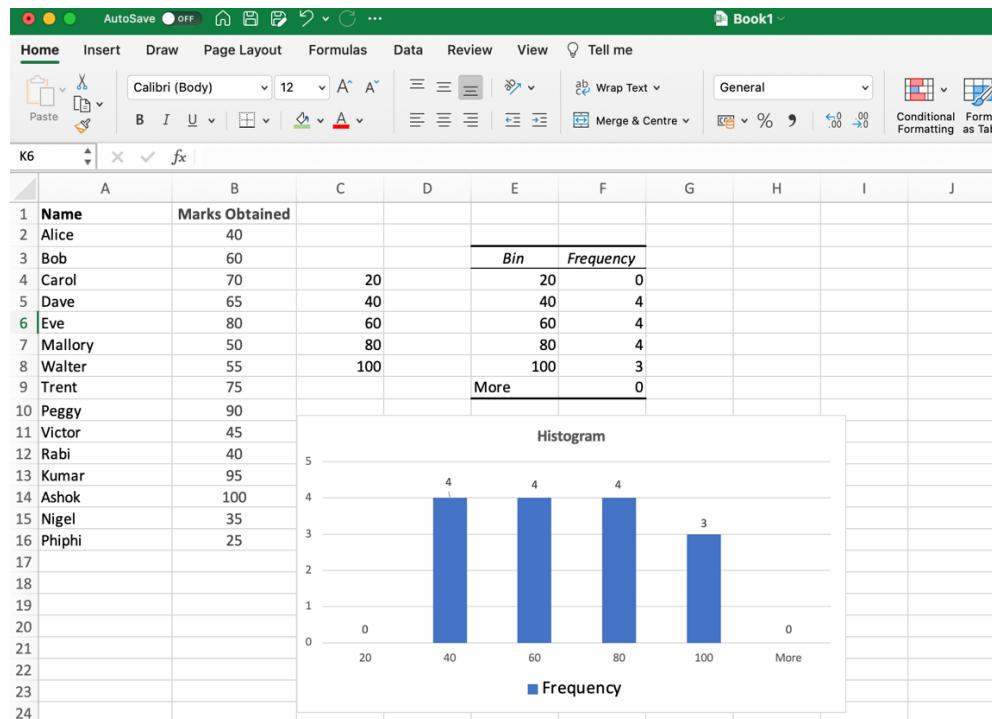


Scatter Diagram: Scatter charts are specifically used to show how one variable is related to another. There are seven scatter chart options: scatter, scatter with smooth lines and markers, scatter with smooth lines, scatter with straight lines and markers, scatter with straight lines, bubble, and 3-D bubble. For plotting a scatter chart, one needs data points for two or more variables.

Select the data > click insert tab > go to X Y Scatter chart > pick any chart of your preference. Excel will create the graphical representation as following:



Histogram: Select data > click on data tab > select data Analysis > click histogram > select



input range (B2:B16)> select bin (class intervals, here it is C4:C8) > check Chart Output > click ok. Excel will produce the frequency table against the specified bin value and also will create a histogram diagram like following.

2. Inferential Statistics:

Inferential statistics is used for estimating the population data by analysing the samples obtained from it. It helps in generalizing about the population by using different analytical tests and tools. Various sampling techniques are used to select random samples that will represent the population accurately. Some of the important methods are simple random sampling, stratified sampling, cluster sampling, and systematic sampling techniques.

Inferential statistics can be defined as a field of statistics that uses analytical tools for drawing conclusions about a population by examining random samples. In inferential statistics, a statistic is taken from the sample data (e.g., sample mean) that is used to make inferences about the population parameter (e.g., the population mean). One sample t-test is the most used one and sets a basic understanding of all other kinds of hypothesis testing methods.

One sample t-test:

The one-sample t test compares a given sample mean \bar{X} to a known or hypothesized value of the population mean μ_0 provided the population standard deviation σ is unknown. Excel does not have a built-in one-sample t test. However, the use of Excel functions and formulas makes the computations quite simple. The value of t -statistic can be calculated from the given formula:

$$t = \frac{\bar{X} - \mu_0}{s_{\bar{x}}}$$

where, \bar{X} is the sample mean, μ_0 is the known or hypothesized population mean and $s_{\bar{x}}$ is the standard error of mean. To calculate the t -statistic in excel we need to first find the following values.

Consider a sample of 12 young female adults, we have the measurement of their heights in inches. Let us assume the national average height of 18-year-old girls is 66.5 inches. We want to perform a one-sample T-test in Excel to determine if there is any significant difference between the heights of the sample data compared with the national average height (66.5 inches).

	A	B	C	D	E	F	G
1	Height (inches)						
2	65.78331	Mean	68.3242167				
3	71.51	Standard Deviation	1.64570038				
4	69.39	Count	12				
5	68.2166	Standard Error of Mean	1.1				
6	67.78781	Degrees of Freedom	11				
7	68.69784						
8	69.80204						
9	70.012						
10	67.902	Hypothesized Mean	66.5	given			
11	66.782						
12	66.487						
13	67.52	t-statistic	1.65				
14							
15		p-value	0.12717676				
16							

The null hypothesis and alternative hypothesis for this test are:

Null hypothesis: There is no significant difference between the heights of the sample, compared with the national average.

Alternative hypothesis: There is significant difference between the heights of the sample, compared with the national average.

First of all, compute mean, standard deviation, standard error, degrees of freedom to calculate the value of the t -statistic as shown in the above screenshot then in an empty cell, enter =TDIST (t, df, tails) to compute the p-value.

t – the cell containing the t-statistic

df – The cell containing the degrees of freedom.

tails –1 if you want to perform a one-tailed analysis, or 2 if you want to do a two-tailed analysis. p-value for this example is 0.127.

Let us assume alpha level is set at 0.05, then since the p-value is above the alpha level, we will accept the null hypothesis and reject the alternative hypothesis. In other words, there is no significant difference between the heights of the sample, compared with the national average.

Data Analysis using R Software

Soumen Pal¹, B N Mandal²

¹ICAR-Indian Agricultural Statistics Research Institute, New Delhi - 110 012

²ICAR-Indian Agricultural Research Institute, Jharkhand

¹soumen.pal@icar.gov.in

R is a free software environment for statistical computing and graphics. It compiles and runs on a wide variety of UNIX platforms, Windows and MacOS. R is a vehicle for newly developing methods of interactive data analysis. It has developed rapidly and has been extended by a large collection of packages.

R environment

The R environment provides an integrated suite of software facilities for data manipulation, calculation, and graphical display. It has

- a data handling and storage facility,
- a suite of operators for calculations on arrays and matrices,
- a large, integrated collection of intermediate tools for data analysis,
- graphical facilities for data analysis and display, and
- a well-developed, simple and effective programming language (called ‘S’) which includes conditionals, loops, user defined functions and input and output facilities.

Origin

R can be regarded as an implementation of the S language which was developed at Bell Laboratories by Rick Becker, John Chambers and Allan Wilks, and also forms the basis of the S-Plus systems. Robert Gentleman and Ross Ihaka of the Statistics Department of the University of Auckland started the project on R in 1995 and hence the name software has been named as ‘R’.

R was introduced as an environment within which many classical and modern statistical techniques can be implemented. A few of these are built into the base R environment, but many are supplied as packages. There are a number of packages supplied with R (called “standard” and “recommended” packages) and many more are available through the CRAN family of Internet sites (via <http://cran.r-project.org>) and

elsewhere.

Availability

Since R is an open source project, it can be obtained freely from the website www.r-project.org. One can download R from any CRAN mirror out of several CRAN (Comprehensive R Archive Network) mirrors. Latest available version of R is *R version 4.3.2* and it has been released on October 31 2023.

Installation

To install R in windows operating system, simply double click on the setup file. It will automatically install the software in the system.

Usage

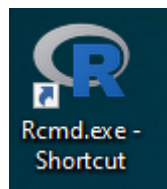
R can work under Windows, UNIX and Mac OS. In this note, we consider usage of R in Windows set up only.

Difference with other packages

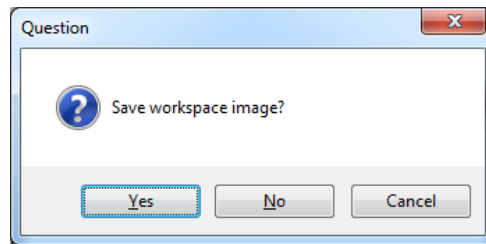
There is an important difference between R and the other statistical packages. In R, a statistical analysis is normally done as a series of steps, with intermediate results being stored in objects. Thus whereas SAS and SPSS will give large amount of output from a given analysis, R will give minimal output and store the results in an object for subsequent interrogation by further R functions.

Invoking R

If properly installed, usually R has a shortcut icon on the desktop screen and/or you can find it under Start|All Programs|R menu.

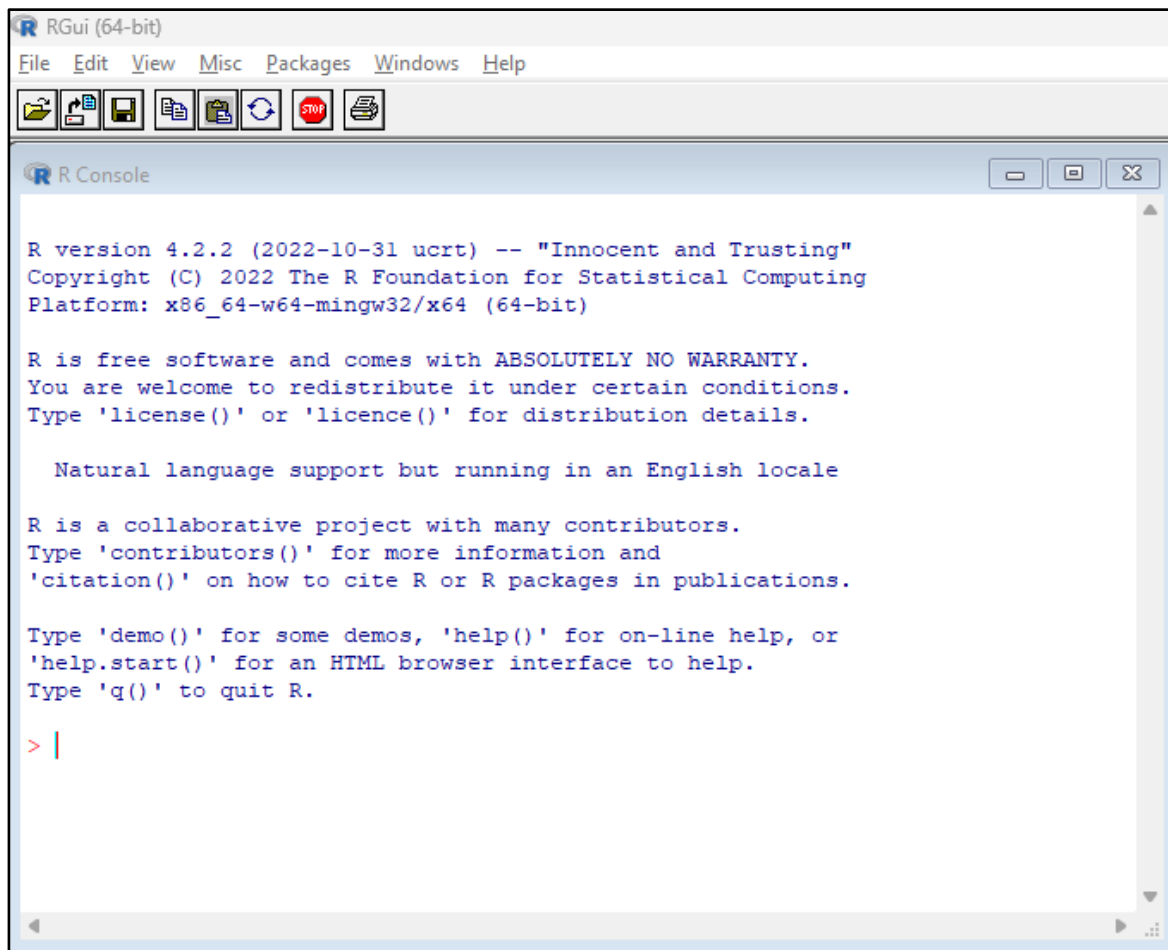


To quit R, type *q()* at the R prompt (*>*) and press Enter key. A dialog box will ask whether to save the objects you have created during the session so that they will become available next time when R will be invoked.



Windows of R

R has only one window and when R is started it looks like



R commands

- i. R commands are case sensitive, so X and x are different symbols and would refer to different variables.
- ii. Elementary commands consist of either expressions or assignments.
- iii. If an expression is given as a command, it is evaluated, printed and the value is lost.

- iv. An assignment also evaluates an expression and passes the value to a variable but the result is not automatically printed.
- v. Commands are separated either by a semi-colon (;), or by a newline.
- vi. Elementary commands can be grouped together into one compound expression by braces '{' and '}'.
- vii. Comments can be put almost anywhere, starting with a hashmark (#). Anything written after # marks to the end of the line is considered as a comment.
- viii. Window can be cleared of lines by pressing Ctrl + L keys.

Executing commands from or diverting output to a file

If commands are stored in an external file, say 'D:/commands.txt' they may be executed at any time in an R session with the command

```
> source("d:/commands.txt")
```

For Windows Source is also available on the File menu.

The function *sink()*,

```
> sink("d:/record.txt")
```

will divert all subsequent output from the console to an external file, 'record.txt' in D drive. The command

```
> sink()
```

restores it to the console once again.

Simple manipulations of numbers and vectors

R operates on named data structures. The simplest such structure is the numeric vector, which is a single entity consisting of an ordered collection of numbers. To set up a vector named x, say, consisting of five numbers, namely 10.4, 5.6, 3.1, 6.4 and 21.7, use the R command

```
> x <- c(10.4, 5.6, 3.1, 6.4, 21.7)
```

The function *c()* assigns the five numbers to the vector x. The assignment operator (<-) 'points' to the object receiving the value of the expression. One can use the '=' operator as an alternative.

A single number is taken as a vector of length one.

Assignments can also be made in the other direction, using the obvious change in the assignment operator. So the same assignment could be made using

> c(10.4, 5.6, 3.1, 6.4, 21.7) -> x

If an expression is used as a complete command, the value is printed. So now if we were to use the command

> 1/x

the reciprocals of the five values would be printed at the terminal.

The elementary arithmetic operators

+ addition

– subtraction

* multiplication

/ division

^ exponentiation

Arithmetic functions

log, exp, sin, cos, tan, sqrt,

Other basic functions

max(x) – maximum element of vector x,

min(x)- minimum element of vector x,

range (x) – range of the values of vector x ,

length(x) - the number of elements in x,

sum(x) - the total of the elements in x,

prod(x) – product of the elements in x

mean(x) – average of the elements of x

var(x) – sample variance of the elements of (x)

sort(x) – returns a vector with elements sorted in increasing order.

Logical operators

< - less than

<= less than or equal to

> greater than

`>=` greater than or equal to

`==` equal to

`!=` not equal to.

Other objects in R

Matrices or arrays - multi-dimensional generalizations of vectors.

Lists - a general form of vector in which the various elements need not be of the same type, and are often themselves vectors or lists.

Functions - objects in R which can be stored in the project's workspace. This provides a simple and convenient way to extend R.

Matrix facilities

A matrix is just an array with two subscripts. R provides many operators and functions those are available only for matrices. Some of the important R functions for matrices are

`t(A)` – transpose of the matrix A

`nrow(A)` – number of rows in the matrix A

`ncol(A)` – number of columns in the matrix A

`A %*% B` – Cross product of two matrices A and B

`A*B` – element by element product of two matrices A and B

`diag(A)` – gives a vector of diagonal elements of the square matrix A

`diag(a)` – gives a matrix with diagonal elements as the elements of vector a

`eigen(A)` – gives eigen values and eigen vectors of a symmetric matrix A

`rbind(A,B)` – concatenates two matrix A and B by appending B matrix below A matrix (They should have same number of columns)

`cbind(A, B)` - concatenates two matrix A and B by appending B matrix in the right of A matrix (They should have same number of rows)

Data frame

Data frame is an array consisting of columns of various mode (numeric, character, etc). Small to moderate size data frame can be constructed by `data.frame()` function. For example, following is an illustration how to construct a data frame from the car data*:

Make	Model	Cylinder	Weight	Mileage	Type
Honda	Civic	V4	2170	33	Sporty
Chevrolet	Beretta	V4	2655	26	Compact
Ford	Escort	V4	2345	33	Small
Eagle	Summit	V4	2560	33	Small
Volkswagen	Jetta	V4	2330	26	Small
Buick	Le Sabre	V6	3325	23	Large
Mitsubishi	Galant	V4	2745	25	Compact
Dodge	Grand Caravan	V6	3735	18	Van
Chrysler	New Yorker	V6	3450	22	Medium
Acura	Legend	V6	3265	20	Medium

```
> Make<-c("Honda","Chevrolet","Ford","Eagle","Volkswagen","Buick","Mitsbusihi",
+ "Dodge","Chrysler","Acura")
> Model=c("Civic","Beretta","Escort","Summit","Jetta","Le Sabre","Galant",
+ "Grand Caravan","New Yorker","Legend")
```

Note that the plus sign (+) in the above commands are automatically inserted when the carriage return is pressed without completing the list. Save some typing by using *rep()* command. For example, *rep("V4",5)* instructs R to repeat V4 five times.

```
> Cylinder<-c(rep("V4",5),"V6","V4",rep("V6",3))
> Cylinder
[1] "V4" "V4" "V4" "V4" "V4" "V6" "V4" "V6" "V6" "V6"
> Weight<-c(2170,2655,2345,2560,2330,3325,2745,3735,3450,3265)
> Mileage<-c(33,26,33,33,26,23,25,18,22,20)
> Type<-c("Sporty","Compact",rep("Small",3),"Large","Compact","Van",rep("Medium",2))
```

Now *data.frame()* function combines the six vectors into a single data frame.

```
> Car<-data.frame(Make,Model,Cylinder,Weight,Mileage,Type)
> Car
  Make      Model Cylinder Weight Mileage  Type
1  Honda      Civic      V4   2170     33 Sporty
2 Chevrolet Beretta      V4   2655     26 Compact
3   Ford    Escort      V4   2345     33  Small
4  Eagle   Summit      V4   2560     33  Small
5 Volkswagen   Jetta      V4   2330     26  Small
6   Buick  Le Sabre      V6   3325     23  Large
7 Mitsbusihi  Galant      V4   2745     25 Compact
8   Dodge Grand Caravan      V6   3735     18   Van
9  Chrysler New Yorker      V6   3450     22 Medium
10   Acura   Legend      V6   3265     20 Medium

> names(Car)
[1] "Make"      "Model"     "Cylinder"  "Weight"    "Mileage"   "Type"
```

Just as in matrix objects, partial information can be easily extracted from the data frame:

```
> Car[1,]

  Make Model Cylinder Weight Mileage   Type
1 Honda Civic       V4   2170     33 Sporty
```

In addition, individual columns can be referenced by their labels:

```
> Car$Mileage
[1] 33 26 33 33 26 23 25 18 22 20
> Car[,5]      #equivalent expression
> mean(Car$Mileage) #average mileage of the 10 vehicles
[1] 25.9
> min(Car$Weight)
[1] 2170
```

table() command gives a frequency table:

```
> table(Car$Type)

Compact   Large   Medium   Small   Sporty   Van
      2       1       2       3       1       1
```

If the proportion is desired, type the following command instead:

```
> table(Car$Type)/10

Compact   Large   Medium   Small   Sporty   Van
    0.2     0.1     0.2     0.3     0.1     0.1
```

Note that the values were divided by 10 because there are that many vehicles in total. If you don't want to count them each time, the following does the trick:

```
> table(Car$Type)/length(Car$Type)
```

Cross tabulation is very easy, too:

```
> table(Car$Make, Car$Type)

      Compact Large Medium Small Sporty Van
Acura      0      0      1      0      0      0
Buick      0      1      0      0      0      0
Chevrolet  1      0      0      0      0      0
Chrysler   0      0      1      0      0      0
Dodge      0      0      0      0      0      1
Eagle      0      0      0      1      0      0
Ford       0      0      0      1      0      0
Honda      0      0      0      0      1      0
Mitsubishi 1      0      0      0      0      0
Volkswagen 0      0      0      1      0      0
```


What if you want to arrange the data set by vehicle weight? *order()* gets the job done.

```
> i<-order(Car$Weight);i
```

```
[1] 1 5 3 4 2 7 10 6 9 8
```

```
> Car[i,]
```

	Make	Model	Cylinder	Weight	Mileage	Type
1	Honda	Civic	V4	2170	33	Sporty
5	Volkswagen	Jetta	V4	2330	26	Small
3	Ford	Escort	V4	2345	33	Small
4	Eagle	Summit	V4	2560	33	Small
2	Chevrolet	Beretta	V4	2655	26	Compact
7	Mitsubishi	Galant	V4	2745	25	Compact
10	Acura	Legend	V6	3265	20	Medium
6	Buick	Le Sabre	V6	3325	23	Large
9	Chrysler	New Yorker	V6	3450	22	Medium
8	Dodge	Grand Caravan	V6	3735	18	Van

Creating/editing data objects

```
>y<-c(1,2,3,4,5);y
```

```
[1] 1 2 3 4 5
```

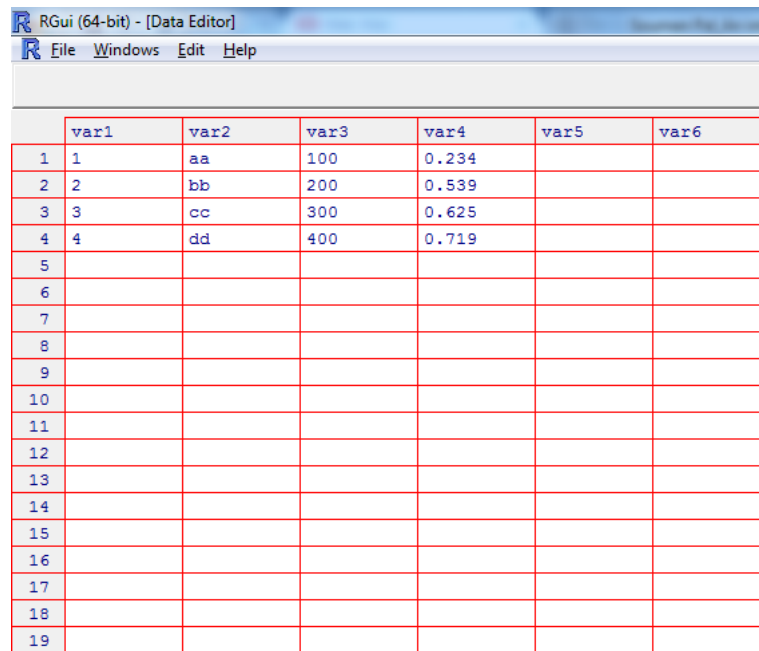
If you want to modify the data object, use *edit()* function and assign it to an object. For example, the following command opens R Editor for editing.

```
> y<-edit(y)
```

If you prefer entering the data.frame in a spreadsheet style data editor, the following command invokes the built-in editor with an empty spreadsheet.

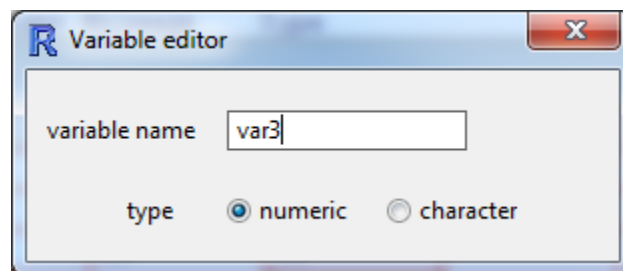
```
> data1<-edit(data.frame())
```


After entering a few data points, it looks like this:



	var1	var2	var3	var4	var5	var6
1	1	aa	100	0.234		
2	2	bb	200	0.539		
3	3	cc	300	0.625		
4	4	dd	400	0.719		
5						
6						
7						
8						
9						
10						
11						
12						
13						
14						
15						
16						
17						
18						
19						

You can also change the variable name by clicking once on the cell containing it. Doing so opens a dialog box:



When finished, click  in the upper right corner of the dialog box to return to the Data Editor window. Close the Data Editor to return to the R command window (R Console). Check the result by typing:

```
> data1
```

Reading data from files

When data files are large, it is better to read data from external files rather than entering data through the keyboard. To read data from an external file directly, the external file should be arranged properly.

The first line of the file should have a name for each variable. Each additional line of the file has the values

for each variable.

Input file form with names and row labels:

Price	Floor	Area	Rooms	Age	isNew
52.00	111.0	830	5	6.2	no
54.75	128.0	710	5	7.5	no
57.50	101.0	1000	5	4.2	yes
57.50	131.0	690	6	8.8	no
59.75	93.0	900	5	1.9	yes
...					

By default numeric items (except row labels) are read as numeric variables and non-numeric variables, such as `isNew` in the example, as factors. This can be changed if necessary.

The function `read.table()` can then be used to read the data frame directly

```
> HousePrice <- read.table("d:/houses.data", header = TRUE)
```

Reading comma delimited data

The following commands can be used for reading comma delimited data into R.

`read.csv(filename)` This command reads a .CSV file into R. You need to specify the exact filename with path.

`read.csv(file.choose())` This command reads a .CSV file but the `file.choose()` part opens up an explorer type window that allows you to select a file from your computer. By default, R will take the first row as the variable names.

`read.csv(file.choose(), header=T)`

This reads a .CSV file, allowing you to select the file, the header is set explicitly. If you change to `header=F` then the first row will be treated like the rest of the data and not as a label.

Storing variable names

Through `read.csv()` or `read.table()` functions, data along with variable labels is read into R memory. However, to read the variables' names directly into R, one should use `attach(dataset)` function. For example,

```
>attach(HousePrice)
```

causes R to directly read all the variables' names eg. Price, Floor, Area etc. it is a good practice to use the *attach(datafile)* function immediately after reading the *datafile* into R.

Packages

All R functions and datasets are stored in packages. The contents of a package are available only when the package is loaded. This is done to run the codes efficiently without much memory usage. To see which packages are installed at your machine, use the command

```
> library()
```

To load a particular package, use a command like

```
> library(forecast)
```

Users connected to the Internet can use the *install.packages()* and *update.packages()* functions to install and update packages. Use *search()* to display the list of packages that are loaded.

Standard packages

The standard (or base) packages are considered part of the R source code. They contain the basic functions those allow R to work with the datasets and standard statistical and graphical functions. They should be automatically available in any R installation.

Contributed packages and CRAN

There are a number of contributed packages for R, written by many authors. Various packages deal with various analyses. Most of the packages are available for download from CRAN (<https://cran.r-project.org/web/packages/>), and other repositories such as Bioconductor (<http://www.bioconductor.org/>). The collection of available packages changes frequently. As on June07, 2019, the CRAN package repository contains 14346 available packages.

Getting Help

Complete help files in HTML and PDF forms are available in R. To get help on a particular command/function etc., type *help (command name)*. For example, to get help on function 'mean', type *help(mean)* as shown below

```
> help(mean)
```

This will open the help file with the page containing the description of the function mean.

Another way to get help is to use “?” followed by function name. For example,

```
>?mean
```

will open the same window again.

In this lecture note, all R commands and corresponding outputs are given in `Courier New` font to differentiate from the normal texts. Since R is case-sensitive, i.e. typing *Help(mean)*, would generate an error message,

```
> Help(mean)
```

```
Error in Help(mean) : could not find function "Help"
```

Further Readings

Various documents are available in <https://cran.r-project.org/manuals.html> from beginners’ level to most advanced level. The following manuals are available in pdf form:

1. An Introduction to R
2. R Data Import/Export
3. R Installation and Administration
4. Writing R Extensions
5. The R language definition
6. R Internals
7. The R Reference Index

RStudio

RStudio is an integrated development environment (IDE) that allows to interact with R more readily. RStudio is similar to the standard RGui, but is considerably more user friendly. It has more drop-down menus, windows with multiple tabs, and many customization options.

Installation of RStudio

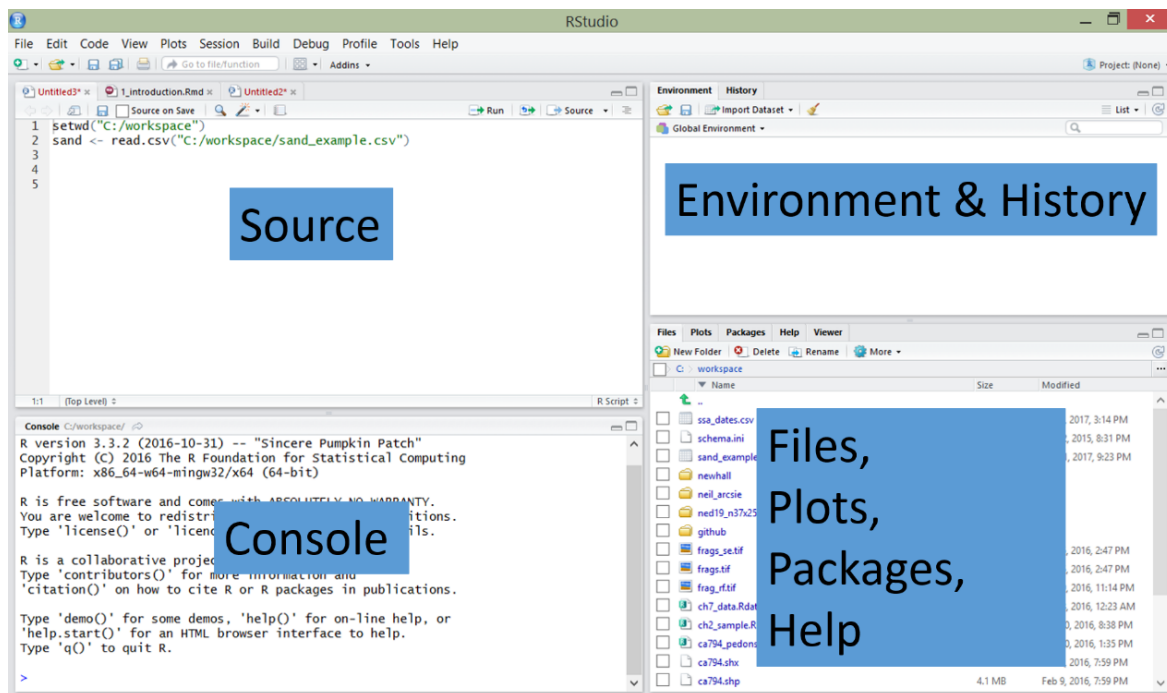
RStudio requires R 3.0.1+ that means R software should be pre-installed before using RStudio.

RStudio requires a 64-bit operating system, and works exclusively with the 64 bit version of R. If you are on a 32 bit system or need the 32 bit version of R, you can use an older version of RStudio

(<https://www.rstudio.com/products/rstudio/older-versions/>). RStudio free desktop version can be downloaded from the following link:

<https://www.rstudio.com/products/rstudio/download/#download>

The first time RStudio is opened, three windows are seen. A forth window is hidden by default, but can be opened by clicking the **File** drop-down menu, then **New File**, and then **R Script**.



Importing Data in R Studio

1. Click on the import dataset button in the top-right section under the environment tab. Select the file you want to import and then click open. The Import Dataset dialog will appear as shown below
2. After setting up the preferences of separator, name and other parameters, click on the Import button. The dataset will be imported in R Studio and assigned to the variable name as set before.

Import Dataset

Name: data1

Encoding: Automatic

Heading: ☒ Yes ☐ No

Row names: Automatic

Separator: Tab

Decimal: Period

Quote: Double quote (")

Comment: #

na.strings: NA

☒ Strings as factors

Input File

```
x y
1 11
2 22
3 33
4 44
5 55
6 66
7 34
8 99
9 62
10 99
```

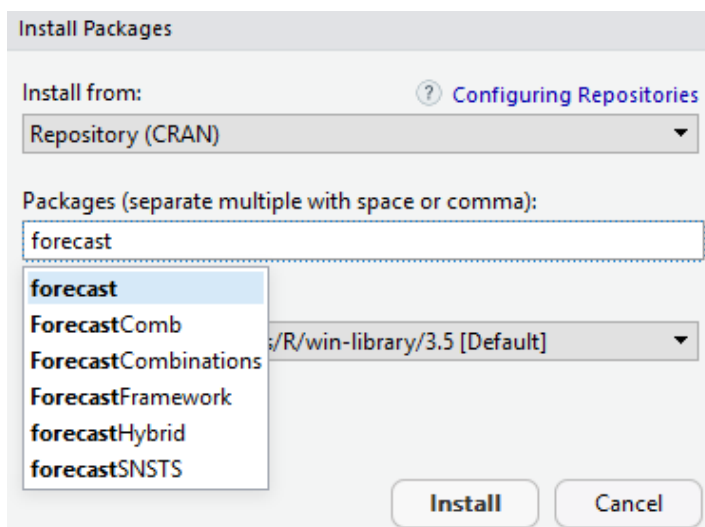
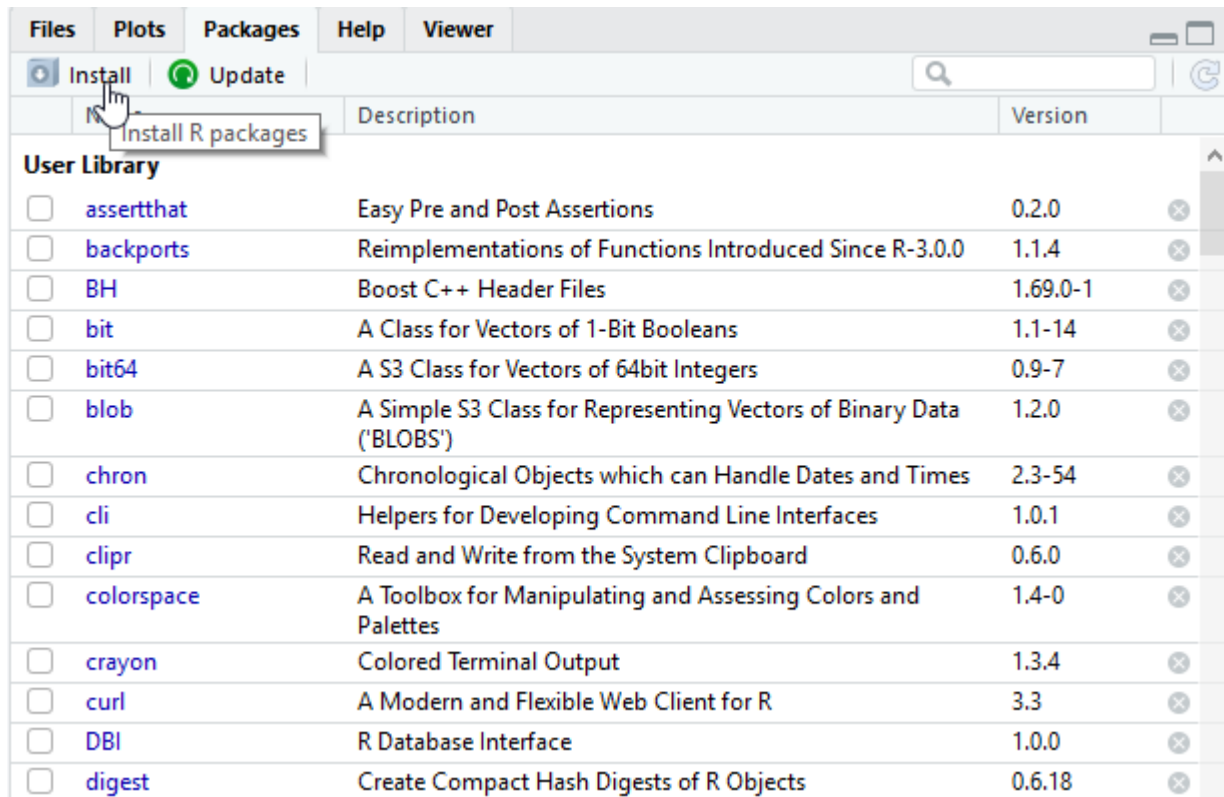
Data Frame

```
x y
1 11
2 22
3 33
4 44
5 55
6 66
7 34
8 99
9 62
10 99
```

Import Cancel

Installing Packages in RStudio

Within the **Packages** tab, a list of all the packages currently installed on the working computer and 2 buttons labeled either “Install” or “Update” are seen. To install a new package simply select the Install button. It is possible to install one or more than one packages at a time by simply separating them with a comma.



Loading Packages in RStudio

Once a package is installed, it must be loaded into the R session to be used.

Files

Plots

Packages

Help

Viewer

Install

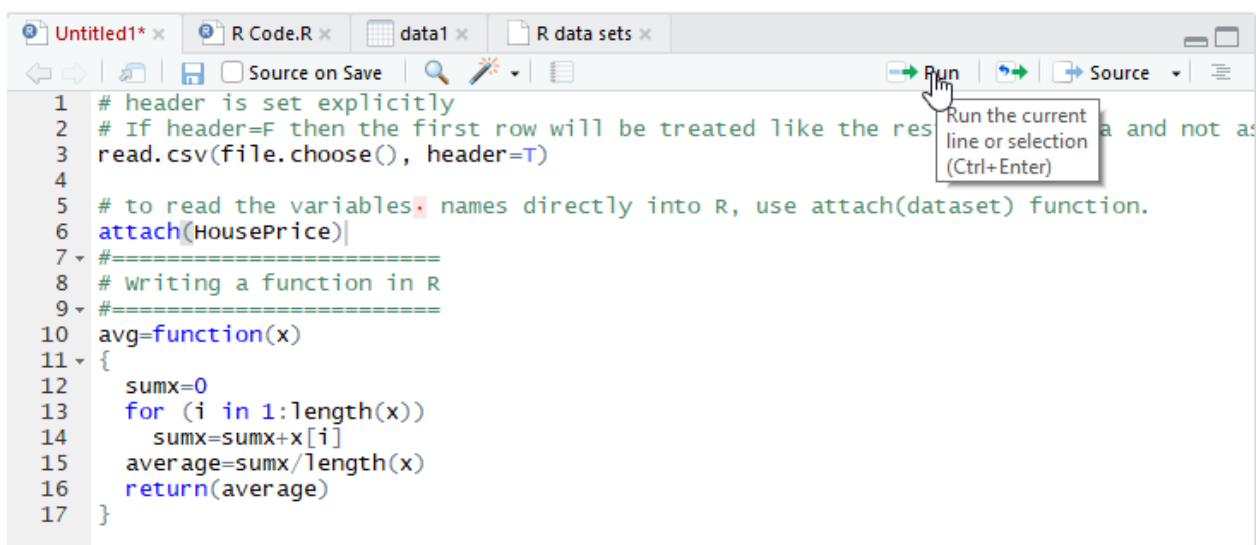
Update

	Name	Description	Version	
		Theory Group (Formerly: E1071), TU Wien		
<input type="checkbox"/>	ellipsis	Tools for Working with ...	0.2.0.1	
<input type="checkbox"/>	fansi	ANSI Control Sequence Aware String Functions	0.4.0	
<input type="checkbox"/>	forcats	Tools for Working with Categorical Variables (Factors)	0.4.0	
<input checked="" type="checkbox"/>	forecast	Forecasting Functions for Time Series and Linear Models	8.5	
<input type="checkbox"/>	fracdiff	Fractionally differenced ARIMA aka ARFIMA(p,d,q) models	1.4-2	
<input type="checkbox"/>	ggplot2	Create Elegant Data Visualisations Using the Grammar of Graphics	3.1.0	

Writing Scripts in RStudio

RStudio's Source Tabs serve as a built-in text editor. Prior to executing R functions at the Console, commands are typically written down (or scripted). To write a script, simply open a new R script file by clicking File>New File>R Script. Within the text editor type out a sequence of functions.

- Place each function (e.g. `read.csv()`) on a separate line.
- If a function has a long list of arguments, place each argument on a separate line.
- A command can be executed from the text editor by placing the cursor on a line and typing Ctrl + Enter, or by clicking the Run button.
- An entire R script file can be executed by clicking the Source button.



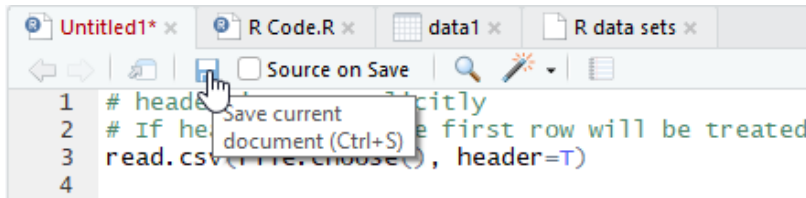
Saving R files in RStudio

In R, several types of files can be saved to keep track of the work performed. The file types include: script,

workspace, history and graphics.

R script (.R)

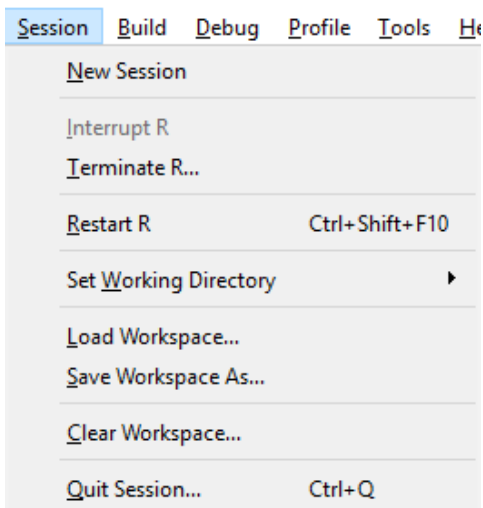
An R script is a text file of R commands that have been typed. To save R scripts in RStudio, click the save button from R script tab. Save scripts with the .R extension.



To open an R script, click the file icon.

Workspace (.Rdata)

The R workspace consists of all the data objects created or loaded during the R session. It is possible to save or load the workspace at any time during the R session from the menu by clicking Session > Save Workspace As..., or the save button on the Environment Tab.



R history (.Rhistory)

Rhistory file is a text file that lists all of the commands that have been executed. It does not keep a record of the results. To load or save R history from the History Tab click the **Open File** or **Save** button.

```

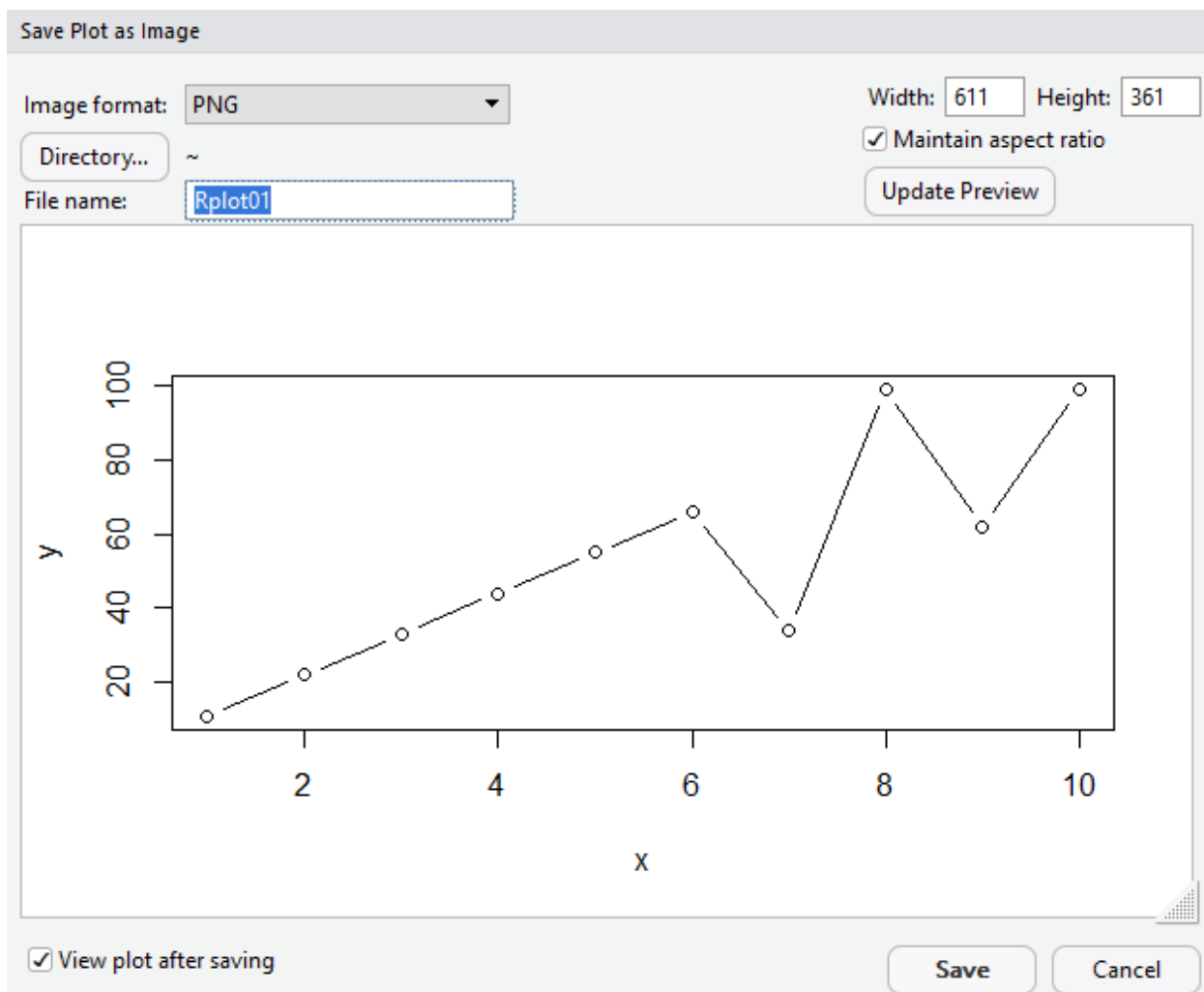
Environment History Connections
avg=function(x)
{
  sumx=0
  for (i in 1:length(x))
  sumx=sumx+x[i]
  average=sumx/length(x)
  return(average)
}

```

R Graphics

Graphic outputs can be saved in various formats like pdf, png, jpeg, bmp etc.

To save a graphic: (1) Click the **Plots** Tab window, (2) click the **Export** button, (3) **Choose** desired format, (4) **Modify** the export settings as desired and (4) click **Save**.



References

1. GitHub (2023). https://ncss-tech.github.io/stats_for_soil_survey/book/intro.html
2. Scribd (2023). <https://www.scribd.com/document/320460543/Basic-Tutorial-R-Studio-Tutorial>
3. DataAnalytics.org.uk (2023). <https://www.dataanalytics.org.uk/getting-started-with-r/>
4. CRAN (2023). <https://www.cran.r-project.org>
5. Posit Software (2023). <https://www.rstudio.com>
6. Matloff, N. (2011). *The art of R programming: A tour of statistical software design*. No Starch Press.
7. Venables, W. N., Smith, D. M. and R Development Core Team (2009). An introduction to R: Notes on R: A programming Environment for Data Analysis and Graphics, version 1.7. 1.

Descriptive Statistics and Exploratory Data Analysis

Achal Lama

ICAR-Indian Agricultural Statistics Research Institute, New Delhi - 110 012

achal.lama@icar.gov.in

“Statistics is defined as the science of collection, presentation, analysis and interpretation of numerical data”-Croxtton and Cowden

Scope:

- ❖ Biology
- ❖ Education
- ❖ Economics
- ❖ Business management
- ❖ Policy decisions

Recent developments in the field of computer technology have enabled Statistics to integrate their models into information systems, making Statistics a part of decision making procedures.

Role of Statistics:

- Language of the data
- Provides scientific way to extract and retrieve the information hidden inside the data
- Provides forecasting
- ❖ It cannot do miracles
- ❖ It cannot change the process or phenomenon
- ❖ Identify objective(s) of Statistical Analysis
- ❖ Generate data to achieve the objectives:
 - ❖ Laboratory/ Field experiment
 - ❖ Survey
 - ❖ Primary Data

- ❖ Secondary Data
- ❖ Use appropriate Statistical tool
- ❖ Make inferences about the results obtained

Observation:

The units on which we measure the data is called observation

Ex: Height of the participants

Population:

It is a collection of all the units

Ex: Total no. of participants in this training program

Sample:

It is a subset/fraction of the population (representative) taken randomly

Ex: Recording height of 10 participants (sample size=10)

Variable:

It is represented by X, Y, Z, \dots

Values collected are represented by x, y, z, \dots

Ex: X = Height

$$x_1 = 150 \text{ cm}, x_2 = 155 \text{ cm}, x_3 = 160 \text{ cm}, \dots$$

Univariate:

Single variable of interest

Multivariate:

More than one variable of interest

Ex: X = Height, Y = Weight, Z = Age

Variables

➤ **Quantitative**

It represents some measurable quantities and be ordered in logical or natural way.

Ex: Price of vegetables/Kg (30,40,50,...)

Height (150cm, 160cm, 155cm,...)

✓ **Discrete**

Takes finite number of values (countable)

Ex: No. of tillers/plant, no. of fruits/tree,...(10,20,15,...)

✓ **Continuous**

Takes infinite number of values (uncountable and measurable)

Ex: Height of a plant, length of a leaf, ...(1.43m, 2.52cms)

➤ **Qualitative**

Cannot be ordered in logical or natural way

Ex: Performance (Good, Average, Excellent,...)

Hair Color (Black, Brown, White,...)

Grouped Data:

Sometimes the original values of the data are grouped or data is available in form of groups

Ex: Plant height (154cms, 155cms, 151cms, 162cms, 166cms, 168cms, 173cms, 175cms)

Group 1: 150-160; 3

Group 2: 161-170; 3

Group 3: 171-180; 2

Original values are lost and we have only the idea about the size of the group

Classification of data:

Process of arranging data into groups or classes based on similarity and resemblance

Absolute and Relative frequency:

Let us consider that 10 students have appeared for an exam

Pass (P) and Fail (F)

P P P F F P P F P P

a for P and b for F

a=7

b=3

a and b are absolute frequency

$ra = a/a+b = 7/10 = 0.7$ or 70%

$rb = b/a+b = 3/10 = 0.3$ or 30%

ra and rb are the relative frequency

R code:

Enter data as x

table(x) # Absolute frequency

table(x)/length(x) # Relative frequency

Absolute and Relative frequency:

R code:

Enter data as x (some name as per your convenience)

P=1 and F=2

1 1 1 2 2 1 1 2 1 1

result=c(1 1 1 2 2 1 1 2 1 1)

table(result) # **Absolute frequency**

table(result)/length(result) # **Relative frequency**

Frequency Distribution:

Arrangement of ungrouped data into groups is called **frequency distribution of data**

Classifies the data into different classes by dividing the entire range of values of variables into suitable number of groups called **class**

Defined lower and upper boundary of a class is called the lower limit and upper limit respectively

Difference between the limits is called the width of a class or class interval

Value of the variate lies in between the lower and upper limits

Ex: Class intervals:

10-15: Mid value= $10+15/2 = 12.5$

15-20 : Mid value= $15+20/2 = 17.5$

Number of observations in a class is called Absolute frequency or frequency

Absolute frequency divided by the total frequency of all classes is relative frequency

Classifies the data into different classes by dividing the entire range of values of variables into suitable number of groups called **class**

Cumulative frequency of a class is the number of observations less than or equal to upper limit of a class

Example: Let us consider the number of fruits /tree of a mango orchard

nf=32,35,45,83,74,55,68,38,35,55,66,65,42,68,72,84,67,36,42,58

range(nf) # returns the maximum and minimum

[1] 32 84

This information allows us to divide the data in class following intervals:

31-40 41-50 51-60 61-70 71-80 and 81-90

The class width being 10 and a sequence from 30-90

> **breaks=seq(30,90,by=10)**

> breaks

```
[1] 30 40 50 60 70 80 90
> cut.nf=cut(nf,breaks,right=FALSE)
> cut.nf
[1] [30,40) [30,40) [40,50) [80,90) [70,80) [50,60) [60,70) [30,40) [30,40) [50,60) [60,70) [60,70) [40,50)
[60,70), [70,80) [80,90) [60,70) [30,40) [40,50) [50,60)
Levels: [30,40) [40,50) [50,60) [60,70) [70,80) [80,90)
#Absolute frequency
> table(cut.nf)
cut.nf
[30,40) [40,50) [50,60) [60,70) [70,80) [80,90)
      5      3      3      5      2      2
> cbind(table(cut.nf)) # For tabular view
      [,1]
[30,40)  5
[40,50)  3
[50,60)  3
[60,70)  5
[70,80)  2
[80,90)  2
> rf=table(cut.nf)/length(cut.nf) # Relative frequency
cut.nf
[30,40) [40,50) [50,60) [60,70) [70,80) [80,90)
  0.25  0.15  0.15  0.25  0.10  0.10
> cbind(rf) #Tabular form
      rf
[30,40) 0.25
[40,50) 0.15
[50,60) 0.15
[60,70) 0.25
[70,80) 0.10
[80,90) 0.10
Cumulative Distribution Function (CDF):
```

It gives us an idea about the cumulative frequencies up to a certain point

The cumulative frequencies are computed by the function **cumsum**

```
> cbind(cumsum(table(cut.nf)))
```

```
  [,1]
```

```
[30,40)  5
```

```
[40,50)  8
```

```
[50,60) 11
```

```
[60,70) 16
```

```
[70,80) 18
```

```
[80,90) 20
```

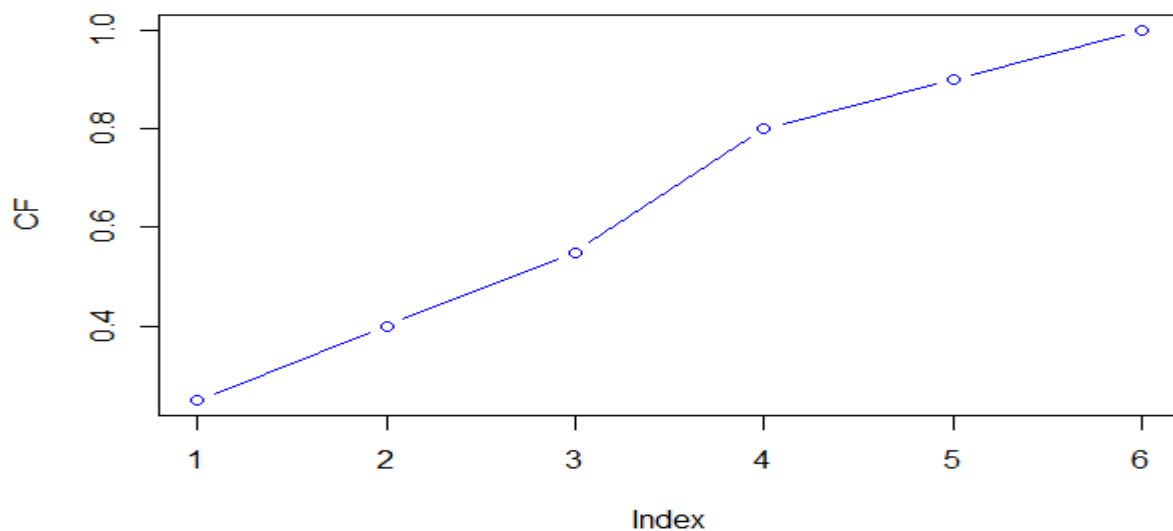
```
> cumsum(table(cut.nf))/length(cut.nf)
```

```
[30,40) [40,50) [50,60) [60,70) [70,80) [80,90)
```

```
 0.25  0.40  0.55  0.80  0.90  1.00
```

```
CF=cbind(cumsum(table(cut.nf))/length(cut.nf))
```

```
plot(CF,col="blue", type="b", lty=1, lwd=1, pch=1, cex=1)
```



In simple bar chart, we make bars of equal width but variable length, i.e. the magnitude of a quantity is represented by the height or length of the bars

```
> barplot(nf, main = "Fruits/Plant", xlab = "Trees",ylab = "No. of Fruits")
```



Let us consider this hypothetical data collected from an experimental filed

```
> nil=matrix(nrow=4,ncol=3,data=c(5,4,2,6,5,4,7,5,3,5,2,4),byrow = T) # Creating matrix
```

```
> nil
```

```
 [,1] [,2] [,3]
```

```
[1,]  5   4   2
```

```
[2,]  6   5   4
```

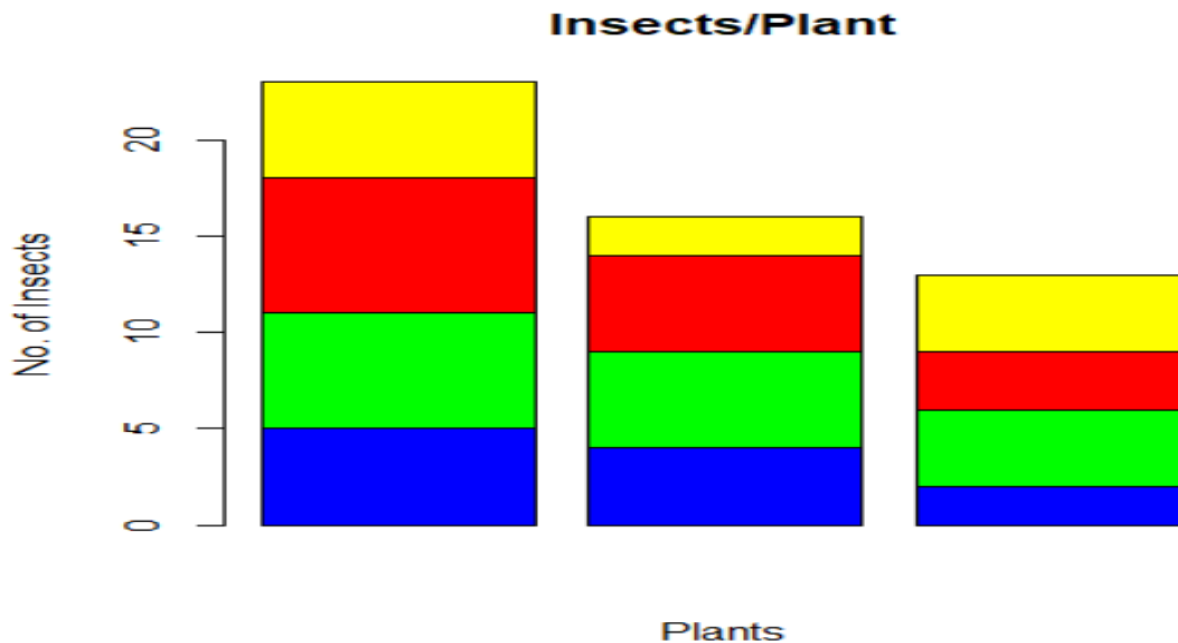
```
[3,]  7   5   3
```

```
[4,]  5   2   4
```

No. of insects/leaf	Plant 1	Plant 2	Plant 3
Day 1	5	4	2
Day 2	6	5	4
Day 3	7	5	3
Day 4	5	2	4

```
>barplot(nil,main="Insects/Plant",
```

```
xlab="Plants",ylab="No.ofInsects",col=c("blue","green","red","yellow"))
```



Pie Chart:

It is a circle partitioned into segments where each segment represents a category

The size of each segment depends upon the relative frequency determined by an angle (relative frequency X 360 degree)

Ex: Participants in this training program : 150

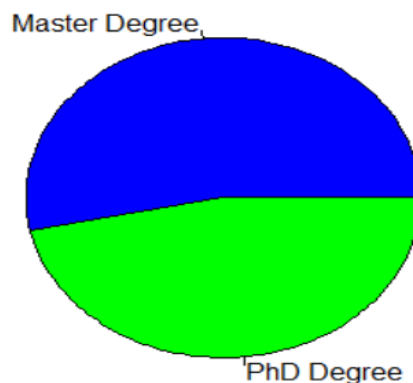
Suppose: 80 hold Master's degree

70 are PhD's

```
> participants<-c(80,70)
```

```
>pie(participants,labels = c("Master Degree", "PhD Degree"),main="Qualification of the  
Participants", col=c("blue","green"))
```

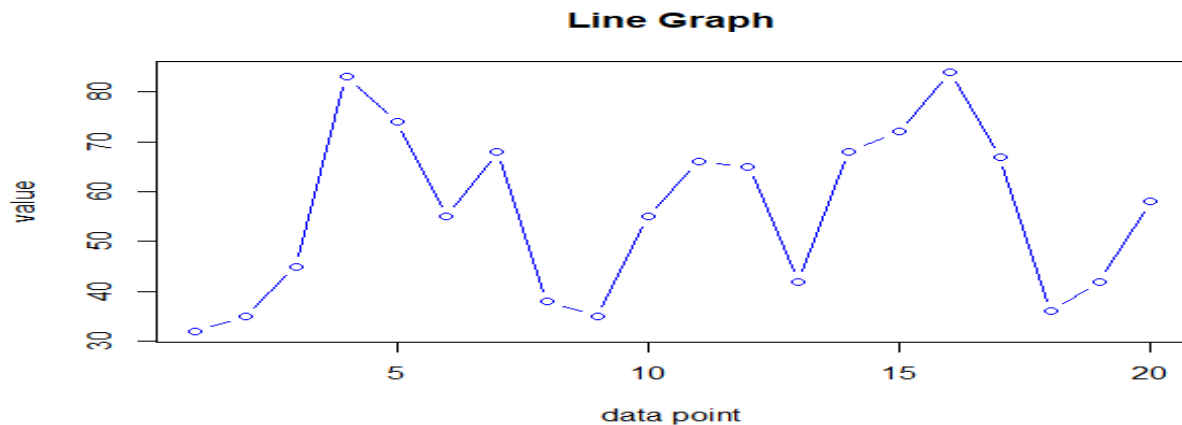
Qualification of the Participants



Line Graphs are used to display quantitative values over a continuous interval or time period

A Line Graph is most frequently used to show trends and analyse how the data has changed over time

```
plot(nf, main="Line Graph", xlab="data point", ylab="value", col="blue", type="b", lty=1,
lwd=1, pch=1, cex=1)
```



Measures of central tendency

- ❖ Usually data set has numerous variable and each variable have many observations
- ❖ Difficult to handle each observation and dig out the information from each observation
- ❖ Our interest is in summary of information hidden inside the data
- ❖ We generally conceive data as some averages
- ❖ The average performance of students is 80%
- ❖ In statistics this is referred to as “average” or the central tendency of the data
- ❖ **Arithmetic Mean**
- ❖ **Harmonic Mean**
- ❖ **Geometric Mean**
- ❖ **Median**
- ❖ **Mode**
- ❖ **Quantiles**

Arithmetic Mean (AM):

Ungrouped Data

The AM for a set of observations $x_1, x_2, x_3, \dots, x_n$ are defined as:

$$\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i$$

mean(x)

We assume x to be nf (number of fruits/plant)

> nf

[1] 32 35 45 83 74 55 68 38 35 55 66 65 42 68 72 84 67 36 42 58

> mean(nf)

[1] 56

> nf=c(NA,32,35,45,83,74,55,68,38,35,55,66,65,42,68,72,84,67,36,42,58)

> mean(nf,na.rm = TRUE)

[1] 56

Arithmetic Mean (AM):

Grouped Data

The AM for a set of observations $x_1, x_2, x_3, \dots, x_n$ are defined as:

$$\bar{x} = \frac{1}{n} \sum_{i=1}^K n_i m_i = \sum_{i=1}^K f_i m_i$$

K = total no of class

m_i = mid point of iclass

n =total frequency

$f_i=n_i/n$ =frequency of each class

Another way of representation is as weighted mean:

$$\bar{x} = \frac{\sum_{i=1}^K w_i m_i}{\sum_{i=1}^K w_i}$$

> f=as.numeric(table(cut.nf))

> f

[1] 5 3 3 5 2 2

>m=c(35,45,55,65,75,85)

>weighted.mean(m,f)

[1] 56

Median:

Median of a given distribution is the value of the variable which divides the distribution in to

two equal parts

It is the value such that number of observations preceding as well as succeeding from the median is equal or which exceeds and exceeded by the same number of observations

Median is thus a Positional Average only

First of all, the given observations of the distribution are arranged in ascending/descending order in case of ungrouped data. Median is calculated as follows:

If number of observations is odd

Median = Value of $\left(\frac{n+1}{2}\right)^{th}$ item

If the number of observations is even

Median = Average of $\left(\frac{n}{2}\right)^{th}$ and $\left(\frac{n+1}{2}\right)^{th}$ items

In case of the grouped data (continuous frequency distribution), a separate column of cumulative frequencies is also made

Find the number $n/2$

Cumulative frequency in which this number $n/2$ falls

The corresponding class interval is called the Median Class

After locating the Median Class, following formula is used for calculation of median.

$$\text{Median} = l + \frac{h}{f} \left(\frac{n}{2} - c \right)$$

Where,

l = Lower class limit of the Median Class

f = Frequency of the Median Class

$n = \Sigma f$ = Sum of the frequencies of various class intervals

c = Cumulative frequency of the class preceding the Median Class

h = Class interval size of the Median Class

> median(nf)

[1] 56.5

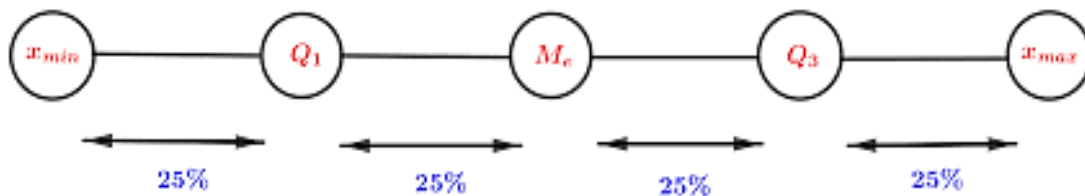
Partition Values:

The frequency distribution is partitioned to have an idea about the concentration of values over the entire frequency distribution

The important ones are :

Median, Quartile, Decile, Percentile

```
> quantile(nf)
 0%  25%  50%  75% 100%
32.0 41.0 56.5 68.0 84.0
> quantile(nf,na.rm=TRUE)
```



Decile:

It partitions the dataset into 10 equal parts (D_1, D_2, \dots, D_{10})

```
> prob=seq(0,1,0.1)
> prob
[1] 0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0
> quantile(nf,prob)
 0% 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%
32.0 35.0 37.6 42.0 51.0 56.5 65.4 67.3 68.8 74.9 84.0
>prob=seq(0,1,0.01)
> prob
[1] 0.00 0.01 0.02 0.03 0.04 0.05 0.06 0.07 0.08 0.09 0.10 0.11 0.12 0.13 0.14
```

Mode is the value which occurs most frequently in the given set of observations i.e. it is the value of the variable which is predominant in the given set of observations

If the data having only one mode the distribution is said to be uni-model and is said to be bi-model, if data have two modes

For ungrouped data, mode is calculated by inspecting the given data

The value which occurs maximum number of times in the distribution is called the Mode of the given distribution

For grouped data, locate the Modal Class/Group

The class/group which has the maximum frequency is called the Modal Class/Group

After locating the Modal Class/Group, the following formula is applied for calculation of Mode of the given frequency distribution

$$\text{Mode} = l + \frac{f_m - f_1}{(f_m - f_1) + (f_m - f_2)} \times h$$

Where,

l is the lower class limit of the modal group

f_m is the frequency of the modal group

f_1 is the frequency of the class interval preceding the modal group

f_2 is the frequency of the class interval preceding the modal group

h is the class interval of the modal group

Geometric mean of a set of n observations is the n^{th} root of the multiplication of all the n observations

Hence the geometric mean denoted by G ; of n observations $x_i, i = 1, 2, \dots, n$ is given by the formula

$$G = (x_1 \cdot x_2 \cdot \dots \cdot x_n)^{1/n}$$

$$G = \text{Antilog} \left[\frac{1}{n} \sum_{i=1}^n \log x_i \right]$$

In case of grouped frequency distribution, geometric mean is given by the formula

$$G = \text{Antilog} \left[\frac{1}{n} \sum_{i=1}^n f_i \log x_i \right] \text{ where } n = \sum_{i=1}^n f_i$$

> GM=prod(nf)^(1/length(nf))

> GM

[1] 53.49774

Harmonic mean is defined as the quotient of “number of the given values” and “sum of the reciprocals of the given values”

Harmonic mean in mathematical terms is defined as follows:

For ungrouped data:
$$\text{HM} = \frac{n}{\sum(1/n)}$$

For grouped data:
$$\text{HM} = \frac{\sum f}{\sum \left(\frac{f}{x} \right)}$$

> HM=1/mean(1/nf)

> HM

[1] 50.99942

Measures of central tendency give us single figure which represent the entire distribution or set of observations or around which the observations of the set of data concentrated

But they are inadequate to give us the complete idea of the distribution because they do not tell us the extent to which the observations of the distribution vary from the central value

X: 10,15,20,35,30

mean= 20

Y: 2,8,10,20,60

mean= 20

For the study of dispersion, there are some measures which show whether the dispersion is small or large

There are two types of measure of dispersion:

(a) Absolute Measure of Dispersion

(b) Relative Measure of Dispersion

Absolute Measures of Dispersion:

These measures give us an idea about the amount of dispersion in a set of observations

1. Range
2. Quartile Deviation or Semi Inter Quartile Range
3. Mean Deviation
4. Variance and Standard deviation

Relative Measure of Dispersion:

These measures are calculated for the comparison of dispersion in two or more than two sets of observations

These measures are free of the units in which the original data is measured

1. Coefficient of Range
2. Coefficient of Quartile Deviation
3. Coefficient of Mean Deviation
4. Coefficient of Variation

Range is defined as the difference between the maximum and the minimum values of the given observations

If x_m denotes the maximum value and x_0 denotes the minimum value, range is defined as:

$$Range = x_m - x_0$$

$$Coefficient\ of\ Range = \frac{x_m - x_0}{x_m + x_0}$$

> range=max(nf)-min(nf)

```
> range
```

```
[1] 52
```

```
> range(nf)
```

```
[1] 32 84
```

Quartile Deviation/Semi Inter Quartile Range

It is based on the lower quartile Q_1 and the upper quartile Q_3

$$\text{Quartile Deviation} = \frac{Q_3 - Q_1}{2}$$

$$\text{Coefficient of Quartile Deviation} = \frac{Q_3 - Q_1}{Q_3 + Q_1}$$

```
> IQR(nf)
```

```
[1] 27
```

```
> qd=IQR(nf)/2
```

```
> qd
```

```
[1] 13.5
```

The mean deviation is defined as the arithmetic mean of the absolute deviations of all the values taken from some suitable average which may be the arithmetic mean, the median or the mode

The mean deviation of a set of sample data in which the suitable average (AM) is \bar{X} , is given by the relation:

$$\text{Mean Deviation} = \frac{\sum |X - \bar{X}|}{n}$$

For frequency distribution

$$\text{Mean Deviation} = \frac{\sum f |X - \bar{X}|}{\sum f}$$

Mean deviation is a better measure of dispersion than Range and Quartile Deviation

```
> MD=mean(abs(nf-mean(nf)))
```

```
> MD
```

```
[1] 14.5
```

The standard deviation is the absolute measure of dispersion

Its relative measure is called standard coefficient of dispersion or coefficient of standard deviation

$$\text{Coefficient of Standard Deviation} = \frac{\sigma}{\bar{X}}$$

The coefficient of variation (CV) is given by the formula

$$\text{Coefficient of Variation} = \frac{\sigma}{\bar{X}} \times 100$$

Coefficient of variation is a pure number and the unit of observations cannot be mentioned with its value

References:

Goon, A.M., Gupta, M.K. and Dasgupta, R. 1986. *Outline of Statistics*. Vol. I. World Press.

Goon, A.M., Gupta, M.K. and Dasgupta, R. 2008. *Fundamentals of Statistics*. Vol. I.

Atlantic Publishers

<https://nptel.ac.in/courses/111/104/111104120/>

Planning and Designing of Experiments using R

Seema Jaggi*, Anindita Datta¹ and Mohammed Harun¹

**Krishi Anusandhan Bhavan II, New Delhi-110012*

ICAR-Indian Agricultural Statistics Research Institute, New Delhi - 110 012

seema.jaggi@icar.gov.in, anindita.datta@icar.gov.in, mohd.harun@icar.gov.in

An experiment is usually associated with a scientific method for testing certain phenomena. An experiment facilitates the study of such phenomena under controlled conditions and thus creating controlled conditions is an essential component. Scientists in the biological fields who are involved in research constantly face problems associated with planning, designing, and conducting experiments. Basic familiarity and understanding of statistical methods that deal with issues of concern would be helpful in many ways. Researchers who collect data and then look for a statistical technique that would provide valid results will find that there may not be solutions to the problem and that the problem could have been avoided first by a properly designed experiment. Obviously, it is important to keep in mind that we cannot draw valid conclusions from poorly planned experiments. Second, the time and cost involved in many experiments are enormous and a poorly designed experiment increases such costs in time and resources. For example, an agronomist who carries out fertilizer experiments knows the time limitation of the experiment. He knows that when seeds are to be planted and harvested. The experimenter plot must include all components of a complete design. Otherwise, what is omitted from the experiment will have to be carried out in subsequent trials in the next cropping season or next year. The additional time and expenditure could be minimized by a properly planned experiment that will produce valid results as efficiently as possible. Good experimental designs are products of the technical knowledge of one's field, an understanding of statistical techniques and skill in designing experiments.

Any research endeavor may entail the phases of Conception, Design, Data collection, Analysis and Dissemination. Statistical methodologies can be used to conduct better scientific experiments if they are incorporated into entire scientific process, i.e., From inception of the problem to experimental design, data analysis and interpretation. When planning experiments, we must keep in mind that large uncontrolled variations are common occurrences. Experiments are generally undertaken by researchers to compare effects of several conditions on some phenomena or in discovering an unknown effect of particular process.

An experiment facilitates the study of such phenomena under controlled conditions. Therefore, the creation of controlled condition is the most essential characteristic of experimentation. How we formulate our questions and hypotheses are critical to the experimental procedure that will follow. For example, a crop scientist who plants the same variety of a crop in a field may find variations in yield that are due to periodic variations across a field or to some other factors that the experimenter has no control over. The methodologies used in designing experiments will separate with confidence and accuracy a varietal difference of crops from the uncontrolled variations.

The different concepts in planning of experiments can be well explained through chapati tasting experiment.

Consider an experiment to detect the taste difference in chapati made of wheat flour of c306 and pv 18 varieties. The null hypothesis we can assume here is that there is no taste difference in chapatis made of c306 or pv18 wheat flours. After the null hypothesis is set, we have to fix the level of significance at which we can operate. The pv18 is a much higher yielding variety than c306. Hence a false rejection may not help the country to grow more pv18 and the wheat production may decrease while a false acceptance may give more production of pv18 wheat and the consumption may be less or practically nil. Thus the false acceptance or false rejection are of practically equal consequence and we agree to choose the level of significance at $\alpha = 0.05$. Now to execute the experiment, a subject is to be found with extrasensory powers who can detect the taste differences. The colours of c306 and pv18 are different and anyone, even without tasting the chapatis, can distinguish the chapatis of either kind by a mere glance. Thus the taster of the chapatis has to be blindfolded before the chapatis are given for tasting. Afterwards, the method is to be decided in which the experiment will be conducted. The experiment can be conducted in many ways and of them three methods are discussed here:

- Give the taster equal number of chapatis of either kind informing the taster about it.
- Give the taster pairs of chapatis of each kind informing the taster about it.
- Give the taster chapatis of either kind without providing him with any information. Let us use 6 chapatis in each of these methods.

Under first method of experimentation, if the null hypothesis is true, then the experimenter cannot distinguish the two kinds of chapatis and he will randomly select 3 chapatis out of 6 chapatis given to him, as made of pv18 wheat. In that case, all correct guesses are made if selection exactly coincides with

the exactly used wheat variety and the probability for such an occurrence is:

$$\frac{1}{\binom{6}{3}} = \frac{1}{20} = 0.05$$

Under second method, the pv18 wheat variety chapaties are selected from each pair given if the null hypothesis is true. Furthermore, independent choices are made of pv18 variety chapaties from each pair. Thus the probability of making all correct guesses is

$$1/(2)^3 = 1/8 = 0.125.$$

In third method the experimenter has to make the choice for each chapati and the situation is analogous at calling heads or tails in a coin tossing experiment. The probability of making all correct guesses would then be:

$$1/2^6 = 1/64 = .016.$$

If the experimenter makes all correct guesses in third method as its probability is smaller than the selected $\alpha = 0.05$, we can reject the null hypothesis and conclude that the two wheat varieties give different tastes at chapaties. In other methods the probability of making all correct guesses does not exceed $\alpha = 0.05$ and hence with either method, we cannot reject the null hypothesis even if all correct guesses are made.

However, if 8 chapaties are used by first method and if the taster guesses all of them, we can reject the null hypothesis, at 0.05 level of significance, as the probability of making all correct guesses would then be $\frac{1}{\binom{8}{3}} = \frac{1}{56}$ which is smaller than 0.05. 8 chapaties will not enable us to reject the null hypothesis

even if all correct guesses are made by second method as the probability of making all correct guesses is

$\left(\frac{1}{4}\right)^4 = \frac{1}{16} = 0.06$ it is easy to see that if 10 chapaties are given by second method and if all correct guesses are made, then we can reject the null hypothesis at 0.05 level of significance. Not to unduly influence the taster in making guesses, we should also present the chapaties in a random order rather than systematically presenting them for tasting.

The above discussed chapati tasting experiment brings home the following salient features of

experimentation:

- All the extraneous variations in the data should be eliminated or controlled excepting the variations due to the treatments under study. One should not artificially provide circumstances for one treatment to show better results than others.
- For a given size of the experiment, though the experiment can be done in many ways, even the best results may not turn out to be significant with some designs, while some other design can detect the treatment differences. Thus there is an imperative need to choose the right type of design, before the commencement of the experiment, lest the results may be useless.
- If for some specific reasons related to the nature of the experiment, a particular method has to be used in experimentation, then adequate number of replications of each treatment have to be provided in order to get valid inferences.
- The treatments have to be randomly allocated to the experimental units.

The terminologies often used in planning and designing of experiments are listed below.

Treatment

Treatment refers to controllable quantitative or qualitative factors imposed at a certain level by the experimenter. For an agronomist several fertilizer concentrations applied to a particular crop or a variety of crop is a treatment. Similarly, an animal scientist looks upon several concentrations of a drug given to animal species as a treatment. In agribusiness we may look upon impact of advertising strategy on sales as a treatment. To an agricultural engineer, different levels of irrigation may constitute a treatment.

Experimental Unit

An experimental unit is an entity that receives a treatment e.g., for an agronomist or horticulturist it may be a plot of a land or batch of seed, for an animal scientist it may be a group of pigs or sheep, for a scientist engaged in forestry research it may be different tree species occurring in an area, and for an agricultural engineer it may be a manufactured item. Thus, an experimental unit may be looked upon as a small subdivision of the experimental material, which receives the treatment.

Experimental Error

Differences in yields arising out of experimental units treated alike are called Experimental Error.

Controllable conditions in an experiment or experimental variable are termed as a factor. For example, a fertilizer, a new feed ration, and a fungicide are all considered as factors. Factors may be qualitative or quantitative and may take a finite number of values or type. Quantitative factors are those described by numerical values on some scale. The rates of application of fertilizer, the quantity of seed sown are examples of quantitative factors. Qualitative factors are those factors that can be distinguished from each other, but not on numerical scale e.g., type of protein in a diet, sex of an animal, genetic make up of plant etc. While choosing factors for any experiment researcher should ask the following questions, like What treatments in the experiment should be related directly to the objectives of the study? Does the experimental technique adopted require the use of additional factors? Can the experimental unit be divided naturally into groups such that the main treatment effects are different for the different groups? What additional factors should one include in the experiment to interact with the main factors and shed light on the factors of direct interest? How desirable is it to deliberately choose experimental units of different types?

Basic Principles of Design of Experiments

Given a set of treatments which can provide information regarding the objective of an experiment, a design for the experiment, defines the size and number of experimental units, the manner in which the treatments are allotted to the units and also appropriate type and grouping of the experimental units. These requirements of a design ensure validity, interpretability and accuracy of the results obtainable from an analysis of the observations.

These purposes are served by the principles of:

- Randomization
- Replication
- Local (Error) control

Randomization

After the treatments and the experimental units are decided the treatments are allotted to the experimental units at random to avoid any type of personal or subjective bias, which may be conscious or unconscious. This ensures validity of the results. It helps to have an objective comparison among the treatments. It also

ensures independence of the observations, which is necessary for drawing valid inference from the observations by applying appropriate statistical techniques.

Depending on the nature of the experiment and the experimental units, there are various experimental designs and each design has its own way of randomization. Various speakers while discussing specific designs in the lectures to follow shall discuss the procedure of random allocation separately.

Replication

If a treatment is allotted to r experimental units in an experiment, it is said to be replicated r times. If in a design each of the treatments is replicated r times, the design is said to have r replications. Replication is necessary to

- Provide an estimate of the error variance which is a function of the differences among observations from experimental units under identical treatments.
- Increase the accuracy of estimates of the treatment effects.

Though, more the number of replications the better it is, so far as precision of estimates is concerned, it cannot be increased infinitely as it increases the cost of experimentation. Moreover, due to limited availability of experimental resources too many replications cannot be taken.

The number of replications is, therefore, decided keeping in view the permissible expenditure and the required degree of precision. Sensitivity of statistical methods for drawing inference also depends on the number of replications. Sometimes this criterion is used to decide the number of replications in specific experiments.

Error variance provides a measure of precision of an experiment, the less the error variance the more precision. Once a measure of error variance is available for a set of experimental units, the number of replications needed for a desired level of sensitivity can be obtained as below.

Given a set of treatments an experimenter may not be interested to know if two treatment differ in their effects by less than a certain quantity, say, d . In other words, he wants an experiment that should be able to differentiate two treatments when they differ by d or more.

The significance of the difference between two treatments is tested by t-test where

$$t = \frac{\bar{y}_i - \bar{y}_j}{\sqrt{2s^2 / r}},$$

Here, \bar{y}_i , and \bar{y}_j are the arithmetic means of two treatment effects each based on r replications, s^2 is measure of error variation.

Given a difference d , between two treatment effects such that any difference greater than d should be brought out as significant by using a design with r replications, the following equation provides a solution of r .

$$t = \frac{|d|}{\sqrt{2s^2 / r}},$$

$$r = \frac{t_0^2}{d^2} \times 2s^2 \quad \dots(1)$$

where t_0 is the critical value of the t-distribution at the desired level of significance, that is, the value of t at 5 or 1 per cent level of significance read from the t-table. If s^2 is known or based on a very large number of observations, made available from some pilot pre-experiment investigation, then t is taken as the normal variate. If s^2 is estimated with n degree of freedom (d.f.) then t_0 corresponds to n d.f.

When the number of replication is r or more as obtained above, then all differences greater than d are expected to be brought out as significant by an experiment when it is conducted on a set of experimental units which has variability of the order of s^2 . For example, in an experiment on wheat crop conducted in a seed farm in Bhopal, to study the effect of application of nitrogen and phosphorous on yield a randomized block design with three replications was adopted. There were 11 treatments two of which were (i) 60 Kg/ha of nitrogen (ii) 120 Kg/ha of nitrogen. The average yield figures for these two application of the fertilizer were 1438 and 1592 Kg/ha respectively and it is required that differences of the order of 150 Kg/ha should be brought out significant. The error mean square (s^2) was 12134.88. Assuming that the experimental error will be of the same order in future experiments and t_0 is of the order of 2.00, which is likely as the error d.f. is likely to be more than 30 as there are 11 treatments; Substituting in (1), we get:

$$r = \frac{2t_0^2 s^2}{d^2} = \frac{2 \times 2^2 \times 2134.88}{150^2} = 4 \text{ (approx.)}$$

Thus, an experiment with 4 replications is likely to bring out differences of the order of 150 Kg/ha as significant.

Another criterion for determining r is to take a number of replications which ensures at least 10 d.f. for the estimate of error variance in the analysis of variance of the design concerned since the sensitivity of the experiment will be very much low as the F test (which is used to draw inference in such experiments) is very much unstable below 10 d.f.

Local Control

The consideration in regard to the choice of number of replications ensure reduction of standard error of the estimates of the treatment effect because the standard error of the estimate of a treatment effect is $\sqrt{s^2 / r}$, but it cannot reduce the error variance itself. It is, however, possible to devise methods for reducing the error variance. Such measures are called *error control* or local control. One such measure is to make the experimental units homogenous. Another method is to form the units into several homogenous groups, usually called blocks, allowing variation between the groups.

A considerable amount of research work has been done to divide the treatments into suitable groups of experimental units so that the treatment effect can be estimated more precisely. Extensive use of combinatorial mathematics has been made for formation of such group treatments. This grouping of experiment units into different groups has led to the development of various designs useful to the experimenter. We now briefly describe the various term used in designing of an experiment

Blocking

It refers to methodologies that form the units into homogeneous or pre-experimental subject-similarity groups. It is a method to reduce the effect of variation in the experimental material on the Error of Treatment of Comparisons. For example, animal scientist may decide to group animals on age, sex, breed or some other factors that he may believe has an influence on characteristic being measured. Effective blocking removes considerable measure of variation from the experimental error. The selection of source of variability to be used as basis of blocking, block size, block shape and orientation are crucial for

blocking. The blocking factor is introduced in the experiment to increase the power of design to detect treatment effects.

The importance of good designing is inseparable from good research (results). The following examples point out the necessity for a good design that will yield good research. First, a nutrition specialist in developing country is interested in determining whether mother's milk is better than powdered milk for children under age one. The nutritionist has compared the growth of children in village A, who are all on mother's milk against the children in village B, who use powdered milk. Obviously, such a comparison ignores the health of the mothers, the sanitary-conditions of the villages, and other factors that may have contributed to the differences observed without any connection to the advantages of mother's milk or the powdered milk on the children. A proper design would require that both mother's milk and the powdered milk be alternatively used in both villages, or some other methodology to make certain that the differences observed are attributable to the type of milk consumed and not to some uncontrollable factor. Second, a crop scientist who is comparing 2 varieties of maize, for instance, would not assign one variety to a location where such factors as sun, shade, unidirectional fertility gradient, and uneven distribution of water would either favor or handicap it over the other. If such a design were to be adopted, the researcher would have difficulty in determining whether the apparent difference in yield was due to variety differences or resulted from such factors as sun, shade, soil fertility of the field, or the distribution of water. These two examples illustrate the type of poorly designed experiments that are to be avoided.

Analysis of Variance

Analysis of Variance (ANOVA) is a technique of partitioning the overall variation in the responses into different assignable sources of variation, some of which are specifiable and others unknown. Total variance in the sample data is partitioned and is expressed as the sum of its non-negative components is a measure of the variation due to some specific independent source or factor or cause. ANOVA consists in estimation of the amount of variation due to each of the independent factors (causes) separately and then comparing these estimates due to ascribable factors (causes) with the estimate due to chance factor the latter being known as experimental error or simply the error.

Total variation present in a set of observable quantities may, under certain circumstances, be partitioned into a number of components associated with the nature of classification of the data. The systematic procedure for achieving this is called *Analysis of Variance*. The initial techniques of the analysis of

variance were developed by the [statistician](#) and [geneticist R. A. Fisher](#) in the 1920s and 1930s, and is sometimes known as Fisher's analysis of variance, due to the use of Fisher's [F-distribution](#) as part of the test of [statistical significance](#).

Thus, ANOVA is a statistical technique that can be used to evaluate whether there are differences between the average value, or mean, across several population groups. With this model, the *response variable is continuous* in nature, whereas the *predictor variables are categorical*. For example, in a clinical trial of hypertensive patients, ANOVA methods could be used to compare the effectiveness of three different drugs in lowering blood pressure. Alternatively, ANOVA could be used to determine whether infant birth weight is significantly different among mothers who smoked during pregnancy relative to those who did not. In a particular case, where two population means are being compared, ANOVA is equivalent to the independent two-sample *t*-test.

The fixed-effects model of ANOVA applies to situations in which the experimenter applies several treatments to the subjects of the experiment to see if the [response variable](#) values change. This allows the experimenter to estimate the ranges of response variable values that the treatment would generate in the population as a whole. In it factors are fixed and are attributable to a finite set of levels of factor eg. Sex, year, variety, fertilizer etc.

Consider for example a clinical trial where three drugs are administered on a group of men and women some of whom are married and some are unmarried. The three classifications of sex, drug and marital status that identify the source of each datum are known as factors. The individual classification of each factor is known as levels of the factors. Thus, in this example there are 3 levels of factor drug, 2 levels of factor sex and 2 levels of marital status. Here all the effects are fixed. Random effects models are used when the treatments are not fixed. This occurs when the various treatments (also known as factor levels) are sampled from a larger population. When factors are random, these are generally attributable to infinite set of levels of a factor of which a random sample are deemed to occur eg. research stations, clinics in Delhi, sire, etc. Suppose new inject-able insulin is to be tested using 15 different clinics of Delhi state. It is reasonable to assume that these clinics are random sample from a population of clinics from Delhi. It describe the situations where both fixed and random effects are present.

In any ANOVA model, general mean is always taken as fixed effect and error is always taken as random effect. Thus class of model can be classified on the basis of factors, other than these two factors. ANOVA can be viewed as a generalization of t -tests: a comparison of differences of means across more than two groups.

The ANOVA is valid under certain assumptions. These assumptions are:

- Samples have been drawn from the populations that are normally distributed.
- Observations are independent and are distributed normally with mean zero and variance σ^2 .
- Effects are additive in nature.

The ANOVA is performed as one-way, two-way, three-way, etc. ANOVA when the number of factors is one, two or three respectively. In general if the number of factors is more, it is termed as multi-way ANOVA.

Completely Randomized Design

Designs are usually characterized by the nature of grouping of experimental units and the procedure of random allocation of treatments to the experimental units. In a completely randomized design the units are taken in a single group. As far as possible the units forming the group are homogeneous. This is a design in which only randomization and replication are used. There is no use of local control here.

Let there be v treatments in an experiment and n homogeneous experimental units. Let the i^{th} treatment be replicated r_i times ($i = 1, 2, \dots, v$) such that $\sum_{i=1}^v r_i = n$. The treatments are allotted at random to the units.

Normally the number of replications for different treatments should be equal as it ensures equal precision of estimates of the treatment effects. The actual number of replications is, however, determined by the availability of experimental resources and the requirement of precision and sensitivity of comparisons. If the experimental material for some treatments is available in limited quantities, the numbers of their replication are reduced. If the estimates of certain treatment effects are required with more precision, the numbers of their replication are increased.

Randomization

There are several methods of random allocation of treatments to the experimental units. The v treatments are first numbered in any order from 1 to v . The n experimental units are also numbered suitably. One of the methods uses the random number tables. Any page of a random number table is taken. If v is a one-digit number, then the table is consulted digit by digit. If v is a two-digit number, then two-digit random

numbers are consulted. All numbers greater than v including zero are ignored.

Let the first number chosen be n_1 ; then the treatment numbered n_1 is allotted to the first unit. If the second number is n_2 which may or may not be equal to n_1 then the treatment numbered n_2 is allotted to the second unit. This procedure is continued. When the i^{th} treatment number has occurred r_i times, ($i=1,2,\dots,v$) this treatment is ignored subsequently. This process terminates when all the units are exhausted.

One drawback of the above procedure is that sometimes a very large number of random numbers may have to be ignored because they are greater than v . It may even happen that the random number table is exhausted before the allocation is complete. To avoid this difficulty the following procedure is adopted. We have described the procedure by taking v to be a two-digit number.

Let P be the highest two-digit number divisible by v . Then all numbers greater than P and zero are ignored. If a selected random number is less than v , then it is used as such. If it is greater than or equal to v , then it is divided by v and the remainder is taken to be the random number. When a number is completely divisible by v , then the random number is v . If v is an n -digit number, then P is taken to be the highest n -digit number divisible by v . The rest of the procedure is the same as above.

Analysis

This design provides a one-way classified data according to levels of a single factor. For its analysis the following model is taken:

$$y_{ij} = \mu + t_i + e_{ij}, \quad i = 1, \Lambda, v; j = 1, \Lambda, r_i,$$

where y_{ij} is the random variable corresponding to the observation y_{ij} obtained from the j^{th} replicate of the i^{th} treatment, μ is the general mean, t_i is the fixed effect of the i^{th} treatment and e_{ij} is the error component which is a random variable assumed to be normally and independently distributed with zero means and a constant variance σ^2 .

Let $\sum_j y_{ij} = T_i$ ($i=1,2,\dots,v$) be the total of observations from i^{th} treatment. Let further $\sum_i T_i = G$.

Correction factor (C.F.) = G^2/n .

$$\text{Sum of squares due to treatments} = \sum_{i=1}^v \frac{T_i^2}{r_i} - C.F.$$

$$\text{Total sum of squares} = \sum_{i=1}^v \sum_{j=1}^{r_i} y_{ij}^2 - C.F.$$

ANALYSIS OF VARIANCE

Sources of variation	Degrees of freedom (D.F.)	Sum of squares (S.S.)	Mean squares (M.S.)	F
Treatments	$v - 1$	SST $= \sum_{i=1}^v \frac{T_i^2}{r_i} - C.F.$	$MST = SST / (v - 1)$	MST/MSE
Error	$n - v$	$SSE = \text{by subtraction}$	$MSE = SSE / (n - v)$	
Total	$n - 1$	$\sum_{ij} y_{ij}^2 - C.F.$		

The hypothesis that the treatments have equal effects is tested by F-test where F is the ratio MST / MSE with $(v - 1)$ and $(n - v)$ degrees of freedom.

3. Randomized Complete Block Design

It has been seen that when the experimental units are homogeneous then a CRD should be adopted. In any experiment, however, besides treatments the experimental material is a major source of variability in the data. When experiments require a large number of experimental units, the experimental units may not be homogeneous, and in such situations CRD can not be recommended. When the experimental units are heterogeneous, a part of the variability can be accounted for by grouping the experimental units in such a way that experimental units within each group are as homogeneous as possible. The treatments are then allotted randomly to the experimental units within each group (or blocks). The principle of first forming homogeneous groups of the experimental units and then allotting at random each treatment once in each group is known as local control. This results in an increase in precision of estimates of the treatment contrasts, due to the fact that error variance that is a function of comparisons within blocks, is smaller because of homogeneous blocks. This type of allocation makes it possible to eliminate from error variance a portion of variation attributable to block differences. If, however, variation between the blocks is not significantly large, this type of grouping of the units does not lead to any advantage; rather some degrees of freedom of the error variance is lost without any consequent decrease in the error variance. In such situations it is not desirable to adopt randomized complete block designs in preference to completely

randomized designs.

If the number of experimental units within each group is same as the number of treatments and if every treatment appears precisely once in each group then such an arrangement is called a ***randomized complete block design***.

Suppose the experimenter wants to study v treatments. Each of the treatments is replicated r times (the number of blocks) in the design. The total number of experimental units is, therefore, vr . These units are arranged into r groups of size v each. The error control measure in this design consists of making the units in each of these groups homogeneous.

The number of blocks in the design is the same as the number of replications. The v treatments are allotted at random to the v plots in each block. This type of homogeneous grouping of the experimental units and the random allocation of the treatments separately in each block are the two main characteristic features of randomized block designs. The availability of resources and considerations of cost and precision determine actual number of replications in the design.

Analysis

The data collected from experiments with randomized block designs form a two-way classification, that is, classified according to the levels of two factors, viz., blocks and treatments. There are vr cells in the two-way table with one observation in each cell. The data are orthogonal and therefore the design is called an *orthogonal design*. We take the following model:

$$y_{ij} = \mu + t_i + b_j + e_{ij}, \quad \begin{pmatrix} i = 1, 2, \dots, v; \\ j = 1, 2, \dots, r \end{pmatrix},$$

where y_{ij} denotes the observation from i^{th} treatment in j^{th} block. The fixed effects μ, t_i, b_j denote respectively the general mean, effect of the i^{th} treatment and effect of the j^{th} block. The random variable e_{ij} is the error component associated with y_{ij} . These are assumed to be normally and independently distributed with zero means and a constant variance σ^2 .

Following the method of analysis of variance for finding sums of squares due to blocks, treatments and error for the two-way classification, the different sums of squares are obtained as follows: Let

$\sum_j y_{ij} = T_i$ ($i = 1, 2, \dots, v$) = total of observations from i^{th} treatment and $\sum_j y_{ij} = B_j$ ($j = 1, 2, \dots, r$) = total of observations from j^{th} block. These are the marginal totals of the two-way data table. Let further, $\sum_i T_i = \sum_j B_j = G$.

Correction factor ($C.F.$) = G^2/rv , Sum of squares due to treatments = $\sum_i \frac{T_i^2}{r} - C.F.$,

Sum of squares due to blocks = $\sum_j \frac{B_j^2}{v} - C.F.$, Total sum of squares = $\sum_{ij} y_{ij}^2 - C.F.$

ANALYSIS OF VARIANCE				
Sources of variation	Degrees of freedom (D.F.)	Sum of squares (S.S.)	Mean squares (M.S.)	F
Blocks	$r - 1$	$SSB = \sum_j \frac{B_j^2}{v} - C.F.$	$MSB = SSB / (r - 1)$	MSB/MSE
Treatments	$v - 1$	$SST = \sum_i \frac{T_i^2}{r} - C.F.$	$MST = SST / (v - 1)$	MST/MSE
Error	$(r - 1)(v - 1)$	$SSE = \text{by subtraction}$	$MSE = SSE / (v - 1)(r - 1)$	
Total	$vr - 1$	$\sum_{ij} y_{ij}^2 - C.F.$		

The hypothesis that the treatments have equal effects is tested by F-test, where F is the ratio MST / MSE with $(v - 1)$ and $(v - 1)(r - 1)$ degrees of freedom. We may then be interested to either compare the treatments in pairs or evaluate special contrasts depending upon the objectives of the experiment. This is done as follows:

The critical difference for testing the significance of the difference of two treatment effects, say $t_i - t_j$ is $C.D. = t_{(v-1)(r-1), \alpha/2} \sqrt{2MSE/r}$, where $t_{(v-1)(r-1), \alpha/2}$ is the value of Student's t at the level of significance α and degree of freedom $(v - 1)(r - 1)$. If the difference of any two-treatment means is greater than the C.D. value, the corresponding treatment effects are significantly different.

4. Latin Square Design

Latin square designs are normally used in experiments where it is required to remove the heterogeneity of experimental material in two directions. These designs require that the number of replications equal the number of *treatments* or *varieties*.

Definition 1. A Latin square arrangement is an arrangement of v symbols in v^2 cells arranged in v rows and v columns, such that every symbol occurs precisely once in each row and precisely once in each column. The term v is known as the **order** of the Latin square.

If the symbols are taken as A, B, C, D , a Latin square arrangement of order 4 is as follows:

A	B	C	D
B	C	D	A
C	D	A	B
D	A	B	C

A Latin square is said to be in the **standard form** if the symbols in the first row and first column are in natural order, and it is said to be in the **semi-standard form** if the symbols of the first row are in natural order. Some authors denote both of these concepts by the term **standard form**. However, there is a need to distinguish between these two concepts. The standard form is used for randomizing the Latin-square designs, and the semi-standard form is needed for studying the properties of the orthogonal Latin squares.

Definition 2. If in two Latin squares of the same order, when superimposed on one another, every ordered pair of symbols occurs exactly once, the two Latin squares are said to be **orthogonal**. If the symbols of one Latin square are denoted by Latin letters and the symbols of the other are denoted by Greek letters, the pair of orthogonal Latin squares is also called a **graeco-latin square**.

Definition 3. If in a set of Latin squares every pair is orthogonal, the set is called a set of **mutually orthogonal latin squares (MOLS)**. It is also called a **hypergraeco latin square**.

The following is an example of graeco latin square:

<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>	α	γ	δ	β	<i>A</i> α	<i>B</i> γ	<i>C</i> δ	<i>D</i> β
<i>B</i>	<i>A</i>	<i>D</i>	<i>C</i>	β	δ	γ	α	<i>B</i> β	<i>A</i> δ	<i>D</i> γ	<i>C</i> α
<i>C</i>	<i>D</i>	<i>A</i>	<i>B</i>	γ	α	β	δ	<i>C</i> γ	<i>D</i> α	<i>A</i> β	<i>B</i> δ
<i>D</i>	<i>C</i>	<i>B</i>	<i>A</i>	δ	β	α	γ	<i>D</i> δ	<i>C</i> β	<i>B</i> α	<i>A</i> γ

We can verify that in the above arrangement every pair of ordered Latin and Greek symbols occurs exactly once, and hence the two latin squares under consideration constitute a graecolatin square.

It is well known that the maximum number of MOLS possible of order v is $v - 1$. A set of $v - 1$ MOLS is known as a complete set of MOLS. Complete sets of MOLS of order v exist when v is a ***prime or prime power***.

Randomization

According to the definition of a Latin square design, treatments can be allocated to the v^2 experimental units (may be animal or plots) in a number of ways. There are, therefore, a number of Latin squares of a given order. The purpose of randomization is to select one of these squares at random. The following is one of the methods of random selection of Latin squares.

Let a $v \times v$ Latin square arrangement be first written by denoting treatments by Latin letters *A, B, C, etc.* or by numbers *1, 2, 3, etc.* Such arrangements are readily available in the ***Tables for Statisticians and Biometricians*** (Fisher and Yates, 1974). One of these squares of any order can be written systematically as shown below for a 5×5 Latin square:

<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>	<i>E</i>
<i>B</i>	<i>C</i>	<i>D</i>	<i>E</i>	<i>A</i>
<i>C</i>	<i>D</i>	<i>E</i>	<i>A</i>	<i>B</i>
<i>D</i>	<i>E</i>	<i>A</i>	<i>B</i>	<i>C</i>
<i>E</i>	<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>

For the purpose of randomization rows and columns of the Latin square are rearranged randomly. There is no randomization possible within the rows and/or columns. For example, the following is a row randomized square of the above 5×5 Latin square;

A	B	C	D	E
B	C	D	E	A
E	A	B	C	D
D	E	A	B	C
C	D	E	A	B

Next, the columns of the above row randomized square have been rearranged randomly to give the following random square:

E	B	C	A	D
A	C	D	B	E
D	A	B	E	C
C	E	A	D	B
B	D	E	C	A

As a result of row and column randomization, but not the randomization of the individual units, the whole arrangement remains a Latin square.

Analysis of Latin Square Designs

In Latin square designs there are three factors. These are the factors P , Q , and treatments. The data collected from this design are, therefore, analyzed as a three-way classified data. Actually, there should have been v^3 observations as there are three factors each at v levels. But because of the particular allocation of treatments to the cells, there is only one observation per cell instead of v in the usual three way classified orthogonal data. As a result we can obtain only the sums of squares due to each of the three factors and error sum of squares. None of the interaction sums of squares of the factors can be obtained. Accordingly, we take the model

$$Y_{ijs} = \mu + r_i + c_j + t_s + e_{ijs}$$

where y_{ijs} denotes the observation in the i^{th} row, j^{th} column and under the s^{th} treatment; $\mu, r_i, c_j, t_s (i, j, s = 1, 2, \dots, v)$ are fixed effects denoting in order the general mean, the row, the column and the treatment effects. The e_{ijs} is the error component, assumed to be independently and normally distributed with zero mean and a constant variance, σ^2 .

The analysis is conducted by following a similar procedure as described for the analysis of two-way classified data. The different sums of squares are obtained as below: Let the data be arranged first in a

row \times column table such that y_{ij} denotes the observation of (i, j) th cell of table.

Let $R_i = \sum_j y_{ij} = i^{th}$ row total ($i = 1, 2, \dots, v$), $C_j = \sum_i y_{ij} = j^{th}$ column total ($j = 1, 2, \dots, v$), $T_s =$ sum of those observations which come from s^{th} treatment ($s = 1, 2, \dots, v$), $G = \sum_i R_i =$ grand total. Correction

factor, $C.F. = \frac{G^2}{v^2}$. Treatment sum of squares = $\sum_s \frac{T_s^2}{v} - C.F.$, Row sum of squares = $\sum_i \frac{R_i^2}{v} - C.F.$,

Column sum of squares = $\sum_j \frac{C_j^2}{v} - C.F.$

Analysis of Variance of $v \times v$ Latin Square Design				
Sources of Variation	D.F.	S.S.	M.S.	F
Rows	$v - 1$	$\sum_i \frac{R_i^2}{v} - C.F.$		
Columns	$v - 1$	$\sum_j \frac{C_j^2}{v} - C.F.$		
Treatments	$v - 1$	$\sum_s \frac{T_s^2}{v} - C.F.$	s_t^2	s_t^2 / s_e^2
Error	$(v - 1)(v - 2)$	By subtraction	s_e^2	
Total	$v^2 - 1$	$\sum_{ij} y_{ij}^2 - C.F.$		

The hypothesis of equal treatment effects is tested by F -test, where F is the ratio of treatment mean squares to error mean squares. If F is not significant, treatment effects do not differ significantly among themselves. If F is significant, further studies to test the significance of any treatment contrast can be made in exactly the same way as discussed for randomized block designs.

SAS Code

Analysis of data obtained from experiment conducted under CRD setup

data crd;

input treatment yield;

cards;

1 850.5

1	453.6
1	878.85
1	623.7
1	510.3
1	765.45
1	680.4
1	595.35
1	538.65
1	850.5
1	850.5
1	793.8
1	1020.6
1	708.75
1	652.05
1	623.7
1	396.9
1	822.15
1	680.4
1	652.05
1	538.65
1	850.5
1	680.4
1	.
1	.
2	510.3
2	963.9
2	652.05
2	1020.6
2	878.85
2	567
2	680.4
2	538.65

2	567
2	510.3
2	425.25
2	567
2	623.7
2	538.65
2	737.1
2	453.6
2	481.95
2	368.55
2	567
2	595.35
2	567
2	595.35
2	.
2	.
2	.
3	992.25
3	850.5
3	1474.2
3	510.3
3	850.5
3	793.8
3	453.6
3	935.55
3	1190.7
3	481.95
3	623.7
3	878.85
3	1077.3
3	850.5
3	680.4

```
3      737.1
3      737.1
3      708.75
3      708.75
3      652.05
3      567
3      453.6
3      652.05
3      567
3      .
;
```

```
proc glm;
class treatment;
model yield = treatment;
means treatment;
means treatment/lsd;
run;
```

Analysis of data obtained from experiment conducted under RBD setup

```
data rbd;
input block treatment yield;
cards;
1      1      6.9
1      2      6.48
1      3      6.52
1      4      6.9
1      5      6
1      6      7.9
2      1      4.6
2      2      5.57
```

2	3	7.6
2	4	6.65
2	5	6.18
2	6	7.57
3	1	4.4
3	2	4.28
3	3	5.3
3	4	6.75
3	5	5.5
3	6	6.8
4	1	4.81
4	2	4.45
4	3	5.3
4	4	7.75
4	5	5.5
4	6	6.62

;

proc glm;

class block treatment;

model yield = block treatment;

means treatment;

means treatment/tukey;

run;

Analysis of data obtained from experiment conducted under LSD setup

data lsd;

input row column trt yld;

cards;

1	1	3	3.1
1	2	6	5.95
1	3	1	1.75

1	4	5	6.4
1	5	2	3.85
1	6	4	5.3
2	1	2	4.8
2	2	1	2.7
2	3	3	3.3
2	4	6	5.95
2	5	4	3.7
2	6	5	5.4
3	1	1	3
3	2	2	2.95
3	3	5	6.7
3	4	4	5.95
3	5	6	7.75
3	6	3	7.1
4	1	5	6.4
4	2	4	5.8
4	3	2	3.8
4	4	3	6.55
4	5	1	4.8
4	6	6	9.4
5	1	6	5.2
5	2	3	4.85
5	3	4	6.6
5	4	2	4.6
5	5	5	7
5	6	1	5
6	1	4	4.25
6	2	5	6.65
6	3	6	9.3
6	4	1	4.95
6	5	3	9.3

6 6 2 8.4

;

proc glm;

class blk trt;

model yld = blk trt;

means trt;

means trt/tukey;

run;

Database Management System

Dr. Shashi Dahiya

ICAR-Indian Agricultural Statistics Research Institute, New Delhi - 110 012

shashi.dahiya@icar.gov.in

What is Database?

An organised group of data that is kept and accessible electronically is referred to as a database. It is a digital repository that enables the effective management, storage, and retrieval of both organised and unorganised data. Information like client records, financial data, inventory listings, and much more can be stored in databases. Example of database are – Microsoft Access, OpenOffice Base, Oracle, MySQL and PostgreSQL etc.

What is a Database Management System?

Data is stored, retrieved, and analyzed using software called database management systems (DBMS). Users can create, read, update, and remove data in databases using a Database Management System, which acts as an interface between them and the databases. Example of database Management System are – Microsoft Access, OpenOffice Base, Oracle, MySQL and PostgreSQL etc.

Data can be organized into two types:

Flat File: Data is stored in a single table. Usually suitable for less amount of data. basically for small-scale organization where data does not need to be structured in a complex way. Example of database software are Microsoft Excel or Google Sheets.

Relational: Data is stored in multiple tables and the tables are linked using a common field. Relational is suitable for medium to large amount of data. Example of database software are MySQL, Microsoft SQL Server, and Oracle Database..

What is Database Server?

Database servers are dedicated computers that are designed to store data and provide database services to other computers. We use a database to store, organize, manage and retrieve data efficiently and effectively. The database servers run only database and database related software.

Advantages of Database

Reduces Data Redundancy

Data redundancy means when same data is stored more than one places, which increased complexity and wasted of storage space, so, database helps to reduce data redundancy.

Sharing of Data

A database allows its users to exchange data among themselves. The data can only be shared with users who have received the appropriate degrees of authorization because there are different levels of access to the data.

Basic Definitions:

Data Integrity

Data integrity refers to the accuracy and consistency of data stored in a database management system (DBMS). Database ensures that the data is reliable and can be trusted for decision making and other critical business processes.

Data Security

A database's concept of data security is important. The database should only be accessible to authorised users, whose identities must be verified using a username and password.

Privacy

The privacy rule in a database ensures that only authorized users can access the database and view data according to the specific privacy constraints. To maintain data security, access levels are set in the database so that a user can only view data that they are not allowed to modify if the permission is not given.

For example, in social networking sites, different accounts have different access constraints, and users are only allowed to view the other account data that is permitted for their specific account.

Backup and Recovery

Database Management System automatically take backup and recovery processes to ensure the protection and availability of data. This means that the DBMS automatically creates and manages backups of the database at regular intervals, and also provides tools to recover data in the event of a failure or data loss.

Data Consistency

Data consistency refers to the accuracy and reliability of data stored in a database or other data storage system. It means that the data is consistent and valid across all instances where it is stored or accessed.

For example, if a customer's name and address are stored in multiple tables within a database, the data must be consistent across all table.

Types of Databases and Database Applications

- **Traditional Applications:**
 - Numeric and Textual Databases
- **More Recent Applications:**
 - Multimedia Databases
 - Geographic Information Systems (GIS)
 - Data Warehouses
 - Real-time and Active Databases
 - Many other applications

Some Indicative examples of types of Data in ICAR Institutes

- Laboratory/field measurements and observations;
- Germplasm related information;
- Information on products such as vaccines, bio-pesticides, equipments, software,

- source code for software/information systems, both data as well as material;
- Novel microbes or virus isolates of scientific/commercial importance;
- Technologies, processes, methodologies;
- Information related to surveys, sample survey data, bio-prospecting, know-how etc. ;
- Publications, photographs, audio-visual materials, databases, drawings, etc.;
- Information related to IPR applications, technology commercialisation etc.;
- Information of services functions (e.g. training, consultancy, contract research, contract service, infrastructure facilities, liaison, other facilitation/ partnerships);
- Information/data on the compliance of guidelines and special regulations that apply to the project's implementation such as those involving exchange of genetic material, bio safety, handling of hazardous materials etc.

Typical DBMS Functionality

- Define a particular database in terms of its data types, structures, and constraints
 - Construct or Load the initial database contents on a secondary storage medium
 - Manipulating the database:
 - Retrieval: Querying, generating reports
 - Modification: Insertions, deletions and updates to its content
 - Accessing the database through Web applications
 - Processing and Sharing by a set of concurrent users and application programs – yet, keeping all data valid and consistent
 - Other features:
 - Protection or Security measures to prevent unauthorized access
 - “Active” processing to take internal actions on data Presentation and Visualization of data
- Maintaining the database and associated programs over the lifetime of the database application called database, software, and system maintenance

Simplified database system environment

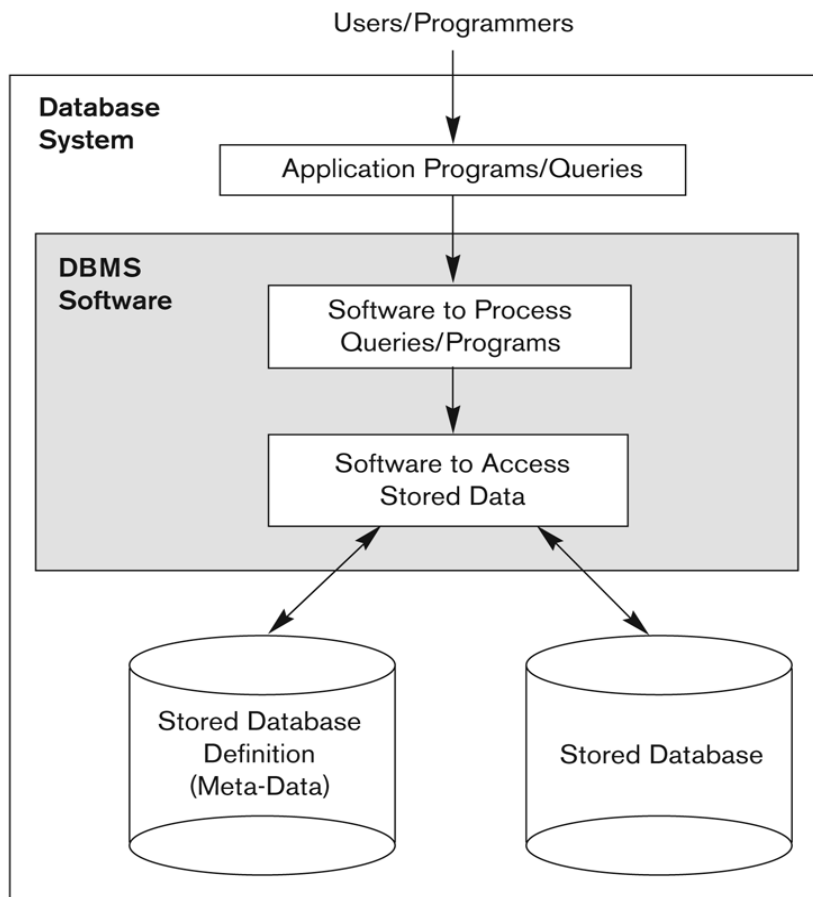


Figure 1.1
A simplified database system environment.

Example of a Database

- Mini-world for the example:
 - Part of a UNIVERSITY environment.
- Some mini-world entities:
 - STUDENTs
 - COURSEs
 - SECTIONs (of COURSEs)
 - (academic) DEPARTMENTs
 - INSTRUCTORs

Example of a Database (with a Conceptual Data Model)

- Some mini-world relationships:
 - SECTIONs are of specific COURSEs
 - STUDENTs take SECTIONs

- COURSEs have prerequisite COURSEs
 - INSTRUCTORs teach SECTIONs
 - COURSEs are offered by DEPARTMENTs
 - STUDENTs major in DEPARTMENTs
- Note: The above entities and relationships are typically expressed in a conceptual data model, such as the ENTITY-RELATIONSHIP data model

Example of a simple database

COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	04	King
92	CS1310	Fall	04	Anderson
102	CS3320	Spring	05	Knuth
112	MATH2410	Fall	05	Chang
119	CS1310	Fall	05	Anderson
135	CS3380	Fall	05	Stone

GRADE_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

Main Characteristics of the Database Approach

- Self-describing nature of a database system:

- A DBMS catalog stores the description of a particular database (e.g. data structures, types, and constraints)
- The description is called meta-data.
- This allows the DBMS software to work with different database applications.
- Insulation between programs and data:
 - Called program-data independence.
 - Allows changing data structures and storage organization without having to change the DBMS access programs.

Example of a simplified database catalogue

RELATIONS

Relation_name	No_of_columns
STUDENT	4
COURSE	4
SECTION	5
GRADE_REPORT	3
PREREQUISITE	2

COLUMNS

Column_name	Data_type	Belongs_to_relation
Name	Character (30)	STUDENT
Student_number	Character (4)	STUDENT
Class	Integer (1)	STUDENT
Major	Major_type	STUDENT
Course_name	Character (10)	COURSE
Course_number	XXXXNNNN	COURSE
....
....
....
Prerequisite_number	XXXXNNNN	PREREQUISITE

Main Characteristics of the Database Approach

Data Abstraction:

A data model is used to hide storage details and present the users with a conceptual view of the database. Programs refer to the data model constructs rather than data storage details.

Support of multiple views of the data:

Each user may see a different view of the database, which describes only the data of interest to that user.

Main Characteristics of the Database Approach

Sharing of data and multi-user transaction processing:

- Allowing a set of concurrent users to retrieve from and to update the database.
- Concurrency control within the DBMS guarantees that each transaction is correctly executed or aborted
- Recovery subsystem ensures each completed transaction has its effect permanently recorded in the database
- OLTP (Online Transaction Processing) is a major part of database applications. This allows hundreds of concurrent transactions to execute per second.

Database Users

Users may be divided into

- Those who actually use and control the database content, and those who design, develop and maintain database applications (called “Actors on the Scene”), and
- Those who design and develop the DBMS software and related tools, and the computer systems operators (called “Workers Behind the Scene”).

Actors on the scene:

Database administrators: Responsible for authorizing access to the database, for coordinating and monitoring its use, acquiring software and hardware resources, controlling its use and monitoring efficiency of operations.

Database Designers: Responsible to define the content, the structure, the constraints, and functions or transactions against the database. They must communicate with the end-users and understand their needs.

Categories of End-users

End-users: They use the data for queries, reports and some of them update the database content.

End-users can be categorized into:

- **Casual:** Access database occasionally when needed
- **Naïve or Parametric:** They make up a large section of the end-user population. They use previously well-defined functions in the form of “canned transactions” against the database. Examples are bank-tellers or reservation clerks who do this activity for an entire shift of operations.

- **Sophisticated:** These include business analysts, scientists, engineers, others thoroughly familiar with the system capabilities. Many use tools in the form of software packages that work closely with the stored database.
- **Stand-alone:** Mostly maintain personal databases using ready-to-use packaged applications. An example is a tax program user that creates its own internal database. Another example is a user that maintains an address book

Features of Database

There are some key features of a database:

One or more tables can be added in the database.

Decreased storage costs and space requirements

Users can use query languages in a database.

Multiple users can access the data from the database .

Unique keys aid in preventing errors caused by human or technological mishaps.

Primary Key, Composite Primary Key and Foreign Key in a Database

In the RDBMS data can be integrated using keys. These are Primary Key, Composite Primary Key, and Foreign Key, Key are used to make the relationship between the tables.

Primary Key – This unique field is called the Primary Key (PK). primary key is a field or a set of fields that uniquely identify each record in a table. A primary key must be unique and cannot contain null values.

Composite Primary Key – A composite primary key is a primary key that consists of two or more fields that together uniquely identify each record in a table.

Foreign Key – A composite primary key is a primary key that consists of two or more fields that together uniquely identify each record in a table.

What is RDBMS?

A database management system that is based on the relational model is called an RDBMS (Relation Database Management System). Tables are used to organise data in relational databases. A relational database management system (RDBMS) is used to store, manage, query, and retrieve data.

Data Types

The type of data (value) that will be stored in the database is defined by its datatype. Important to know the different types of data helps to ensure that each property's value is as expected and that data is collected in the correct format.

Data types in OpenOffice base are broadly classified into five categories listed below.

- Numeric Types
- Alphanumeric Types
- Binary Types
- Date time
- Other Variable types

Database Objects

a. Tables: Data is arranged into rows and columns in a table, which is a type of data structure. It can be applied to both the storage and presentation of structured data.

b. Columns or Fields or Attributes: Data is arranged vertically from top to bottom in columns. Each row of the table has one column, which is a collection of data values of a specific basic type. The structure by which the rows are put together is provided by the columns.

c. Rows or Records or Tuples: A row, also known as a Record or Tuple, in a table represents a single data item. A database table can be represented graphically as being made up of rows and columns, or fields. Every row in a table has the same structure and represents a group of connected data.

Numeric Types

Numerical data types are data types that store numeric values in a database. Numeric data types can be further divided into several subtypes, including:

Name	Data type	Description
BOOLEAN	Yes / No	Values as 0 or 1. Example: True or False, Yes or No.
TINYINT	Tiny Integer	Store integer range between 0 to 255
SMALLINT	Small Integer	Store integer range between -2^{15} to $+2^{15}-1$
INTEGER	Integer	Store integer range between -2^{31} to $+2^{31}-1$
BIGINT	Big Integer	Range between -2^{63} to $+2^{63}-1$
NUMERIC	Number	Maximum precision of $e^{(+/-)251}$
DECIMAL	Decimal	Maximum precision of $e^{(+/-)251}$
REAL	Real	2^{-1074} to $(2 \cdot 2^{-52}) \cdot 2^{1023}$
FLOAT	Float	2^{-1074} to $(2 \cdot 2^{-52}) \cdot 2^{1023}$
DOUBLE	Double	2^{-1074} to $(2 \cdot 2^{-52}) \cdot 2^{1023}$

Alphanumeric Types

Data that has both letters and numbers is referred to as alphanumeric type.

Name	Data type	Description
LONGVARCHAR	Memo	Stores up to the max length or number indicated by user. It accepts any UTF 8 Character.
CHAR	Text (fix)	Stores exactly the length specified by user. Pads with trailing spaces for shorter strings. Accepts any UTF 8 Character.
VARCHAR	Text	Stores up to the specified length. No padding (Same as long var char)
VARCHAR_IGNORE CASE	Text	Stores up the specified length. Comparisons are not case sensitive but stores capitals as you type them.

Binary Types

For storing data in binary formats, binary data types are utilised. In a database, binary data types can be used to store things like music and image files. The binary data type can generally be used to store files in any format.

Name	Data type	Description
LONGVARBINARY	Image	Stores any array of bytes (images, sounds, etc.). No validation required.
BINARY	Binary (fix)	Stores any array of bytes. No validation required.
VARBINARY	Binary	Stores any array of bytes. No validation required.

Date Time

When specifying date and time values for a column used in a database table, date time data types are used. Information like dates of birth, admissions, product sales, and other dates can be stored in databases using date and time data types.

Name	Description	Format
Date	Stores month, day and year information	1/1/99 to 1/1/9999
Time	Stores hour, minute and second information	Seconds since 1/1/1970
Timestamp	Stores date and time information	

Other Data Types

Name	Description
Other/Object	Stores serialized Java objects â€™ user application must supply serialization routines

Following are some properties of data of the numeric type:

AutoValue – if set to yes then field will get the auto numeric values.

Length – By default length of the field is 10 but the size of the field can be set to maximum length.

Default Value – A default value can be set for a field if user don't provide any value while entering the values in the table.

Format example – This property helps to set the format of the data entered in the field such as 91-222-333.

Following are some properties of data of the character type:

Entry Required – if set to yes then it will be must to insert the value in the field which means that field cannot be left blank.

Length – By default length of the field is 10 but the size of the field can be set to maximum length.

Default Value – A default value can be set for a field if user don't provide any value while entering the values in the table.

Format example – This property helps to set the format of the data entered in the field such as 91-222-333.

Referential Integrity

The relationship between tables is referred to as referential integrity. Referential integrity is used to maintain accuracy and consistency of data in a relationship. In Base, data can be linked between two or more tables with the help of primary key and foreign key constraints.

Referential integrity helps to avoid:

1. Adding records to a related table if there is no associated record available in the primary key table.
2. Changing values in a primary if any dependent records are present in associated table(s).
3. Deleting records from a primary key table if there are any matching related records available in associated table(s).

Creating and Editing Relationships between Tables

An association or link between two or more tables is referred to as a relationship. You don't have to enter the same data again in different tables when you relate two tables.

Relationships between tables helps to

- Save time as there is no need to enter the same data in separate tables.
- Reduce data-entry errors.
- Summarize data from related tables.

Type of Relationships in Database

There are three types of relationships which can be created in tables:

1. ONE to ONE
2. ONE to MANY OR MANY to ONE
3. MANY to MANY

ONE to ONE

In this relationship, both the tables must have primary key columns.

ONE to MANY OR MANY to ONE

In this relationship, one of the table must have primary key column. It signifies that one column of primary key table is associated with all the columns of associated table.

MANY to MANY

In this relationship, no table has the primary key column. It signifies that all the columns of primary key table are associated with all the columns of associated table.

Retrieve Data Using Query

In order to describe the data structure and to modify the data in the database, queries are used as instructions. A query enables the joining and filtering of data from various tables.

Database Languages having two type:

- DDL (Data Definition Language)
- DML (Data Manipulation Language)

DDL Statements:

- Create: Using this statement, a database or set of tables can be created.
- Alternate: This statement is used to change the table's structure.
- Drop: This statement is used to remove database objects from the system.

DML statements:

SELECT: The statement "SELECT" is used to get data from the database.

INSERT: The statement "INSERT" is used to add a new record to the database.

DELETE: The database can be cleaned out by using the statement DELETE.

UPDATE: This statement is used to modify the database's information.

Database Query

Query is a computer languages. In order to describe the data structure and to modify the data in the database, queries are used as instructions. Query can extract particular data from a database. We can filter and join data from various tables with the help of a query. By using the criteria you supply query will filter the data.

Select Statement

A select query is a language in a database that displays data in Datasheet view. Data from tables is displayed by a query rather than being stored by it. A query may display data from one or more tables, from other queries, or from both of these sources simultaneously.

The SELECT statement has many optional clauses:

WHERE specifies which rows to retrieve.

ORDER BY specifies an order in which to return the rows.

Syntax of Select Statement is –

```
SELECT * FROM <TABLENAME>;
```

base-management-system-class-10-notes/

Query related to Simple Select Statement –

Table Name – product

Product_No	Product_Name	Price	Quantity
25	Soap	40	80
31	Powder	80	30
45	Shampoo	250	25
52	Soap Box	120	100

Question – Write a Query to display all record from the table;

```
Select * from product;
```

Output –

Product_No	Product_Name	Price	Quantity
25	Soap	40	80
31	Powder	80	30
45	Shampoo	250	25
52	Soap Box	120	100

Question – Write a Query to display product name from the table;

```
Select Product_Name from product;
```

Output –

Product_Name
Soap
Powder
Shampoo
Soap Box

Advantages of Using the Database Approach:

- Controlling redundancy in data storage and in development and maintenance efforts. Sharing of data among multiple users.
- Restricting unauthorized access to data.
- Providing persistent storage for program Objects
- Providing Storage Structures (e.g. indexes) for efficient Query Processing
- Providing backup and recovery services.
- Providing multiple interfaces to different classes of users.
- Representing complex relationships among data.
- Enforcing integrity constraints on the database.
- Drawing inferences and actions from the stored data using deductive and active rules

Additional Implications of Using the Database Approach:

Potential for enforcing standards: This is very crucial for the success of database applications in large organizations. Standards refer to data item names, display formats, screens, report structures, meta-data (description of data), Web page layouts, etc.

Reduced application development time: Incremental time to add each new application is reduced.

Flexibility to change data structures: Database structure may evolve as new requirements are defined.

Availability of current information: Extremely important for on-line transaction systems such as airline, hotel, car reservations.

Economies of scale: Wasteful overlap of resources and personnel can be avoided by consolidating data and applications across departments.

Historical Development of Database Technology:

Early Database Applications: The Hierarchical and Network Models were introduced in mid 1960s and dominated during the seventies. A bulk of the worldwide database processing still occurs using these models, particularly, the hierarchical model.

Relational Model based Systems: Relational model was originally introduced in 1970, was heavily researched and experimented within IBM Research and several universities. Relational DBMS Products emerged in the early 1980s.

Object-oriented and emerging applications: Object-Oriented Database Management Systems(OODBMSs) were introduced in late 1980s and early 1990sto cater to the need of complex data processing in CAD and other applications. Their use has not taken off much. Many relational DBMSs have incorporated object database concepts, leading to a new category called object-relational DBMSs (ORDBMSs). Extended relational systems add further capabilities (e.g. for multimedia data, XML, and other data types)

Data on the Web and E-commerce Applications: Web contains data in HTML (Hypertext markup language) with links among pages.

Extending Database Capabilities:

New functionality is being added to DBMSs in the following areas:

- Scientific Applications
- XML (eXtensible Markup Language)
- Image Storage and Management
- Audio and Video Data Management

- Data Warehousing and Data Mining
- Spatial Data Management
- Time Series and Historical Data Management

The above gives rise to new research and development in incorporating new data types, complex data structures, new operations and storage and indexing schemes in database systems.

Web Application Development using HTML

Alka Arora

ICAR-Indian Agricultural Statistics Research Institute, New Delhi - 110 012

alka.arora@icar.gov.in

Background

A **web application** is a client-server software application in which the client, typically a web browser, interfaces with the server over a network. The client-side, executed within the browser, is responsible for presenting the user interface and handling user interactions. It employs technologies such as HTML, CSS, and JavaScript to create a dynamic and interactive user experience. The server-side, hosted on remote servers, manages the application's logic, data storage, and processing. Communication between the client and server occurs through standardized protocols such as HTTP. Web applications are designed to be accessed over the internet, providing users with a platform-independent experience. They often interact with databases to store and retrieve data, and their architecture allows for scalability, as multiple users can simultaneously access and utilize the application. The modular nature of web applications enables developers to update and maintain the software centrally, ensuring users receive the latest features and improvements without requiring manual installations. Examples include online collaboration tools, social media platforms, and e-commerce websites.

Introduction

Web application development using HTML involves creating the structural foundation of interactive and dynamic web pages. **HTML**, or **Hypertext Markup Language**, serves as the standard markup language for constructing the elements and layout of web content. In the context of technical language. HTML5, or Hypertext Markup Language version 5, is the latest evolution of the standard markup language used to create and structure content on the World Wide Web. It is a major revision of the HTML specification and includes new features, capabilities, and improvements over its predecessor, HTML 4.01. HTML5 was officially released as a W3C Recommendation in October 2014.

HTML has evolved continuously over time, with each version introducing new features and improvements

HTML 1.0 (Level 0):	HTML 1.0 was the first version of HTML, introduced in 1991. It included basic structural elements like headings, paragraphs, lists,
----------------------------	---

	and links. It did not have many of the features we consider essential for modern web development.
HTML 2.0 (Level 1):	HTML 2.0 was published as an Internet Engineering Task Force (IETF) standard in 1995. It introduced support for hypertext and graphics. This version started to include basic styling and text formatting.
HTML 3.2 (Level 2):	HTML 3.2, released in 1997, brought improvements such as tables and form elements, making web pages more interactive. This version marked a significant step forward in terms of functionality and layout possibilities.
HTML 4.01 (Level 3):	HTML 4.01, published in 1999, introduced features like frames, scripting support, and improved support for multimedia elements. It aimed to provide a more standardized approach to web development.
XHTML (eXtensible HyperText Markup Language):	XHTML was an attempt to bring HTML into the world of XML, providing stricter syntax rules. It was based on HTML 4, but with a syntax that followed XML rules. XHTML 1.0 and 1.1 were introduced.
HTML5:	HTML5 is the latest and ongoing version of HTML. It started gaining traction around 2010. HTML5 introduced a wide range of new features, including native support for audio and video, the <canvas> element for drawing graphics, local storage, and semantic elements like <header>, <nav>, <article>, and <footer>. It also improved support for forms, introduced the <svg> element for scalable vector graphics, and much more.

Key features and enhancements of HTML5 include:

1. Semantics: HTML5 introduces new semantic elements such as <header>, <footer>, <nav>, <article>, <section>, and <aside>. These elements provide a more meaningful structure to web documents, making it easier for both developers and browsers to understand the content's purpose.

2. **Multimedia Support:** HTML5 includes native support for embedding audio and video content without the need for third-party plugins like Adobe Flash. The <audio> and <video> elements allow developers to include media directly in web pages.
3. **Offline Web Applications:** HTML5 introduces the Application Cache (App Cache), which enables web applications to work offline by storing resources locally. This is useful for creating web apps that can continue to function even when the user is not connected to the internet.
4. **Web Storage:** HTML5 provides two types of local storage options: session Storage and local Storage. These allow web applications to store data on the client side, persisting across sessions or tabs.
5. **Geolocation:** HTML5 supports geolocation services, allowing web applications to access and use the device's location information through the Geolocation API.
6. **Responsive Web Design:** HTML5 supports responsive web design principles, making it easier to create web pages that adapt and provide an optimal viewing experience across various devices and screen sizes.

HTML5 has become a fundamental technology for modern web development, providing developers with the tools to create more feature-rich and interactive web applications. Its adoption has been widespread, and it forms the basis for many contemporary web technologies and frameworks.

Creating a Simple HTML Document

An HTML document is created with the help of elements. It is similar to writing a letter where you have to specify the address part, then the subject followed by the body of the letter and the end part. In an HTML document also you have to specify the head, body and the title part with the help of elements. An element consists of a start-tag, an-end tag and the data characters enclosed by the two tags.

Element names are written in upper case letters (e.g., BODY). Attribute names are written in lower case letters (e.g., lang, onsubmit). Recall that in HTML, element and attribute names are case-insensitive; the convention is meant to encourage readability. Elements can have attributes to emphasize on the particular way the programmer wants the documents to appear.

A tag starts with a less than (<) sign and ends with a greater than (>) sign. Tag and attribute names are not case-sensitive, but are typically written in uppercase to distinguish them from the data characters. An end-tag consists of the tag name immediately preceded by a slash (/). Some tags require that you always provide the matching end-tag; others allow you to omit the end tag if the result is clear and unambiguous.

- a) **Document Type Declaration (<!DOCTYPE html>):**

Declares the document type as HTML5.

b) HTML element.

Marks the beginning of the HTML document. The 'Lang' attribute specifies the language.

`<HTML>`: Starts the HTML Code

`</HTML>`: Ends the HTML Code

c) <head> Element: After the opening `<html>` tag, the next command is the `<head>` element in HTML. The `<head>` element is crucial for organizing metadata and other elements that provide information about the document.

`<meta charset="UTF-8">`:

Specifies the character encoding as UTF-8.

`<meta name="viewport" content="width=device-width, initial-scale=1.0">`:

Configures the viewport for responsive design.

<title> Element: The `<title>` element is used to specify the title of the HTML document. This title is displayed on the title bar or tab of the browser. For example, in Internet Explorer, it appears on the title bar of the window.

The `<title>` element must be placed within the `<head>` section of the HTML document.

<style> Element: The `<style>` element in HTML is used to embed CSS (Cascading Style Sheets) directly within an HTML document. It allows developers to define presentation rules, such as colors, fonts, and layout, for the content of the document.

Inside the `<style>` tag, we add optional CSS styles to give a basic visual structure to the document.

```
<!DOCTYPE html>
<html Lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>My Simple HTML5 Document</title>
<style>
/* Optional: Add some basic styles */
body {
font-family: Arial, sans-serif;
margin: 20px;
}
```

```
</style>  
</head>
```

d) <body> Element: The <body> element in HTML defines the main content of a web page, encompassing text, images, links, and other elements that are visible to the user. It serves as the container for the actual content that is displayed in the browser window.

The <BODY> command allows you to set the background picture, the font colors (i.e. regular text, links, visited links), and houses certain commands for JavaScript, and VBScript.

```
<BODY BACKGROUND="Your Background Picture" BGCOLOR="Color of Background"  
TEXT="Main Text Color" LINK="Color Of Your Links" >
```

NOTE:

- In the <BODY> command, the BGCOLOR is not needed if the BACKGROUND has a value. If both are filled in with a value, then the BACKGROUND overrides the BGCOLOR.
- After specifying the background (bgcolor) color and the text (text) color the screen appears differently.
- Here is the syntax of a very simple HTML document:

Example:

```
<HTML>  
<HEAD> <TITLE>A study of population dynamics</TITLE></HEAD>  
<BODY bgcolor="white" text="black"> ... document body...</BODY>  
</HTML>
```

The attributes that we can use with the BODY tag are:

- background = url

The value of this attribute is a URL that designates an image resource. The image generally tiles the background (for visual browsers).

- text = color

This attribute sets the foreground color for text (for visual browsers).

- link = color

This attribute sets the color of text marking unvisited hypertext links (for visual browsers).

- vlink = color

This attribute sets the color of text marking visited hypertext links (for visual browsers).

- alink = color

This attribute sets the color of text marking hypertext links when selected by the user (for visual

browsers).

We shall use these attributes as and when necessary.

The colors that have been used here are the six-digit number and letter combinations represent colors by giving their RGB (red, green, blue) value. The six digits are actually three two-digit numbers in sequence, representing the amount of red, green, or blue as a hexadecimal value in the range 00-FF. For example, 000000 is black (no color at all), FF0000 is bright red, 0000FF is bright blue, and FFFFFFFF is white (fully saturated with all three colors).

For some basic colors -- typically those in the standard sixteen-color Windows 3.1 palette -- you can also use the name of the color instead of the corresponding RGB value. For example, "black", "red", "blue", and "cyan" are all valid for use in place of RGB values. However, while not all browsers will understand all color names, any browser that can display colors will understand RGB values, so use them whenever possible.

Paragraph Setting and Alignment

In the body part of the document you can set the line space and a new paragraph with the following elements:

**
**

This element inserts a line break in the paragraph and rest of the characters will appear on the next line.

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Paragraph Styling and Alignment</title>
</style>
  /* Internal CSS for paragraph styling */
  p {
    color: blue;
    font-size: 16px;
  }

  /* Internal CSS for paragraph alignment */
  .center-align {
    text-align: center;
  }
```

```
</style>
</head>
<body>
  <!-- Usage of styles for paragraphs -->
  <p>This is a styled paragraph.</p>
  <p class="center-align">This paragraph is center-aligned.</p>
</body>
</html>
```



<P>

This element inserts a paragraph break and denotes a paragraph. The end tag is optional.

Example: showing only the body section of the HTML document.

```
<BODY>
  This is a paragraph. <br>
  This is the end of the paragraph.
<P> This is the new paragraph with paragraph element. </P>
</BODY>
```

This is the new paragraph with Paragraph element.

The paragraph element also has an align attribute, as follows:

align = left | center | right | justify

This attribute specifies the horizontal alignment of its element with respect to the surrounding context.

Possible values:

- left: text lines are rendered flush left.
- center: text lines are centered.
- right: text lines are rendered flush right.
- justify: text lines are justified to both margins.

Dividing Sections Using <HR> tag

A simple and effective way to separate sections within a Web page is to insert a horizontal line, <HR>, which is also called a **horizontal rule**. By default, the line stretches from one side of the page to other.

The <HR> tag takes several optional attributes. For example, you can specify the lines thickness and how much of the browsers window it should span (as a percentage or in pixels)

```
<HR SIZE = "6" WIDTH = "60%">
```

This will display a line six pixels thick that spans 60 percent of the browsers window.

Creating a Hierarchy with Headings

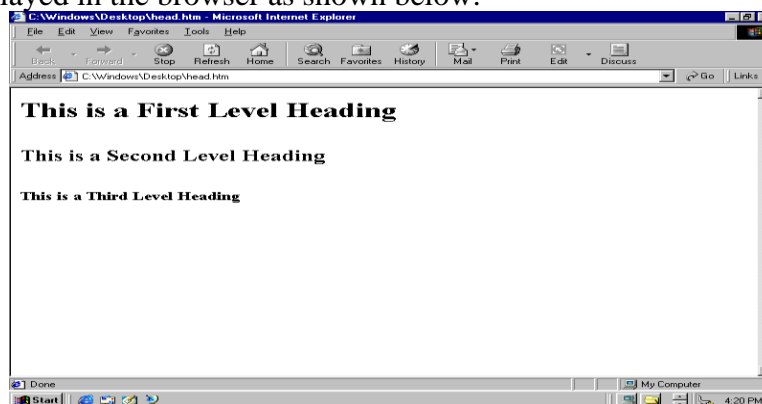
A common way to add structure to a Web page is through the use of headings. A Web page can have a maximum of six levels of headings, the HTML codes for which is conveniently named <H1>, <H2>, <H3>, and so on..

Web browsers might interpret the look of a heading in a slightly different way. Structurally, however all browsers will display headings so that a third-level heading looks subordinate to a second-level heading, a second level heading look subordinate to first-level heading, and so on.

For example

```
<H1>This is a First Level Heading</H1><BR><H2>This is a Second Level  
Heading</H2><br><H3>This is a Third Level Heading</H3>
```

This code will be displayed in the browser as shown below:



Text Formatting and Links

 and <BASEFONT>

The element changes the font size and color for text in its contents. If you use the FONT tag to change text size, you can specify either a fixed or relative size. A fixed size is a number in the range 1 through 7.

* this text is of size 3 *

A relative size is a positive or negative number, preceded by the plus (+) or minus (-) sign, that indicates a size that is relative to the base font size, as set using the <BASEFONT> tag. The following example shows the effect of using relative sizes:

<BASEFONT SIZE=3> This sets the base font size to 3.

** Now the font size is 7.

** Now the font size is 2.

Note: To get back to the basefont end the font tag , else the end tag is optional.

The attributes of the tag are

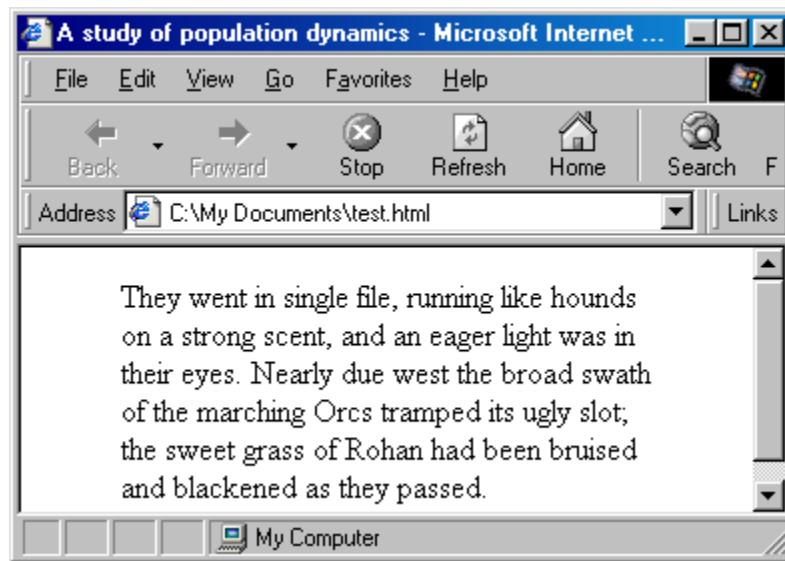
- **color = color**

This attribute sets the text color.

- **face = cdata**

This attribute defines a comma-separated list of font names the user agent should search for in order of preference. You can also use the FACE= attribute with the FONT tag to set the facename of the font used for text. Typical facenames are "Arial", "Times New Roman", and "Courier New", but you can use the facename of any font installed on the device on which your HTML document is viewed. The following

example sets the "Arial" font for the text in a section of the document:



Example:

```
<BODY>
<P>This word is <B>bold</B>; <BR>
this word is <I>italic</I>; <BR>
this word is <U>underlined.</U>;<BR>
<FONT SIZE=4 FACE = "arial"> Size of the alphabets is large and the typeface is arial.</FONT>
</BODY>
```

Creating Lists in HTML

In the List formatting we can create a list, menu, definition etc. to represent the information in the particular format like the content of items can be display in the list format or in the directory format. You can create a variety of lists in your document by using the UL, OL, MENU, and DIR tags in conjunction with the LI tag. You can also create definition lists, which give you a simple two-column list for terms and their definitions.

One can create a bulleted list, consisting of individual items preceded by a bullet character, by using UL and LI, as in the following example:

```
<UL>
<LI>Bulleted Lists
<LI>Ordered Lists
<LI>Directory Lists
</UL>
```

➤ The output will be: **Bulleted Lists**

Ordered Lists
Directory Lists

One can use the OL and LI tags to create an ordered list. The list consists of individual items that are sequentially numbered or lettered. To set the style of numbering or lettering, you use the TYPE= attribute in OL. Similarly, you use the START= attribute to set the initial number or letter. By default, the style is integer numbers starting at 1.

```
<OL START=1>
<LI>Bulleted Lists
<LI>Ordered Lists
<LI>Directory Lists
</OL>
```

- The output will be: **Bulleted Lists**
Ordered Lists
Directory Lists

<DIR> and <MENU>

The <DIR> element was designed to be used for creating multi-column directory lists. The <MENU> element was designed to be used for single column menu lists. Both elements have the same structure as , just different in rendering. In practice the user agent will render <DIR> or <MENU> list exactly as list. We strongly recommend using instead of these elements.

<DL>

A definition list (coded as <DL>) usually consists of alternating a definition term (coded as <DT>) and a definition (coded as <DD>). Web browsers generally format the definition on a new line and indent it.

The following is an example of a definition list:

```
<DL>
<DT> IASRI
<DD> IASRI, Indian Agricultural Statistics Research Institute,
      is located on the campus of the pusa at New Delhi.
<DT> Computer Center
<DD> It is located on the campus of IASRI, New Delhi.
</DL>
```

Nested Lists

Lists can be nested. You can also have a number of paragraphs, each containing a nested list, in a single list item.

Here is a simple nested list:

```
<UL>
<LI> A few New
Indian states:
    <UL>
    <LI> Madhya Pradesh
    <LI> Uttar Pradesh
    <LI> Tamil Nadu
    </UL>
<LI> Two Union Territories:
    <UL>
    <LI> Daman & Due
    <LI> Goa
    </UL>
</UL>
```

<Mailto>

You can make it easy for a reader to send electronic mail to a specific person or mail alias by including the mailto attribute in a hyperlink. The syntax for the mailto element is as follows:

```
<A HREF="mailto:emailinfo@host">Name</a>
For example, enter:
<A HREF="mailto:pubs@ncsa.uiuc.edu">
NCSA Publications Group</a>
```

to create a mail window that is already configured to open a mail window for the NCSA Publications Group alias. (You, of course, will enter another mail address!)

Images

The Command (Image Command)

Images are just as important as links. People like to see pretty pictures, and often tend to wait longer for the graphics to load than to look at plain text. Images, though, can have many different purposes. They can be static, have links associated with them, or they can be moving. You can include images using the image element in any Web page to provide information or to make the page more attractive.

```
<IMG SRC="Location of Image File" WIDTH="Width of Image" HEIGHT="Height of Image"
BORDER="1 to 6, Size of Border">
```

Images that you include in Web pages are called inline images because the images are inserted within a

line of body text. Because the image element is a text level element, it should be nested inside a paragraph or other block level container, and it doesn't start a new paragraph automatically.

To make an image appear as a separate paragraph, enclose it within the paragraph element like this.

```
<P>  
<IMG SRC=" http://www.emf.net / estephen / images/turtleshirt.jpg ">  
</P>
```

If you have the image in the same directory as your HTML file, you can abbreviate the URL and use a tag like this

```
<IMG SRC=" turtleshirt.jpg ">
```

This inserts the image called turtleshirt.jpg on a page.

Using Image Element Attributes:

The tag's attributes are principally intended to tell a browser how the page should be laid out with the image so that text can flow properly around the image.

Describing Images with Alternate Text

You should always use two attributes with any tag: the SRC and ALT attributes, both of which are required. The Alt attribute is used to describe the image in some way. For any browser that isn't displaying images, the alternate text contained inside the ALT attribute is displayed instead. Here's an example of image element using alternate text:

```
<IMG SRC=" images/turtleshirt.jpg " ALT =" Micky Mouse ">
```

If you use this tag, browser can display the words "Micky Mouse" instead of displaying an image.

Placing Images with Alignment Attributes

When you align images with an alignment attribute (ALIGN), there are two entirely separate results:

- Inline images occur in the middle of a line text. If the image is a large one, then the line becomes very tall, and a lot of white space will appear.
- Floating images cause text to wrap around the image. Images can either be left- aligned or right-aligned. The paragraph will flow around the image for several lines, if the image is large.
- The two different behaviors are caused by choosing the attribute value for ALIGN.
- To align an image in a line, choose one of the following attributes for the image element:

ALIGN= "TOP"

ALIGN= "MIDDLE"

ALIGN= "BOTTOM"

The default behavior is **ALIGN= "BOTTOM"**, which means that bottom of an image will align with the bottom of the line of the text. By choosing **ALIGN= "TOP"**, You request that the browser display the top of your image so that that it aligns with the top of the line of the text. Similarly, by choosing **ALIGN= "MIDDLE"** the browser will align the middle of the image with the middle of the line of the text.

Creating Floating Images

To make an image “float” to the left or right side and cause paragraphs to wrap around the image, choose one of the following two attribute values for the ALIGN attribute.

ALIGN= "LEFT"

ALIGN= "RIGHT"

Choosing LEFT or RIGHT as the value for ALIGN causes the image to be placed directly against the left or right margin. The text after the tag will flow around the image.

One drawback is that the horizontal rule (from the `<HR>` tag) and the last paragraph may be next to the picture. We might want to push these items down so they're below the image. If we use the `CLEAR` attribute and the appropriate margin value, then the horizontal rule and the last paragraph will be forced down below the image, for example if the image is on the right margin, then we will use `<BR CLEAR="RIGHT">` tag (placed immediately below the `<HR>` tag or before the `</P>` tag). If your page has images on both the left and right sides, use `<BR CLEAR="ALL">` to force the next line of text to appear below the lowest image.

Sizing an Image with WIDTH and HEIGHT Attributes:

The `WIDTH` and `HEIGHT` attributes indicate the exact size of your image, in pixels.

For example:

```
<IMG SRC= "sbamm.jpg" WIDTH="109" "HEIGHT" >
```

One overwhelming advantage to adding the height and width to an `` tag is that when you do specify the image size for all of your images, browsers take a lot less time to render your page. That's because the browser can determine the layout of the page without having to retrieve each image separately to find out what size it is.

You can specify the image to have particular height and /or width, even if the original dimensions of the image don't match. Navigator and IE will then scale your image, stretching it accordingly.

For example, if your original image's dimensions are 50 by 50, you can specify an `` tag with a `WIDTH` of 200 and a `HEIGHT` of 25.

You can create interesting and artistic effects with this technique, but not every browser knows how to scale images. Most browsers do a poor job (leaving jagged images or strange distortions), so if you want to resize an image permanently, it's better to use an image tool for that purpose.

Note: To scale an image vertically, you can specify just the `HEIGHT` and leave the `WIDTH` automatic. Or you can scale the image horizontally by specifying the `WIDTH` and leaving the `HEIGHT` with its default value.

Setting an Image's Border Width:

By default, no border appears around an image unless that image is a link. However, you can specify a border for an image. If you use the **BORDER="1"** attribute in an tag, then a thin border will appear around the image. You can specify larger values for the Border attribute as well.

Adding White Space with HSPACE and VSPACE:

IE and Navigator do not place images right next to text. Instead, they put a small margin of a few pixels in between text and an image. You can control the amount of horizontal space with the HSPACE attribute and the amount of vertical space with the VSPACE attribute. For example

```
<IMG SRC= "sbamm.jpg" HSPACE="50" >
```

will add 50 pixels of white space around the image.

Take a look at the simple example:

```
<HTML>
<Head>
  <Title>Images</Title>
</Head>
<Body>
  <H1 Align="center">ADDING GRAPHICS </H1>
  <P>
```

Graphics, images, pictures, photographs-whatever you call them, a visual element makes your page more compelling and is the easiest way to give your page a unique look. Have a look at this picture.

```
<IMG SRC= "smlake06.jpg"   ALIGN="right" WIDTH="200" HEIGHT="200" ALT="smlake.jpg"
HSPACE="150" VSPACE="50">
<BR Clear="right">
```

Isn't it nice, interesting and beautiful, making your Web page more attractive.

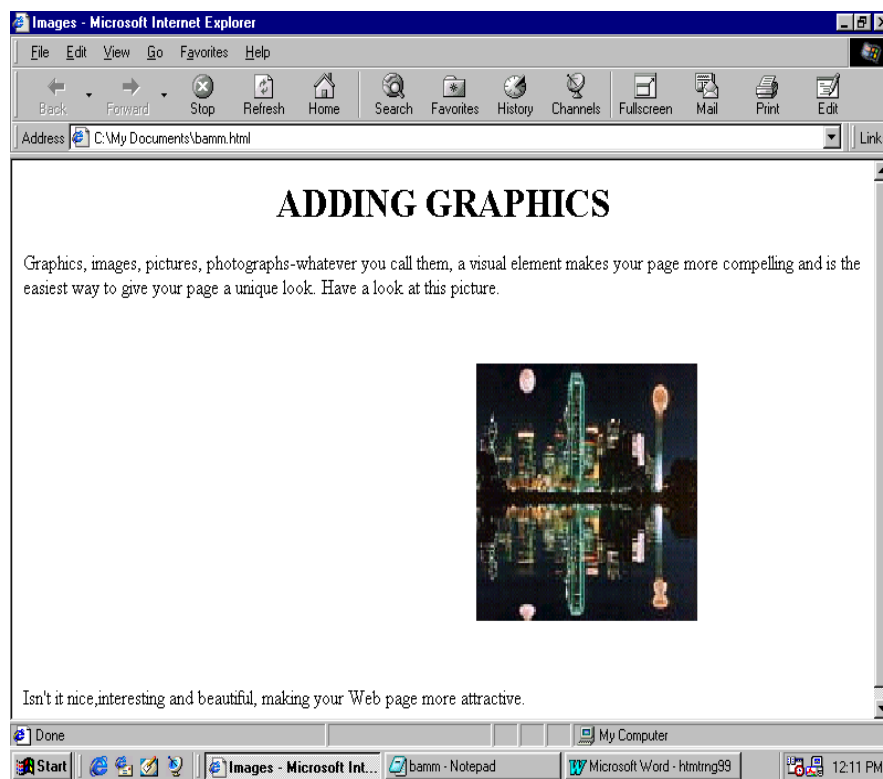
`</P>`

`</Body>`

`</HTML>`

The output of the above HTML code will be displayed as shown below:

External Images, Sounds, and Animations



You may want to have an image open as a separate document when a user activates a link on either a word or a smaller inline version of the image included in your document. This is called an external image, and it is useful if you do not wish to slow down the loading of the main document with large inline images.

To include a reference to an external image, enter:

```
<A HREF="MyImage.gif">link anchor</A>
```

You can also use a smaller image as a link to a larger image. Enter:

```
<A HREF="LargerImage.gif"><IMG SRC="SmallImage.gif"></A>
```

The reader sees the SmallImage.gif image and clicks on it to open the LargerImage.gif file.

Use the same syntax for links to external animations and sounds. The only difference is the file extension of the linked file.

For example,

```
<A HREF="AdamsRib.mov">link anchor</A>
```

specifies a link to a QuickTime movie.

Some common file types and their extensions are:

- plain text (.txt)
- HTML document (.html)
- GIF image (.gif)
- TIFF image (.tiff)
- X Bitmap image (. xbm)
- JPEG image (.jpg or .jpeg)
- PostScript file (.ps)
- AIFF sound file (.aiff)
- AU sound file (.au)
- WAV sound file (.wav)
- QuickTime movie (.mov)
- MPEG movie (.mpeg or .mpg)

Keep in mind your intended audience and their access to software. Most UNIX workstations, for instance, cannot view QuickTime movies.

Tables

<TABLE>, <TR>, <TD>

In the HTML document if you want to represent the information in a particular rows and columns format, you can use the table tags. Tables are very useful for presentation of tabular information as well as a boon to creative HTML authors who use the table tags to present their regular Web pages.

You use the `<TABLE>` tag to create the table and the `<TR>` tags to create the rows and `<TD>` to fill cells of the table with contents. The individual cells are where you place the information of the table, whether



it will be text or images. The following example shows a simple table consisting of two rows and two columns (four cells):

```
<TABLE>
```

```
<TR>
```

```
<TD>Apples
```

```
<TD>Celery
```

```
<TR>
```

```
<TD>Mango
```

```
<TD>Carrot
```

```
</TABLE>
```

➤ The output looks like:

- **WIDTH= n**

You can set the width of the table by using the `WIDTH=` attribute in the `TABLE` table.

- **ALIGN= align type**

You can align the content of each cell to the top, left, right, or bottom of a cell by using the `ALIGN=` and `VALIGN=` attributes in `TR` or `TD`.

- **BORDER=n**

To draw a border around the table and the individual cells, you use the `BORDER=` attribute in the `TABLE` tag. You specify the border width in pixels.

- **BGCOLOR=color and BORDERCOLOR=color**

You can add color to your tables by using the BGCOLOR= and BORDERCOLOR= attributes. These attributes are available in the TABLE, TR, and TD tags, so you can apply colors to all cells in a table, to cells in selected rows, or to individual cells. The BORDERCOLOR= attribute sets the color of the borders drawn around the table, row, or cell.

rules = none|groups|rows|cols|all

This attribute specifies which rules will appear between cells within a table. The rendering of rules is user agent dependent. Possible values:

- none: No rules. This is the default value.
- groups: Rules will appear between row groups and column groups
- rows: Rules will appear between rows only.
- cols: Rules will appear between columns only.
- all: Rules will appear between all rows and columns.

Cells may span several rows or columns. The number of rows or columns spanned by a cell is set by the rowspan and colspan attributes for the TH and TD elements.

In this table definition, we specify that the cell in row four, column two should span a total of three columns, including the current row.

```
<TABLE border="1">
<CAPTION>Cups of coffee consumed by each senator</CAPTION>
<TR><TH>Name<TH>Cups<TH>Type of Coffee<TH>Sugar?
<TR><TD>T. Sexton<TD>10<TD>Espresso<TD>No
<TR><TD>J. Dinnen<TD>5<TD>Decaf<TD>Yes
<TR><TD>A. Soria<TD colspan="3"><em>Not available</em>
</TABLE>
```

This table might be rendered on a tty device by a visual user agent as follows:

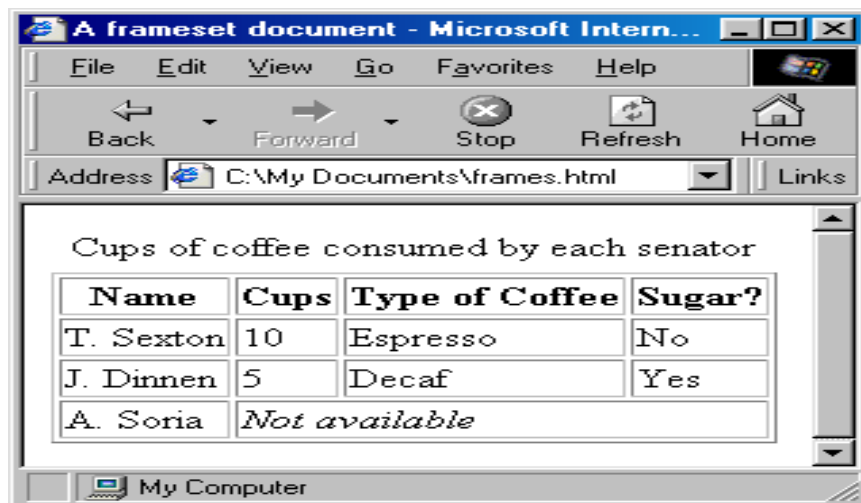
The next example illustrates (with the help of table borders) how cell definitions that span more than one row or column affect the definition of later cells. Consider the following table definition:

```
<TABLE border="1">
<TR><TD>1 <TD rowspan="2">2 <TD>3
<TR><TD>4 <TD>6
<TR><TD>7 <TD>8 <TD>9
</TABLE>
```

As cell "2" spans the first and second rows, the definition of the second row will take it into account. Thus, the second TD in row two actually defines the row's third cell.

Note: That if the TD defining cell "6" had been omitted, an extra empty cell would have been added by the user agent to complete the row.

Similarly, in the following table definition:



The screenshot shows a web browser window titled "A frameset document - Microsoft Intern...". The address bar shows "C:\My Documents\frames.html". The main content area displays a table with the title "Cups of coffee consumed by each senator". The table has four columns: Name, Cups, Type of Coffee, and Sugar?. The data rows are: T. Sexton (10 cups, Espresso, No sugar), J. Dinnen (5 cups, Decaf, Yes sugar), and A. Soria (Not available, which spans the last two columns).

Name	Cups	Type of Coffee	Sugar?
T. Sexton	10	Espresso	No
J. Dinnen	5	Decaf	Yes
A. Soria	Not available		

```
<TABLE border="1">
<TR><TD>1 <TD>2 <TD>3
<TR><TD colspan="2">4 <TD>6
<TR><TD>7 <TD>8 <TD>9
</TABLE>
```

cell "4" spans two columns, so the second TD in the row actually defines the third cell ("6"):

Defining overlapping cells is an error. User agents may vary in how they handle this error (e.g., rendering may vary).

The following example illustrates how one might create overlapping cells. In this table, cell "5" spans two rows and cell "7" spans two columns, so there is overlap in the cell between "7" and "9":

```
<TABLE border="1">
<TR><TD>1 <TD>2 <TD>3
<TR><TD>4 <TD rowspan="2">5 <TD>6
<TR><TD colspan="2">7 <TD>9
</TABLE>
```

<CAPTION>

You can add a caption, row and column headings, and a border to a table by using tags and attributes. The <CAPTION> tag is used to put the caption. By default, the caption is centered above the table, but you can use the ALIGN= attribute to place the caption at the top or bottom and at the left or right edge of the table.

<TH>

The <TH> tag adds headings to the rows and columns of a table. This tag is same as the <TD> tag, but it automatically emphasizes its text to distinguish it from text in other cells.

- **rowspan = number**



This attribute specifies the number of rows spanned by the current cell. The default value of this attribute is one ("1"). The value zero ("0") means that the cell spans all rows from the current row to the last row of the table.

- **colspan = number**

This attribute specifies the number of columns spanned by the current cell. The default value of this attribute is one ("1"). The value zero ("0") means that the cell spans all columns from the current column to the last column of the table.

<THEAD>, <TBODY> and <TFOOT>

If you use tables in the more traditional way (that is, presenting information in rows and columns), there are some additional tags and attributes that can make that job easier. The THEAD, TBODY and TFOOT tags let you divide your tables into parts: header, body, and footer. The COLGROUP and COL tags let you group columns within the table and globally apply properties, such as alignment, to the columns without having to specify these properties in each TD tag.

The following example creates a table that is the full width of the window and in which the contents of each cell are aligned at the top-left of the cell:

```
<HTML>
<HEAD>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0"
</TITLE>Tables</TITLE>
<BODY>
<TABLE WIDTH=100% BORDER=1 BORDERCOLOR=NAVY FRAME=BOX RULES=GROUPS >
<CAPTION>Fruits and Vegetables</CAPTION>
<THEAD>
<TR BGCOLOR=GRAY>
<TH>Fruits<TH>Vegetables
</THEAD>
<TBODY>
<COLGROUP ALIGN=CENTER>
<TR VALIGN=TOP ALIGN=LEFT>
<TD BGCOLOR=LIME>Apples</TD>
<TD BGCOLOR=AQUA>Celery</TD>
</TR>
```

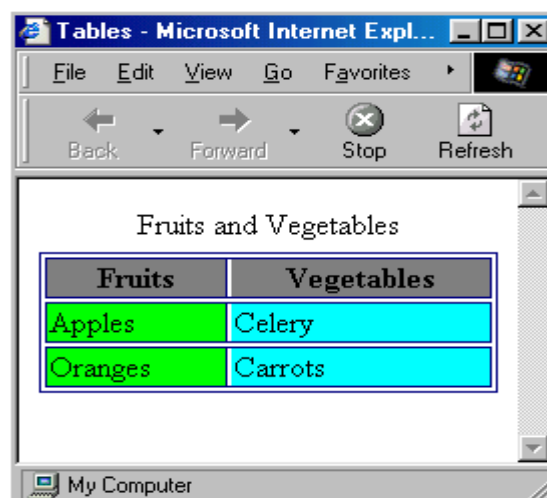


```

</TBODY>
<TBODY>
<TR VALIGN=TOP ALIGN=LEFT>
<TD BGCOLOR=LIME>Oranges</TD>
<TD BGCOLOR=AQUA>Carrots </TD>
</TR>
</TBODY>
</TABLE>
</BODY>
</HTML>

```

- The output of the above HTML code will be displayed as shown below.



Forms

You can design web pages that look like forms. Using forms, your server can interact with users to gather information. Users can browse through the forms, making selections and entering data as they move within the form. In a form, users can select from several choices, using a variety of selection methods, such as buttons, fill in the blanks, and selection lists. The way the users select any information depends on the form elements you use when you create the form. This section describes how to implement the following elements in a form:

Creating Forms

```
<FORM>
```

The `<FORM>` tag creates an HTML form, which lets users input text and make choices from elements such as checkboxes, radio buttons, and selection lists. A user fills out the form, and then submits the form by clicking a button. The way your browser or the server handles your form depends on the way you specify attributes to your form.

- `action = uri`

This attribute specifies a form-processing agent. For example, the value might be an HTTP URI (to submit the form to a program) or a `mailto` URI (to email the form).

- `method = get | post`

This attribute specifies which HTTP method will be used to submit the form data set. Possible (case-insensitive) values are "get" (the default) and "post".

- `get`: With the HTTP "get" method, the form data set is appended to the URI specified by the action attribute and this new URI is sent to the processing agent.

- `post`: With the HTTP "post" method, the form data set is included in the body of the form and sent to the processing agent.

The "get" method should be used when the form is idempotent (i.e., causes no side effects). Many database searches have no visible side effects and make ideal applications for the "get" method.

If the service associated with the processing of a form causes side effects (for example, if the form modifies a database or subscription to a service), the "post" method should be used.

`<INPUT >`

A text field lets the user enter a word, phrase, or series of numbers. Use the `<INPUT>` tag to place text input fields on an HTML form.

The following example creates a text element that is 25 characters long. The text field appears immediately to the right of the words "Last name:" The text field is blank when the form loads.

`<FORM>`

`Last name: <INPUT TYPE="text" NAME="last_name" SIZE=25>`

`</FORM>`

- The output looks like: Last name:

The other attributes of this tag are as follows:

Type = text| password| checkbox| radio| submit| reset| file| hidden| image| button

This attribute specifies the type of control to create. The default value for this attribute is "text".

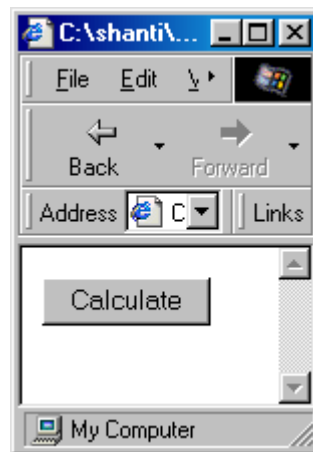
For example:

```
<FORM>
```

```
<INPUT TYPE="button" VALUE="Calculate" NAME="CalcButton">
```

```
</FORM>
```

- Will give an output:



When a user presses the reset button, all elements in the form are reset to their default values. One can use the INPUT tag to implement reset buttons on an HTML form.

When a user presses the submit button, the form is submitted to the server specified in the form's ACTION attribute. One can use the INPUT tag to implement submit buttons on an HTML form. A form can have multiple submit buttons, each with different NAME and VALUE attributes. When one of the submit buttons is pressed, the form sends the server the name & value pair of that button only (in addition to the name & value pairs for all other elements on the form). Some browsers do not support multiple submit

buttons on one form.

Clicking the submit button sends the form to the URL specified in the form's ACTION attribute. This action always loads a new page into the client; it might be the same as the current page, if the action specifies or is not specified.

A set of radio buttons lets the user choose one item from the set. Use the INPUT tag to implement radio buttons on an HTML form.

All radio buttons in a group have the same value for the NAME attribute. When a form is submitted to the server, the name & value pair for the radio button is sent only if the radio button is selected. For a group of radio buttons, only one name & value pair is sent to the server, because only one button in a group can be selected.

A checkbox is a toggle switch that lets the user set a value on or off. Use the INPUT tag to implement checkboxes on an HTML form. When a form is submitted to the server, the name & value pair for the checkbox is sent only if the checkbox is checked.

Multiple checkbox elements can have the same values for the NAME attribute if they have different values for the VALUE attribute. When the form is submitted, the browser sends the name & value pairs for all the checkboxes that are checked. This can yield several name & value pairs with the same name.

Image elements are graphics on an HTML form. Use the INPUT tag to implement image elements within a form. When a user clicks an image element, the form is submitted to the server specified in the form's ACTION attribute.

When the form is submitted, the coordinates where the user clicked are also submitted. The coordinates are measured in pixels from the upper left corner of the image (similar to the ISMAP attribute of the IMG tag). The coordinates are sent in two name & value pairs. Appending “x” or “y” to the element’s name creates the name for each pair. For example, if the image element is named Image1 and the user clicks the image at the x coordinate 28 and the y coordinate 37, the coordinates are sent to the server using the name & value pairs Image1.x=28 and Image1.y=37.

- **name = cdata**

This attribute assigns the control name.

- **value = cdata**

This attribute specifies the initial value of the control. It is optional except when the type attribute has the value "radio".

- **size = cdata**

This attribute tells the user agent the initial width of the control. The width is given in pixels except when type attribute has the value "text" or "password". In that case, its value refers to the (integer) number of characters.

- **maxlength = number**

When the type attribute has the value "text" or "password", this attribute specifies the maximum number of characters the user may enter. This number may exceed the specified size, in which case the user agent should offer a scrolling mechanism. The default value for this attribute is an unlimited number.

checked

When the type attribute has the value "radio" or "checkbox", this boolean attribute specifies that the button is on. User agents must ignore this attribute for other control types.

- **src = uri**

When the type attribute has the value "image", this attribute specifies the location of the image to be used to decorate the graphical submit button.

You can also use the INPUT tag to implement buttons on an HTML form.

<SELECT>

The SELECT element creates a menu. Each choice offered by the menu is represented by an OPTION element. A SELECT element must contain at least one OPTION element.

The attributes for the this tag are as follows:

- **name = cdata**

This attribute assigns the control name.

- size = number

If a SELECT element is presented as a scrolled list box, this attribute specifies the number of rows in the list that should be visible at the same time. Visual user agents are not required to present a SELECT element as a list box; they may use any other mechanism, such as a drop-down menu.

- multiple
- If set, this boolean attribute allows multiple selections. If not set, the SELECT element only permits single selections.

<OPTION >

When rendering a menu choice, user agents should use the value of the label attribute of the OPTION element as the choice. If this attribute is not specified, user agents should use the contents of the OPTION element.

- selected

When set, this boolean attribute specifies that this option is pre-selected.

- value = cdata

This attribute specifies the initial value of the control. If this attribute is not set, the initial value is set to the contents of the OPTION element.

- label = text

This attribute allows authors to specify a shorter label for an option than the content of the OPTION element. When specified, user agents should use the value of this attribute rather than the content of the OPTION element as the option label.

The use of the select and the option tag are explained well by the following example:

<HTML>

<FORM name="test" action="forms.html" method="post">

Name:

<INPUT type="text" id="name" value="Dr. Anurag Verma" size=25 >

*
*

<SELECT name="age" value="24">1,2,3,4,5,6,7,8,9,10

<OPTION>1</OPTION>

<OPTION>2</OPTION>

<OPTION selected>3</OPTION>

```
<OPTION>4</OPTION>
<OPTION>5</OPTION>
<OPTION>6</OPTION>
<OPTION>7</OPTION>
</SELECT>
<INPUT type="submit" name="submit" value="submit">
</FORM>
</HTML>
```

<TEXTAREA>

The TEXTAREA element creates a multi-line text input control. User agents should use the contents of this element as the initial value of the control and should render this text initially.

The attributes for the tag are as follows:

- name = cdata

This attribute assigns the control name.

- rows = number

This attribute specifies the number of visible text lines. Users should be able to enter more lines than this, so user agents should provide some means to scroll through the contents of the control when the contents extend beyond the visible area.

- cols = number

This attribute specifies the visible width in average character widths. Users should be able to enter longer lines than this, so user agents should provide some means to scroll through the contents of the control when the contents extend beyond the visible area. User agents may wrap visible text lines to keep long lines visible without the need for scrolling.

The following HTML example displays two text elements, a select element, and three radio buttons, all of which have default values. The form also has a reset button named "Defaults." If the user changes the value of any of the elements and then clicks the Defaults button, the original values are restored.

```
<HTML>
<FORM NAME="form1">
```

```

<BR>
<B>City: </B>
<INPUT TYPE="text" NAME="city" VALUE="Santa Cruz" SIZE="20">
<B>State: </B>
<INPUT TYPE="text" NAME="state" VALUE="CA" SIZE="2">
<P><SELECT NAME="colorChoice">
    <OPTION SELECTED> Blue
    <OPTION> Yellow
    <OPTION> Green
    <OPTION> Red
</SELECT>
<P><INPUT TYPE="radio" NAME="musicChoice" VALUE="soul-and-r&b"
    CHECKED> Soul and R&B
<BR><INPUT TYPE="radio" NAME="musicChoice" VALUE="jazz">
    Jazz
<BR><INPUT TYPE="radio" NAME="musicChoice" VALUE="classical">
    Classical
<CENTER><INPUT TYPE="image" SRC="alr_conn.gif"></CENTER>

<INPUT TYPE="checkbox" NAME="musicpref_rnb" CHECKED> R&B
<BR><INPUT TYPE="checkbox" NAME="musicpref_jazz" CHECKED> Jazz
<BR><INPUT TYPE="checkbox" NAME="musicpref_blues" CHECKED> Blues
<P><INPUT TYPE="reset" VALUE="Defaults" NAME="reset1">
<INPUT          TYPE="submit"          NAME="SubmitButton"          VALUE="Done">
</FORM>
</HTML>

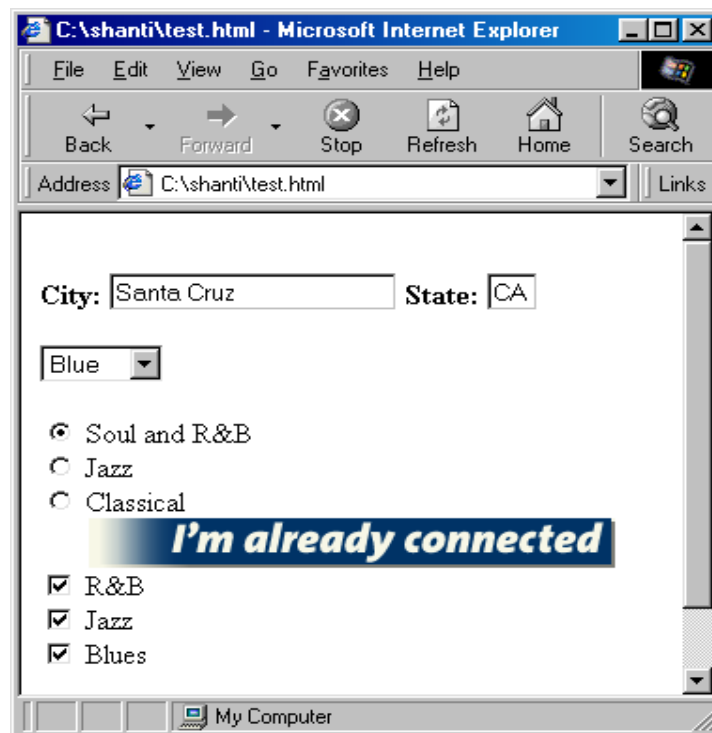
```

renders the Explorer to look like this

Note: You can also include a form in a document that contains a body part.

Hidden elements

Hidden elements are text elements that don't display on the form. Use the INPUT tag to implement hidden elements. A hidden element is used for passing information to the server when a form is submitted. A hidden element cannot be seen or modified by a user (other than by viewing the source of the HTML), but by using JavaScript, you can programmatically change its value. You can use hidden elements for client/server communication or to pass state information from one script or form to another.



When a form is submitted to the server, a hidden element's name & value pair is always sent.

The following example creates a form called LoginForm that contains text fields for user name and password, a submit button, and a cancel button. A hidden element, DefaultPass, stores the initial value of the password field.

```
<FORM NAME="LoginForm">
  <B>User name:</B>
  <INPUT TYPE="text" NAME="userName" SIZE="10">
  <P>
```

```
<B>Password:</B>
<INPUT TYPE="password" NAME="password" SIZE=12
VALUE="treasure">
<INPUT TYPE="hidden" NAME="DefaultPass" VALUE="treasure">

<P>
<INPUT TYPE="submit" VALUE="Log in">
<INPUT TYPE="button" VALUE="Cancel" onClick="window.close()">

</FORM>
```

Password elements

Password elements are text-input fields on an HTML form that conceal their value by displaying asterisks (*). When the user enters text into the field, asterisks (*) hide anything entered from view.

Even though passwords are masked onscreen, the password is sent to the server as straight text and is not encrypted when the form is submitted, unless the server itself uses an encryption method. Be cautious when using password fields because they could be intercepted and read by anyone. The following HTML example creates a form in which if the user enters a password containing more than 25 characters, the text scrolls to make room for the additional characters.

```
<FORM>
<B>User name:</B>
<INPUT TYPE="text" NAME="username" SIZE=10>
<B>Password:</B>
<INPUT TYPE="password" NAME="password" VALUE="" SIZE=25>

</FORM>
```

Sending a form data to a server

HTML forms collect data, but they do not usually process it. To process a form, you can submit the form to a program stored on a web server. The form and the server-side program should be designed together so that the program can process the form data being sent. When a form is submitted, each form element defined with INPUT, SELECT, or TEXTAREA tag is sent to the server in the format name & value; this is called a name & value pair. The name comes from the tag's NAME attribute, and the value is the value of the form element when submitted.

For example, the following form contains an INPUT tag that defines a text element called **LastName**.

`<FORM>`

`Last name:`

`<INPUT TYPE="text" NAME="LastName" VALUE="">`

`<INPUT TYPE="submit" NAME="SubmitButton" VALUE="Done">`

`</FORM>`

Suppose the user enters the value "ICAR-IASRI" for LastName and presses the Done button; the form is submitted and the name & value pair LastName=ICAR-IASRI is sent to the server.

Basics of Programming using Python

Madhu

ICAR-Indian Agricultural Statistics Research Institute, New Delhi - 110 012

madhu1@icar.gov.in

Python is a very popular general-purpose interpreted, interactive, object-oriented, and high-level programming language. Python is dynamically-typed and garbage-collected programming language. It was created by Guido van Rossum during 1985- 1990. Like Perl, Python source code is also available under the GNU General Public License (GPL).

Characteristics of Python

Following are important characteristics of **Python Programming** –

- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- It supports automatic garbage collection.

It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

Python Syntax

```
print('India is my country.')
```

Variables

```
x=2  
y="India"  
print(x)  
print(y)
```

Type Casting

```
x=str(5)  
y=int(5.0)
```

```
z=float(5)
print(x)
print(y)
print(z)
print(type(x))
print(type(y))
```

Single & Double Quotes

```
x="india"
y='india'
print(x)
print(y)
```

Multiline Strings

```
a = """hello,
good morning,
h r u,
all."""
print(a)
```

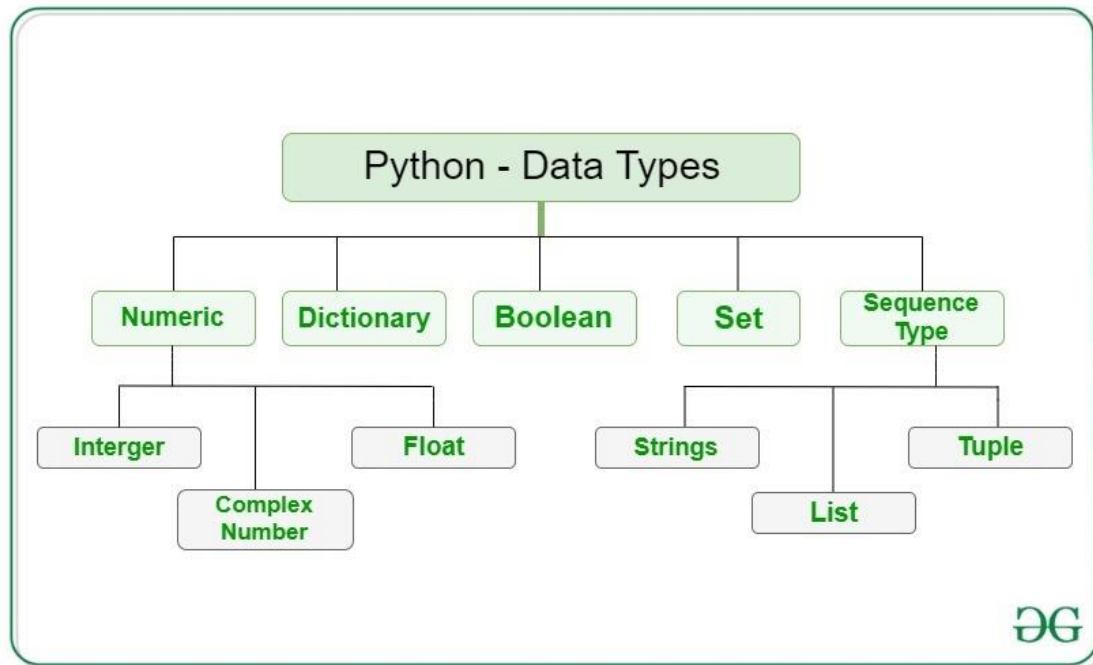
```
a = '''hello,
good morning,
h r u
all.'''
print(a)
```

Python Data Types

Data types are the classification or categorization of data items. It represents the kind of value that tells what operations can be performed on a particular data. Since everything is an object in Python programming, data types are actually classes and variables are instance (object) of these classes.

Following are the standard or built-in data type of Python:

- Numeric
- Sequence Type
- Boolean
- Set
- Dictionary



Numeric

In Python, numeric data type represent the data which has numeric value. Numeric value can be integer, floating number or even complex numbers. These values are defined as int, float and complex class in Python.

- **Integers** – This value is represented by int class. It contains positive or negative whole numbers (without fraction or decimal). In Python there is no limit to how long an integer value can be.
- **Float** – This value is represented by float class. It is a real number with floating point representation. It is specified by a decimal point. Optionally, the character e or E followed by a positive or negative integer may be appended to specify scientific notation.
- **Complex Numbers** – Complex number is represented by complex class. It is specified as (*real part*) + (*imaginary part*)j. For example – 2+3j

Note – type() function is used to determine the type of data type.

Float

```

x = 1.10
y = 1.0
z = -35.59
  
```

```
print(type(x))
print(type(y))
print(type(z))
```

int

```
x = 1
y = 3565
z = -3255522
```

```
print(type(x))
print(type(y))
print(type(z))
```

complex

```
x = 3+5j
y = 5j
z = -5j
```

```
print(type(x))
print(type(y))
print(type(z))
```

Sequence Type

In Python, sequence is the ordered collection of similar or different data types. Sequences allows to store multiple values in an organized and efficient fashion. There are several sequence types in Python –

- String
- List
- Tuple

String

In Python, Strings are arrays of bytes representing Unicode characters. A string is a collection of one or more characters put in a single quote, double-quote or triple quote. In python there is no character data type, a character is a string of length one. It is represented by str class.


Creating String

Strings in Python can be created using single quotes or double quotes or even triple quotes.

Accessing elements of String

In Python, individual characters of a String can be accessed by using the method of Indexing. Indexing allows negative address references to access characters from the back of the String, e.g. -1 refers to the last character, -2 refers to the second last character and so on.

G	E	E	K	S	F	O	R	G	E	E	K	S
0	1	2	3	4	5	6	7	8	9	10	11	12
-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1



```
print("Hello")
print('Hello')
```

```
a = "Hello"
print(a)
```

Strings are Arrays

```
a = "Hello, World!" # 0....n
print(a[1])
print(a[2])
print(a[7])
```

Looping Through a String

```
for x in "banana":
    print(x)
```

String Slicing

```
b = "Hello, World!"
print(b[2:5])
```

```
b = "Hello, World!"
print(b[:5])
```

```
b = "Hello, World!"
```



```
print(b[2:])

b = "Hello, World!"
print(b[-5:-2])
print(b[-1])
```

Strings Functions

```
a = "hello, World!"
print(a.upper())           #Converts a string into upper case
print(a.capitalize())      #Converts the first character to upper case
print(a.casefold())        #Converts string into lower case
print(a.split())           #Splits the string at the specified separator, and returns a
list
print(a.lower())           #Converts a string into lower case
print(a.strip())           # Returns a trimmed version of the string
                           #returns "Hello, World!"
print(a.replace("h", "J")) #Returns a string where a specified value is replaced
with a specified value
print(a.isdigit())         #Returns True if all characters in the string are digits
print(a.isupper())         #Returns True if all characters in the string are upper case
print(len(a))              #len() function returns the length of a string
```

String Concatenation

```
a = "Hello"
b = "World"
c = a + b
print(c)

a = "Hello"
b = "World"
c = a + " " + b
print(c)

"""we cannot combine strings and numbers"""

age = 36
txt = "My name is John, I am " + age
print(txt)
```

we can combine strings and numbers by using the **format() method!**

The `format()` method takes the passed arguments, formats them, and places them in the string where the placeholders `{ }`

```
age = 36
```

```
txt = "My name is John, and I am {}"  
print(txt.format(age))
```

The format() method takes unlimited number of arguments, and are placed into the respective placeholders:

You can use index numbers {0} to be sure the arguments are placed in the correct placeholders.

```
quantity = 3  
itemno = 567  
price = 49.95  
myorder = "I want {} pieces of item {} for {} dollars."  
print(myorder.format(quantity, itemno, price))
```

```
quantity = 3  
itemno = 567  
price = 49.95  
myorder = "I want to pay {2} dollars for {0} pieces of item {1}."  
print(myorder.format(quantity, itemno, price))
```

LIST [] O,C,DUPLICATE

TUPLES () O, NOT CHANGE,D

DICTIONARY KEY:VALUE

SET {} O,C,NOT DUPLICATE

List

Lists are just like the arrays, declared in other languages which is a ordered collection of data. It is very flexible as the items in a list do not need to be of the same type.

Creating List

Lists in Python can be created by just placing the sequence inside the square brackets[].

Accessing elements of List

In order to access the list items refer to the index number. Use the index operator [] to access an item in a list. In Python, negative sequence indexes represent positions from the end of the array. Instead of having

to compute the offset as in `List[len(List)-3]`, it is enough to just write `List[-3]`. Negative indexing means beginning from the end, -1 refers to the last item, -2 refers to the second-last item, etc

Lists are used to store multiple items in a single variable.

List items are ordered, changeable, and allow duplicate values. Lists are created using **square brackets**.

Ordered

When we say that lists are ordered, it means that the items have a defined order, and that order will not change.

If you add new items to a list, the new items will be placed at the end of the list.

Changeable

The list is changeable, meaning that we can change, add, and remove items in a list after it has been created.

Allow Duplicates

Since lists are indexed, lists can have items with the same value

```
thislist = ["apple", "banana", "cherry"]
print(thislist)
print(type(thislist))
```

```
thislist = ["apple", "banana", "cherry", "apple", "cherry"]
print(thislist)                                # Allow Duplicates
```

Length of a List

```
thislist = ["apple", "banana", "cherry"]
print(len(thislist))                          #len function for finding the number of values
in a tuple
```

```
list1 = ["apple", "banana", "cherry"]          #List items can be of any data type
```

```
list2 = [1, 5, 7, 9, 3]
list3 = [True, False, False]
print(type(list2))
print(type(list1))

list1 = ["abc", 34, True, 40, "male"]           #A list can contain different data types
print(type(list1))
```

Indexes of List items

```
thislist = ["apple", "banana", "cherry"]
print(thislist[1])
```

```
thislist = ["apple", "banana", "cherry"]
print(thislist[-1])
print(thislist[-2])
print(thislist[-3])
print(thislist[0])
```

```
thislist = ["apple", "banana", "cherry", "orange", "kiwi", "melon", "mango"]
print(thislist[2:5])
```

```
thislist = ["apple", "banana", "cherry", "orange", "kiwi", "melon", "mango"]
print(thislist[:4])
```

```
thislist = ["apple", "banana", "cherry", "orange", "kiwi", "melon", "mango"]
print(thislist[-4:-1])
```

```
thislist = ["apple", "banana", "cherry"]
if "apple" in thislist:           # true
    print("Yes, 'apple' is in the fruits list")
```

```
thislist = ["apple", "banana", "cherry"]
if "orange" in thislist:         #false
    print("no, 'orange' is not in the fruits list")    #not executed
```

Change Item Value

```
thislist = ["apple", "banana", "cherry", "orange", "kiwi", "mango"]
thislist[1:3] = ["blackcurrant", "watermelon"]           #Change the values
"banana" and "cherry" with the values "blackcurrant" and "watermelon"
print(thislist)
```

```
thislist = ["apple", "banana", "cherry"]
thislist[1:3] = ["watermelon"]
print(thislist)
```

Insert Items

To insert a new list item, without replacing any of the existing values, we can use the **insert() method**.

```
thislist = ["apple", "banana", "cherry"]
thislist.insert(2, "watermelon")           # insert() method inserts an item at the
specified index
```

```
thislist = ["apple", "banana", "cherry"]
thislist.append("orange")                  #append() method add
an item to the end of the list
print(thislist)
```

```
thislist = ["apple", "banana", "cherry"]
tropical = ["mango", "pineapple", "papaya"]
```

```
print(thislist+tropical)
```

```
thislist = ["apple", "banana", "cherry"]
thistuple = ("kiwi", "orange")
thislist.extend(thistuple)                 #extend() method does not have to append lists only, you
can add any iterable object (tuples, sets, dictionaries etc.).
print(thislist)
print(type(thistuple))
```

```
thislist = ["apple", "banana", "cherry"]
thislist.remove("banana")                  #remove() method removes the specified item
```

```
print(thislist)

thislist = ["apple", "banana", "cherry"]
thislist.pop(-1)          #pop() method removes the specified index
print(thislist)

thislist = ["apple", "banana", "cherry"]
thislist.pop()            #do not specify the index, the pop() method removes the last item
print(thislist)

thislist = ["apple", "banana", "cherry"]
del thislist[0]            #del keyword also removes the specified index
print(thislist)

thislist = ["apple", "banana", "cherry"]
del thislist
print(thislist)            #Delete the entire list

thislist = ["apple", "banana", "cherry"]
thislist.clear()           #list still remains, but it has no content
print(thislist)

thislist = ["apple", "banana", "cherry"]
for x in thislist:
    print(x)

thislist = ["apple", "banana", "cherry"]
for i in range(len(thislist)):
    print(thislist[i])

thislist = ["apple", "banana", "cherry"]
i = 0
while i < len(thislist):    #until true
    print(thislist[i])
    i = i + 1
```

Sorting List

```
thislist = ["orange", "mango", "apricot", "apple", "banana"]
```

```
thislist.sort()          #sort() method that will sort the list alphanumerically,  
ascending, by default  
print(thislist)  
  
thislist = [100, 50, 50, 82, 23]  
thislist.sort()  
print(thislist)  
  
thislist = [100, 50, 65, 82, 23]  
thislist.sort(reverse = False)      #To sort descending, use the keyword argument reverse  
= True  
print(thislist)
```

Copy a List

```
thislist = ["apple", "banana", "cherry"]  
mylist = thislist.copy()      #to make a copy, one way is to use the built-in List method  
copy()  
print(mylist)  
  
thislist = ["apple", "banana", "cherry"]  
mylist = list(thislist)      #make a copy is to use the built-in method list()  
print(mylist)
```

Join Two Lists

```
list1 = ["a", "b", "c"]  
list2 = [1, 2, 3]  
  
list3 = list1 + list2      #by using the + operator  
print(list3)  
  
list1 = ["a", "b" , "c"]  
list2 = [1, 2, 3]  
  
for x in list2:  
    list1.append(x)      #Another way to join two lists is by appending all the items
```

```
from list2 into list1, one by one

print(list1)
```

Tuple

Just like list, tuple is also an ordered collection of Python objects. The only difference between tuple and list is that tuples are immutable i.e. tuples cannot be modified after it is created. It is represented by tuple class.

Creating Tuple

In Python, tuples are created by placing a sequence of values separated by ‘comma’ with or without the use of parentheses for grouping of the data sequence. Tuples can contain any number of elements and of any datatype (like strings, integers, list, etc.).

Tuples are used to store multiple items in a single variable.

A tuple is a collection which is ordered and unchangeable.

Tuples are written with **round brackets**. ()

Ordered

When we say that tuples are ordered, it means that the items have a defined order, and that order will not change.

Unchangeable

Tuples are unchangeable, meaning that we cannot change, add or remove items after the tuple has been created.

Allow Duplicates

Since tuples are indexed, they can have items with the same value:


```
thistuple = ("apple", "banana", "cherry")
print(thistuple)

thistuple = ("apple", "banana", "cherry", "apple", "cherry")
print(thistuple)                                #allow duplicates

thistuple = ("apple", "banana", "cherry")
print(len(thistuple))                          #len function for finding the number of values in a tuple

thistuple = ("apple",)                         #use comma (,) if tuple having single value otherwise it is
considered as string
print(type(thistuple))
print(len(thistuple))

thistuple = ("apple")                          #NOT a tuple
print(type(thistuple))

tuple1 = ("apple", "banana", "cherry")
tuple2 = (1, 5, 7, 9, 3)
print(type(tuple1))
print(type(tuple2))

tuple1 = ("abc", 34, True, 40, "male")
print(type(tuple1))

thistuple = ("apple", "banana", "cherry", "orange", "kiwi", "melon", "mango")
print(thistuple[2:5])                          #access the elements of a tuple
```

Change Tuple Values

Once a tuple is created, you cannot change its values. Tuples are unchangeable, or immutable. You can convert the tuple into a list, change the list, and convert the list back into a tuple

```
x = ("apple", "banana", "cherry")  #can not change the tuple
y = list(x)                        #convert tuple into list
y[1] = "kiwi"                      # change list
x = tuple(y)                       #convert list into tuple
```

```
print(x)

thistuple = ("apple", "banana", "cherry")           #Since tuples are immutable, they
do not have a build-in append() method
y = list(thistuple)
y.append("orange")                                   # list have append method
thistuple = tuple(y)
print(thistuple)

thistuple = ("apple", "banana", "cherry")
y = ("orange","mango")
thistuple += y          # allowed to add tuples to tuples

print(thistuple)

thistuple = ("apple", "banana", "cherry")           # they do not have a build-in remove()
method
y = list(thistuple)
y.remove("apple")                                     # list have remove method
thistuple = tuple(y)
print(thistuple)

thistuple = ("apple", "banana", "cherry")
del thistuple
print(thistuple) #this will raise an error because the tuple no longer exists
```

Packing & Unpacking Tuples

When we create a tuple, we normally assign values to it. This is called "**packing**" a tuple.

In Python, we are also allowed to extract the values back into variables. This is called "**unpacking**".

```
a = ("Delhi", 5000, "Agriculture")    #PACKS values into variable a

(City, student, type_ofcollege) = a    #UNPACKS values of variable a

print(City,student,type_ofcollege)
```

If the number of variables is less than the number of values, you can add an * to the variable name and the values will be assigned to the variable as a list.

```
fruits = ("apple", "banana", "cherry", "strawberry", "raspberry")
```

```
(green, yellow, *red) = fruits
```

```
print(green)
print(yellow)
print(red)
```

If the asterisk is added to another variable name than the last, Python will assign values to the variable until the number of values left matches the number of variables left.

```
fruits = ("apple", "mango", "papaya", "pineapple", "cherry")
```

```
(green, *tropic, red) = fruits
```

```
print(green)
print(tropic)
print(red)
```

Join Tuples

```
tuple1 = ("a", "b" , "c")
tuple2 = (1, 2, 3)
```

```
tuple3 = tuple1 + tuple2
print(tuple3)
```

```
fruits = ("apple", "banana", "cherry")
mytuple = fruits * 2
```

```
print(mytuple)
```

Boolean

Data type with one of the two built-in values, True or False. Boolean objects that are equal to True are *truthy* (true), and those equal to False are *falsy* (false). But non-Boolean objects can be evaluated in Boolean context as well and determined to be true or false. It is denoted by the class `bool`.

Note – True and False with capital ‘T’ and ‘F’ are valid booleans otherwise python will throw an error. Booleans represent one of two values: True or False.

```
print(10 > 9)
print(10 == 9)
print(10 < 9)

print(bool("Hello"))          #Almost any value is evaluated to True if it has some sort of
content.
print(bool(15))               #Any string is True, except empty strings.
print(bool(0))                # Any number is True, except 0.
print(bool(""))
print(bool(()))
print(bool([]))
print(bool({}))               # Any list, tuple, set, and dictionary are True, except empty
ones.
print(bool(False))
print(bool(None))
```

Set

In Python, Set is an unordered collection of data type that is iterable, mutable and has no duplicate elements. The order of elements in a set is undefined though it may consist of various elements.

Creating Sets

Sets can be created by using the built-in `set()` function with an iterable object or a sequence by placing the sequence inside curly braces, separated by ‘comma’. Type of elements in a set need not be the same, various mixed-up data type values can also be passed to the set.

Accessing elements of Sets

Set items cannot be accessed by referring to an index, since sets are unordered the items has no index. But you can loop through the set items using a for loop, or ask if a specified value is present in a set, by

using the `in` keyword.

Sets are used to store multiple items in a single variable.

Set Items

Set items are unordered, unchangeable, and do not allow duplicate values.

Unordered

Unordered means that the items in a set do not have a defined order.

Set items can appear in a different order every time you use them, and cannot be referred to by index or key.

Unchangeable

Set items are unchangeable, meaning that we cannot change the items after the set has been created.

Duplicates Not Allowed

Sets cannot have two items with the same value.

Sets are written with **curly brackets**.

```
thisset = {"apple", "banana", "cherry"}  
print(thisset)
```

```
thisset = {"apple", "banana", "cherry", "apple"}  
print(thisset)
```

```
thisset = {"apple", "banana", "cherry"}  
print(len(thisset))           # len() function used to find length
```

```
set1 = {"abc", 34, True, 40, "male"}  
print(set1)                   # A set with strings, integers and boolean values  
print(type(set1))              # type function return data type
```

```
thisset = {"apple", "banana", "cherry"}  
for x in thisset:  
    print(x)
```

```
thisset = {"apple", "banana", "cherry"}  
print("banana" in thisset)  
print("kiwi" in thisset)
```

Add an Item

```
thisset = {"apple", "banana", "cherry"}
thisset.add("orange")          # to add one item to a set use the add() method.
print(thisset)
```

```
thisset = {"apple", "banana", "cherry"}
tropical = {"pineapple", "mango", "papaya"}
thisset.update(tropical)       #To add items from another set into the
current set, use the update() method
print(thisset)
```

```
thisset = {"apple", "banana", "cherry"}
mylist = ["kiwi", "orange"]
thisset.update(mylist)         #update() method does not have to be a set only, it
can be any iterable object (tuples, lists, dictionaries etc.)
print(thisset)
```

Remove Item

```
thisset = {"apple", "banana", "cherry"}
thisset.remove("apple")
thisset.remove("kiwi")        #To remove an item in a set, use the remove(), or the
discard() method.
print(thisset)                #If the item to remove does not exist, remove() will
raise an error
```

```
thisset = {"apple", "banana", "cherry"}
thisset.discard("kiwi")       #If the item to remove does not exist, discard()
will NOT raise an error.
print(thisset)
```

```
thisset = {"apple", "banana", "cherry"}
x = thisset.pop()             # pop() method to remove an item, but this method will
remove the last item
print(x)                      #Remember that sets are unordered, so you will not know
what item that gets removed. The return value of the pop() method is the removed item.
print(thisset)
```

```
thisset = {"apple", "banana", "cherry"}
thisset.clear()               # clear() method empties the set
print(thisset)
```

```
thisset = {"apple", "banana", "cherry"}
del thisset                   # del keyword will delete the set completely
print(thisset)
```

Join Two Sets

```
set1 = {"a", "b" , "c"}
set2 = {1, 2, 3}

set3 = set1.union(set2)      # union() method returns a new set with all items from both
sets
print(set3)

set1 = {"a", "b" ,"b" ,"c"}
set2 = {1, 2, 3}

set1.update(set2)           #update() method inserts the items of set2 into set1.
                           #Both union() and update() will exclude any duplicate items
print(set1)
```

Both union() and update() will exclude any duplicate items.

```
x = {"apple", "banana", "cherry"}
y = {"google", "microsoft", "apple"}

x.intersection_update(y)    #intersection_update() method will keep only the items
that are present in both sets.

print(x)

x = {"apple", "banana", "cherry"}
y = {"google", "microsoft", "apple"}

z = x.intersection(y)      #intersection() method will return a new set, that only contains
                           # the items that are present in both sets.

print(z)

x = {"apple", "banana", "cherry"}
y = {"google", "microsoft", "apple"}

x.symmetric_difference_update(y)  #symmetric_difference_update() method will keep only
the elements that are
                                #NOT present in both sets.

print(x)

x = {"apple", "banana", "cherry"}
y = {"google", "microsoft", "apple"}

z = x.symmetric_difference(y)    # symmetric_difference() method will return a new
set,
                                # that contains only the elements that are NOT
```

present in both sets.

```
print(z)
```

Dictionary

Dictionary in Python is an unordered collection of data values, used to store data values like a map, which unlike other Data Types that hold only single value as an element, Dictionary holds key:value pair. Key-value is provided in the dictionary to make it more optimized. Each key-value pair in a Dictionary is separated by a colon :, whereas each key is separated by a ‘comma’.

Creating Dictionary

In Python, a Dictionary can be created by placing a sequence of elements within curly {} braces, separated by ‘comma’. Values in a dictionary can be of any datatype and can be duplicated, whereas keys can’t be repeated and must be immutable. Dictionary can also be created by the built-in function dict(). An empty dictionary can be created by just placing it to curly braces {}.

Note – Dictionary keys are case sensitive, same name but different cases of Key will be treated distinctly.

Accessing elements of Dictionary

In order to access the items of a dictionary refer to its key name. Key can be used inside square brackets. There is also a method called get() that will also help in accessing the element from a dictionary.

Dictionaries are used to store data values in **key:value pairs**.

A dictionary is a collection which is ordered, changeable and do not allow duplicates.

Ordered

When we say that dictionaries are ordered, it means that the items have a defined order, and that order will not change.

Changeable

Dictionaries are changeable, meaning that we can change, add or remove items after the dictionary has been created.

Duplicates Not Allowed

Dictionaries cannot have two items with the same key.


```
thisdict = {  
    "brand": "Ford",  
  
    "model": "Mustang",  
    "year": 1964  
}  
print(thisdict)
```

Dictionary items are presented in key:value pairs, and can be referred to by using the key name.

```
thisdict = {  
    "brand": "Ford",  
    "model": "Mustang",  
    "year": 1964  
}  
print(thisdict["brand"])  
print(len(thisdict))          # len () function returns length  
print(type(thisdict))        # type() function show datatype
```

```
thisdict = {  
    "brand": "Ford",  
    "model": "Mustang",  
    "year": 1964,  
    "year": 2020              #Dictionaries cannot have two items with the same key.  
                              # Duplicate values will overwrite existing values  
}  
print(thisdict)
```

Accessing Items

```
thisdict = {  
    "brand": "Ford",  
    "model": "Mustang",  
    "year": 1964  
}  
x = thisdict["model"]
```

```
y = thisdict.get("model")      #Get the value of the "model" key
z = thisdict.keys()           #keys() method will return a list of all the keys in the
dictionary
b = thisdict.values()         #values() method will return a list of all the values in the
dictionary
print(x)
print(y)
print(z)
print(b)
```

Change Dictionary Items

```
thisdict = {
    "brand": "Ford",
    "model": "Mustang",
    "year": 1964
}
thisdict["year"] = 2018
print(thisdict)

thisdict = {
    "brand": "Ford",
    "model": "Mustang",
    "year": 1964
}
thisdict.update({"year": 2020}) #update() method will update the dictionary with the items
from the given argument.
print(thisdict)
```

Adding Items

```
thisdict = {
    "brand": "Ford",
    "model": "Mustang",
    "year": 1964
}
thisdict["color"] = "red"
```

```
print(thisdict)

thisdict = {
    "brand": "Ford",
    "model": "Mustang",
    "year": 1964
}
thisdict.update({"color": "red"})      #update() method will update the dictionary with
the items from a given argument
print(thisdict)
```

Removing Items

```
thisdict = {
    "brand": "Ford",
    "model": "Mustang",
    "year": 1964
}
thisdict.pop("model")      #pop() method removes the item with the specified key name
print(thisdict)
```

```
thisdict = {
    "brand": "Ford",
    "model": "Mustang",
    "year": 1964
}
thisdict.popitem()         #popitem() method removes the last inserted item
print(thisdict)
```

```
thisdict = {
    "brand": "Ford",
    "model": "Mustang",
    "year": 1964
}
del thisdict["model"]      #del keyword removes the item with the specified key name
print(thisdict)
```

```
thisdict = {  
    "brand": "Ford",  
    "model": "Mustang",  
    "year": 1964  
}  
del thisdict          #del keyword can also delete the dictionary completely  
print(thisdict)
```

```
thisdict = {  
    "brand": "Ford",  
    "model": "Mustang",  
    "year": 1964  
}  
thisdict.clear()      #clear() method empties the dictionary  
print(thisdict)
```

Loop Through a Dictionary

```
thisdict = {  
    "brand": "Ford",  
    "model": "Mustang",  
    "year": 1964  
}  
for x in thisdict:    #Print all key names in the dictionary, one by one  
    print(x)  
for y in thisdict:  
    print(thisdict[y]) #Print all values in the dictionary, one by one  
for x in thisdict.values(): #values() method to return values of a dictionary  
    print(x)  
for x in thisdict.keys():  
    print(x)          #keys() method to return the keys of a dictionary  
for x, y in thisdict.items(): #items methods to return values for both key and values  
    print(x, y)
```

Copy a Dictionary

```
thisdict = {
```

```
"brand": "Ford",
"model": "Mustang",
"year": 1964
}
mydict = thisdict.copy()          # Make a copy of a dictionary with the copy() method
print(mydict)
print(thisdict)
mydict = dict(thisdict)          # to make a copy is to use the built-in function dict()
print(mydict)
```

Nested Dictionaries

A dictionary can contain dictionaries, this is called nested dictionaries.

```
myfamily = {
  "child1" : {
    "name" : "Emil",
    "year" : 2004
  },
  "child2" : {
    "name" : "Tobias",
    "year" : 2007
  },
  "child3" : {
    "name" : "Linus",
    "year" : 2011
  }
}
print(myfamily)
```

```
child1 = {
  "name" : "Emil",
  "year" : 2004
}
child2 = {
  "name" : "Tobias",
  "year" : 2007
}
```

```
}  
child3 = {  
    "name" : "Linus",  
    "year" : 2011  
}  
  
myfamily = {  
    "child1" : child1,  
    "child2" : child2,  
    "child3" : child3  
}  
print(myfamily)
```

Python Arrays

Array in Python can be created by importing array module. `array(data_type, value_list)` is used to create an array with data type and value list specified in its arguments.

import array as arr

```
# creating an array with integer type  
a = arr.array('i', [1, 2, 3])  
  
# printing original array  
print ("The new created array is : ", end = " ")  
for i in range (0, 3):  
    print (a[i], end = " ")  
print()
```

Accessing Python Array Elements

```
import array as arr  
a = arr.array('i', [2, 4, 6, 8])  
  
print("First element:", a[0])  
print("Second element:", a[1])
```

```
print("Last element:", a[-1])
```

Slicing Python Arrays

```
import array as arr
```

```
numbers_list = [2, 5, 62, 5, 42, 52, 48, 5]  
numbers_array = arr.array('i', numbers_list)
```

```
print(numbers_array[2:5]) # 3rd to 5th  
print(numbers_array[:-5]) # beginning to 4th  
print(numbers_array[5:]) # 6th to end  
print(numbers_array[:]) # beginning to end
```

Changing and Adding Elements

```
import array as arr
```

```
numbers = arr.array('i', [1, 2, 3, 5, 7, 10])
```

```
# changing first element  
numbers[0] = 0  
print(numbers) # Output: array('i', [0, 2, 3, 5, 7, 10])
```

```
# changing 3rd to 5th element  
numbers[2:5] = arr.array('i', [4, 6, 8])  
print(numbers) # Output: array('i', [0, 2, 4, 6, 8, 10])
```

```
import array as arr
```

```
numbers = arr.array('i', [1, 2, 3])
```

```
numbers.append(4)  
print(numbers) # add one item to the array using the append() method
```

```
numbers.extend([5, 6, 7])  
print(numbers) # add several items using the extend() method
```

```
import array as arr

odd = arr.array('i', [1, 3, 5])
even = arr.array('i', [2, 4, 6])

numbers = arr.array('i')    # concatenate two arrays using + operator
numbers = odd + even

print(numbers)

import array as arr

number = arr.array('i', [1, 2, 3, 3, 4])

del number[2]    # removing third element
print(number)    # Output: array('i', [1, 2, 3, 4])

del number    # deleting entire array
print(number)    # Error: array is not defined

import array as arr

numbers = arr.array('i', [10, 11, 12, 12, 13])

numbers.remove(12)    # remove() method to remove the given item
print(numbers)
print(numbers.pop(2))    # pop() method to remove an item at the given index
print(numbers)
```

Python Operators

Operators are special symbols in Python that carry out arithmetic or logical computation. The value that the operator operates on is called the operand.

Arithmetic operators

Arithmetic operators are used to perform mathematical operations like addition, subtraction, multiplication, etc.

+ Add two operands or unary plus $x + y + 2$:

- Subtract right operand from the left or unary minus $x - y - 2$

* Multiply two operands $x * y$

/ Divide left operand by the right one (always results into float) x / y

% Modulus - remainder of the division of left operand by the right $x \% y$ (remainder of x/y)

// Floor division - division that results into whole number adjusted to the left in the number line $x // y$

** Exponent - left operand raised to the power of right $x ** y$ (x to the power y)

"""

$x = 3$

$y = 2$

`print('x + y =', x+y)`

`print('x - y =', x-y)`

`print('x * y =', x*y)`

`print('x / y =', x/y)`

`print('x // y =', x//y)`

`print('x ** y =', x**y)`

Comparison operators

Comparison operators are used to compare values. It returns either True or False according to the condition.

> Greater than - True if left operand is greater than the right $x > y$

< Less than - True if left operand is less than the right $x < y$

== Equal to - True if both operands are equal $x == y$

!= Not equal to - True if operands are not equal $x != y$

>= Greater than or equal to - True if left operand is greater than or equal to the right $x >= y$

<= Less than or equal to - True if left operand is less than or equal to the right $x <= y$

"""

```
x = 5
y = 10

print('x > y is',x>y)
print('x < y is',x<y)
print('x == y is',x==y)
print('x != y is',x!=y)
print('x >= y is',x>=y)
print('x <= y is',x<=y)
```

Logical operators

Logical operators are the and, or, not operators.

and True if both the operands are true x and y

or True if either of the operands is true x or y

not True if operand is false (complements the operand) not x

"""

```
x = True
y = False

print('x and y is',x and y)    #true true

print('x or y is',x or y)        #either true

print('not x is',not x)
```

Assignment operators

Assignment operators are used in Python to assign values to variables.

a = 5 is a simple assignment operator that assigns the value 5 on the right to the variable a on the left.

```
**=**        x = 5        x = 5
```

```
**+=**    x += 5        x = x + 5
```

```
**-=**    x -= 5        x = x - 5
```

```
*=    x *= 5        x = x * 5
```

```
**/=**    x /= 5        x = x / 5
```

```
a = 21
```

```
b = 10
```

```
c = 0
```

```
c = a + b
```

```
print ("Value of c is ", c)
```

```
c += a
```

```
print ("Value of c is ", c)
```

```
c *= a
```

```
print ("Value of c is ", c)
```

```
c /= a
```

```
print ("Value of c is ", c)
```

```
c = 2
```

```
c %= a
```

```
print ("Value of c is ", c)
```

```
c **= a
```

```
print ("Value of c is ", c)
```

```
c //= a
```

```
print ("Value of c is ", c)
```

Bitwise Operators

Bitwise operators are used to compare (binary) numbers:

& AND Sets each bit to 1 if both bits are 1

| OR Sets each bit to 1 if one of two bits is 1

^ XOR Sets each bit to 1 if only one of two bits is 1

~ NOT Inverts all the bits

```
a = 10      #1010    0101
b = 4       #0100
```

```
# Print bitwise AND operation
print("a & b =", a & b)      #0000
```

```
# Print bitwise OR operation
print("a | b =", a | b)     #1110
```

```
# Print bitwise NOT operation
print("~a =", ~a)           # ~a = ~1010
                             #   = -(1010 + 1)
                             #   = -(1011)
                             #   = -11 (Decimal)
```

```
# print bitwise XOR operation
print("a ^ b =", a ^ b)     # Returns 1 if one of the bits is 1 and the other is 0 else
returns false.
```

Shift Operators

These operators are used to shift the bits of a number left or right thereby multiplying or dividing the number by two respectively.

Bitwise right shift: Shifts the bits of the number to the right and fills 0 on voids left(fills 1 in the case of a negative number) as a result.

Bitwise left shift: Shifts the bits of the number to the left and fills 0 on voids right as a result.

```
a = 10
b = -10
```

```
# print bitwise right shift operator
print("a >> 1 =", a >> 1)
print("b >> 1 =", b >> 1)
```

```
a = 5
b = -10
```

```
# print bitwise left shift operator
print("a << 1 =", a << 1)
print("b << 1 =", b << 1)
```

7) Identity operators

is and is not are the identity operators in Python. They are used to check if two values (or variables) are located on the same part of the memory.

```
x1 = 5
y1 = 5
x2 = 'Hello'
y2 = 'Hello'
x3 = [1,2,3]
y3 = [1,2,3]
print(x1 is not y1)      # Output: False

print(x2 is y2)          # Output: True

print(x3 is y3)          # Output: False
```

Membership operators

in and not in are the membership operators in Python. They are used to test whether a value or variable is found in a sequence (string, list, tuple, set and dictionary).

```
x = 'Hello world'
y = {1:'a',2:'b'}

print('H' in x)          # Output: True
```

```
print('hello' not in x)    # Output: True
```

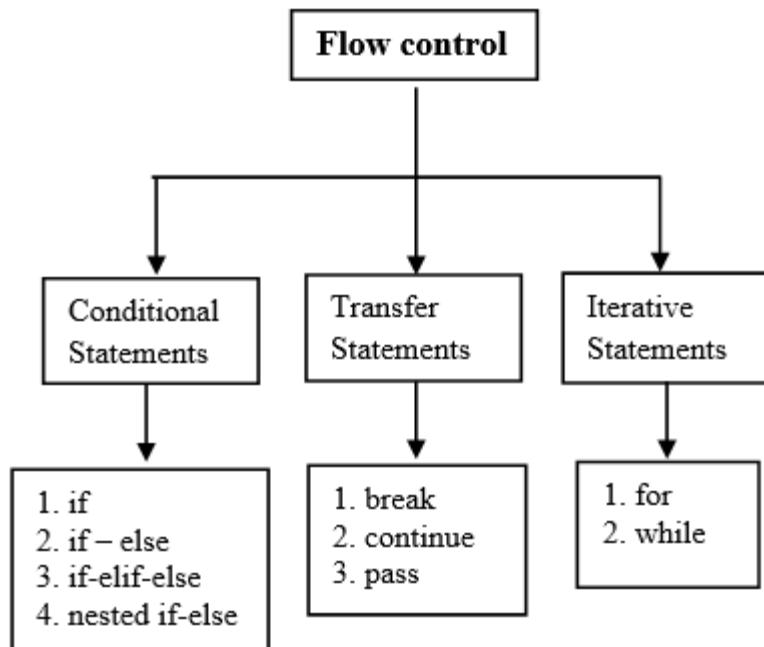
```
print(1 in y)              # Output: True
```

```
print('a' in y)            # Output: False
```

Control Flow Statements

The flow control statements are divided into three categories

1. Conditional statements
2. Iterative statements.
3. Transfer statements



Python If ... Else

```
a = 33
b = 200
if b > a:
    print("b is greater than a")
```

```
a = 33
b = 200
if b > a:
    print("b is greater than a")
```

Elif

```
a = 33
b = 33
if b > a:
    print("b is greater than a")
elif a == b:
    print("a and b are equal")
```

Else

```
a = 200
b = 33
if b > a:
    print("b is greater than a")
elif a == b:
    print("a and b are equal")
else:
    print("a is greater than b")
```

Short Hand If

```
if a > b: print("a is greater than b")
```

Short Hand If ... Else

```
a = 2
b = 330
print("A") if a > b else print("B")
```

Nested If

```
x = 41

if x > 10:
    print("Above ten,")
    if x > 20:
        print("and also above 20!")
    else:
        print("but not above 20.")
```

The pass Statement

```
a = 33
b = 200

if b > a:
    pass
```

while Loop

```
i = 1
while i < 6:
    print(i)
    i += 1
```

break Statement

```
i = 1
while i < 6:
    print(i)
    if i == 3:
        break
    i += 1
```

continue Statement

```
i = 0      #
while i < 6:
    i += 1
    if i == 3:
        continue
    print(i)
```

For Loop

A for loop is used for iterating over a sequence (that is either a list, a tuple, a dictionary, a set, or a string).

```
fruits = ["apple", "banana", "cherry"]
for x in fruits:
    print(x)
```

Looping Through a String


```
for x in "banana":  
    print(x)
```

range() Function

To loop through a set of code a specified number of times, we can use the range() function,

The range() function returns a sequence of numbers, starting from 0 by default, and increments by 1 (by default), and ends at a specified number.

```
for x in range(6):  
    print(x)
```

```
for x in range(2, 6):  
    print(x)                # not including 6
```

```
for x in range(2, 30, 3):    #third parameter is the increment value  
    print(x)
```

```
for x in range(6):  
    print(x)  
else:                #else keyword in a for loop specifies a block of code to be executed  
                    when the loop is finished  
    print("Finally finished!")
```

```
for x in range(6):  
    if x == 3: break    #else block will NOT be executed if the loop is stopped  
                    by a break statement  
    print(x)  
else:  
    print("Finally finished!")
```

Nested Loops

```
adj = ["red", "big", "tasty"]
```

```
fruits = ["apple", "banana", "cherry"]

for x in adj:
    for y in fruits:
        print(x, y)
```

Functions

You use functions in programming to bundle a set of instructions that you want to use repeatedly. That means that a function is a piece of code written to carry out a specified task.

There are three types of functions in Python:

- **User-Defined Functions (UDFs)**, which are functions that users create to help them out.
- **Anonymous functions**, which are also called ****lambda functions**** because they are not declared with the standard `def` keyword.
- **Built-in functions**, such as `help()` to ask for help, `min()` to get the minimum value, `print()` to print an object to the terminal.

Creating a Function

```
def my_function():
    print("Hello from a function")
```

```
"""**Calling** **a** **Function** """
```

```
def my_function():
    print("Hello from a function")
```

```
my_function()
```

```
def my_function(fname):
    print(fname + " Refsnes")
```

#A parameter is the variable listed inside the parentheses in the function definition.

```
my_function("Emil")
my_function("Tobias")
```

#An argument is the value that is sent to the function when it is called.

```
my_function("Linus")
```

Number of Arguments

```
def my_function(fname, lname):  
    print(fname + " " + lname)
```

```
my_function("Emil", "Refsnes")
```

```
def my_function(*kids):  
    #do not know how many arguments that will be  
    #passed into your function, add a * before the parameter name in the function definition  
    print("The youngest child is " + kids[2])
```

```
my_function("Emil", "Tobias", "Linus")
```

```
def my_function(child3, child2, child1):  
    print("The youngest child is " + child3)
```

```
my_function(child1 = "Emil", child2 = "Tobias", child3 = "Linus")  
#send arguments  
with the key = value syntax
```

```
def my_function(**kid):  
    #number of keyword arguments is unknown, add a  
    #double ** before the parameter name  
    print("His last name is " + kid["lname"])
```

```
my_function(fname = "Tobias", lname = "Refsnes")
```

Default Parameter Value

```
def my_function(country = "Norway"):  
    print("I am from " + country)
```

```
my_function("Sweden")
```

```
my_function("India")
```

```
my_function()  
#If we call the function without argument, it uses the  
#default value
```

```
my_function("Brazil")
```

Passing a List as an Argument

```
def my_function(food):
    for x in food:
        print(x)

fruits = ["apple", "banana", "cherry"]

my_function(fruits)
```

Return Values

```
def my_function(x):
    return 5 * x

print(my_function(3))
print(my_function(5))
print(my_function(9))
```

Python Lambda

A lambda function is a small anonymous function. A lambda function can take any number of arguments, but can only have one expression.

```
x = lambda a : a + 10
print(x(5))

Max = lambda a, b : a if(a > b) else b           # Example of lambda function using if-else

print(Max(1, 2))
```

Difference Between Lambda functions and def defined function

```
def cube(y):
    return y*y*y

lambda_cube = lambda y: y*y*y

print(cube(5))
print(lambda_cube(5))
```

Python Built-In Functions

all()

The python all() function accepts an iterable object (such as list, dictionary, etc.). It returns true if all items in passed iterable are true. Otherwise, it returns False. If the iterable object is empty, the all() function returns True.

```
k = [1, 3, 4, 6]          # all values true
print(all(k))

k = [0, False]           # all values false
print(all(k))

k = [1, 3, 7, 0]         # one false value
print(all(k))

k = [0, False, 5]        ## one true value
print(all(k))

k = []                   # empty iterable
print(all(k))

test1 = []
print(test1,'is',bool(test1))
test1 = [0]
print(test1,'is',bool(test1))
test1 = 0.0
print(test1,'is',bool(test1))
test1 = None
print(test1,'is',bool(test1))
test1 = True
print(test1,'is',bool(test1))
test1 = 'Easy string'
print(test1,'is',bool(test1))

x = 10
print('Absolute value of -40 is:', abs(x))    #abs() function is used to return the absolute
```

```
value of a number
floating = -20.83
print('Absolute value of -20.83 is:', abs(floating))
y = bin(x) #bin() function is used to return the binary representation of a specified
integer.
print (y)
s = sum([1, 2,4 ]) #sum() function is used to get the sum of numbers of an
iterable, i.e., list.
print(s)
print(float(9)) # float() function change into float number
print(complex(9)) # complex() function change into complex number
```

Data Handling and Visualization using Python

Samarth Godara

ICAR-Indian Agricultural Statistics Research Institute, New Delhi - 110 012

samarth.godara@icar.gov.in

Introduction

In the ever-expanding realm of data science and analysis, Python has emerged as a powerhouse for handling and visualizing data. Its versatility, extensive libraries, and straightforward syntax make it a preferred choice among professionals and enthusiasts alike. This article delves into the fundamentals of data handling and visualization using Python, exploring essential libraries and techniques to transform raw data into meaningful insights.

Data Handling with Pandas

At the core of data handling in Python lies the Pandas library, a robust and efficient tool for data manipulation and analysis. Pandas provides two primary data structures: Series and DataFrame. A Series is a one-dimensional labeled array, while a DataFrame is a two-dimensional tabular structure akin to a spreadsheet.

Key Concepts:

- Loading and Reading Data: Pandas supports various file formats, such as CSV, Excel, and SQL, making it easy to import and read datasets into DataFrames.
- Data Cleaning: Pandas facilitates the cleaning and preprocessing of data, including handling missing values, removing duplicates, and converting data types.
- Indexing and Slicing: Effective indexing and slicing operations enable users to access specific subsets of data, enhancing the efficiency of data exploration.

At the heart of data handling in Python, the Pandas library stands as a robust and efficient tool, providing a versatile set of functionalities for data manipulation and analysis. Pandas excels in handling structured data, offering powerful tools to load, clean, and manipulate datasets. Two fundamental data structures in Pandas, namely Series and DataFrame, form the cornerstone of its capabilities.

Series and DataFrame: Foundation of Pandas

Series:

A Series in Pandas is a one-dimensional labeled array capable of holding any data type. It is similar to a column in a spreadsheet or a single variable in statistics. Each element in a Series is associated with a label or index, allowing for efficient data retrieval and manipulation.

DataFrame:

A DataFrame, on the other hand, is a two-dimensional tabular structure that resembles a spreadsheet or a SQL table. It is a powerful data structure for handling heterogeneous data, where each column can be of a different data type. DataFrames provide a comprehensive and intuitive way to represent and analyze structured data.

Key Concepts in Pandas:

Loading and Reading Data:

Pandas simplifies the process of loading and reading data from various sources. It supports numerous file formats, including CSV, Excel, SQL databases, and more. The `read_csv()`, `read_excel()`, and `read_sql()` functions enable users to seamlessly import datasets into Pandas DataFrames, facilitating easy access and manipulation.

Data Cleaning:

Data cleaning is a critical step in the data analysis pipeline, and Pandas excels in this aspect. It offers functionalities to handle missing values, remove duplicates, and convert data types. The `dropna()`, `drop_duplicates()`, and `astype()` functions are just a few examples of Pandas tools that streamline the data cleaning process.

Indexing and Slicing:

Effective indexing and slicing operations in Pandas enable users to access specific subsets of data quickly. Whether it's selecting specific columns or filtering rows based on conditions, Pandas provides an array of techniques for efficient data exploration.

Exploratory Data Analysis (EDA) with Pandas

Once data is loaded and cleaned, the next step is Exploratory Data Analysis (EDA). This phase involves understanding the characteristics of the data, identifying patterns, and uncovering insights that lay the groundwork for more advanced analysis.

Key Techniques:

- Descriptive Statistics: Pandas provides functions for generating descriptive statistics, including mean, median, standard deviation, and percentiles, offering a quick snapshot of the dataset's central tendencies.
- Data Visualization: While EDA is not solely about visualization, Pandas integrates seamlessly with Matplotlib and Seaborn, allowing users to create informative plots and charts to better understand the data distribution and relationships.

Unveiling Patterns: Exploratory Data Analysis (EDA) with Pandas

After successfully loading and cleaning a dataset using Pandas, the subsequent phase in the data analysis journey is Exploratory Data Analysis (EDA). This pivotal step involves delving into the dataset to grasp its nuances, identify patterns, and extract insights that form the foundation for more advanced analyses. Pandas, with its rich functionality, plays a key role in facilitating effective EDA.

Key Techniques in EDA with Pandas:

Descriptive Statistics:

Descriptive statistics provide a concise summary of the main aspects of a dataset, offering a snapshot of its central tendencies. Pandas equips users with an array of functions to generate these statistics quickly and efficiently.

- Mean, Median, and Standard Deviation:

Calculating the mean, median, and standard deviation helps understand the distribution and central tendency of numerical features.

- Percentiles:

Utilizing percentiles provides insights into the spread and distribution of data points, helping identify outliers.

Data Visualization:

While EDA is not solely reliant on visualization, it significantly enhances the exploration process. Pandas seamlessly integrates with visualization libraries such as Matplotlib and Seaborn, enabling users to create informative plots and charts that offer a visual representation of the data.

- Histograms:

Histograms provide a visual depiction of the distribution of numerical data, aiding in the identification

of patterns and outliers.

- **Box Plots:**

Box plots are effective for visualizing the distribution, central tendency, and variability of numerical data.

- **Scatter Plots:**

Scatter plots are valuable for exploring relationships between two numerical variables, revealing potential correlations.

Data Visualization with Matplotlib and Seaborn

Matplotlib and Seaborn are fundamental libraries for creating static, animated, and interactive visualizations in Python. Matplotlib provides a high-level interface for drawing attractive and informative statistical graphics, while Seaborn simplifies common visualization tasks and enhances the aesthetics.

Key Features:

- **Line Plots and Scatter Plots:** Matplotlib enables the creation of line plots and scatter plots, essential for visualizing trends and relationships in numerical data.
- **Bar Plots and Histograms:** Seaborn enhances data representation with visually appealing bar plots and histograms, useful for comparing categories and understanding the distribution of numerical data.
- **Heatmaps and Pair Plots:** Seaborn's advanced features include heatmaps for visualizing correlation matrices and pair plots for exploring relationships between multiple variables simultaneously.

Visualizing Insights: Matplotlib and Seaborn in Action

Data visualization is a pivotal aspect of data analysis, providing a medium through which patterns, trends, and relationships within datasets can be effectively communicated. In the Python ecosystem, Matplotlib and Seaborn stand as foundational libraries, empowering users to create compelling and informative visualizations.

Matplotlib: High-Level Visualization

Matplotlib is a versatile and comprehensive plotting library that enables the creation of a wide range of static, animated, and interactive visualizations. Its high-level interface simplifies the process of generating aesthetically pleasing statistical graphics.

Line Plots and Scatter Plots:

- **Line Plots:**

Line plots are instrumental in visualizing trends and patterns in numerical data over a continuous interval.

Matplotlib provides a straightforward method for creating line plots.

- Scatter Plots:

Scatter plots are effective for visualizing the relationships between two numerical variables, facilitating the identification of correlations.

Seaborn: Simplifying Visualization Tasks

Seaborn is built on top of Matplotlib and specializes in simplifying common visualization tasks, enhancing aesthetics, and providing a high-level interface for creating visually appealing plots.

Bar Plots and Histograms:

- Bar Plots:

Seaborn enhances data representation with visually appealing bar plots, making it easy to compare categories within a dataset.

- Histograms:

Seaborn's histogram functionality adds a layer of style to visualize the distribution of numerical data.

Advanced Features: Heatmaps and Pair Plots

Seaborn extends its capabilities to advanced features, catering to more complex visualization needs.

- Heatmaps:

Heatmaps are powerful for visualizing correlation matrices, and Seaborn simplifies their creation.

- Pair Plots:

Pair plots allow for the exploration of relationships between multiple variables simultaneously.

Python's Empowering Ecosystem: From Raw Data to Actionable Insights

In the dynamic world of data analysis, Python has emerged as a formidable force, thanks to its rich ecosystem of libraries that seamlessly handle data and visualize insights. This comprehensive toolkit empowers users to navigate the complexities of raw data and distill actionable insights. Among the standout libraries contributing to Python's prowess are Pandas, Matplotlib, Seaborn, and Plotly.

Pandas: Efficient Data Manipulation

At the forefront of Python's data handling capabilities is Pandas, a versatile library that simplifies the manipulation and analysis of structured data. Whether loading datasets from various sources, cleaning data, or exploring its intricacies through indexing and slicing, Pandas provides an efficient and intuitive

interface. With the power of Pandas, users can transform raw data into a structured format conducive to in-depth analysis.

Matplotlib: Crafting Informative Visuals

Matplotlib, a fundamental visualization library, plays a crucial role in transforming data into compelling visuals. Its high-level interface allows users to create an array of plots, including line plots and scatter plots, to reveal trends and relationships in numerical data. The ability to generate visually appealing plots enhances the communicative power of data, making it easier for analysts to convey complex information in an accessible manner.

Seaborn: Simplifying Visualization Tasks

Building upon Matplotlib, Seaborn is designed to simplify common visualization tasks and enhance aesthetics. With Seaborn, users can create visually striking bar plots, histograms, and more. Its advanced features, such as heatmaps for visualizing correlation matrices and pair plots for exploring relationships between multiple variables, add depth to the visualization toolkit. Seaborn's emphasis on aesthetics ensures that data visualization becomes not just informative but also visually engaging.

Plotly: Interactive Visualization

Plotly takes data visualization to the next level by providing capabilities for interactive plots and dashboards. This library is instrumental in creating dynamic visualizations that allow users to explore data in real-time. Through Plotly, Python users can build interactive charts, maps, and dashboards, facilitating a more engaging and immersive data exploration experience.

In conclusion, Python's strength in data handling and visualization lies in the synergy of its diverse libraries. From efficiently managing data with Pandas to crafting informative visuals with Matplotlib and Seaborn, and finally, to the interactive capabilities of Plotly, Python offers a comprehensive ecosystem for transforming raw data into actionable insights. This empowering toolkit not only caters to the diverse needs of data analysts, scientists, and engineers but also fuels innovation in various domains by providing a solid foundation for data-driven decision-making. As the field of data analysis continues to evolve, Python's ecosystem remains at the forefront, ensuring that users have the tools they need to extract valuable insights from the ever-expanding sea of data.

E-Governance initiatives in ICAR

Dr. Mukesh Kumar

ICAR-Indian Agricultural Statistics Research Institute, New Delhi - 110 012

mukesh.kumar@icar.gov.in



What is E-Governance

E-Governance is the application of Information Technology to the processes of Government functioning to bring about.

Smart

- Moral
- Accountable
- Responsive
- Transparent Governance

Purpose of E-Governance

- E-Governance, expand to electronic governance, is the integration of information and Communication Technology (ICT) in all the process, with the aim of enhancing government ability to address the needs of general public.

- The basic purpose of E-Governance is to simplify processes for all, i.e. government, citizens, businesses, etc at National, State and Local levels.
- To promote good governance. It contains the implementation of information technology in the government processes and functions so as to cause simple, moral, accountable and transparent governance.
- To provide access to government services, dissemination of information, Communication in quick and efficient manner.

Benefits of e-Governance



Type of E-Governance

1. Government to Citizen (G2C)

- E service
- Applications
- Record request
- Tax Remittances
- Public Utility Invoicing
- ePayment
- Complaints and Feedback
- Voting
- Public Opinions

2. Government to Government (G2G)

- Budget Management
- Policy Implementations
- Approvals

- Revenue Management
- Trainings
- Notice and Alerts
- Projects

3. Government to Employee (G2E)

- Attendance
- Policies
- Payroll
- Administration
- Self-Services
- Pension
- Healthcare

4. Government to Business (G2B)

- Taxation
- Licencing
- contracting and Procurement
- Subsidies
- Approvals

National eGovernance Plan (NeGP)

NeGP, takes a holistic view of E-Governance initiatives across the country, integrating them into a collective vision, a shared cause, The Government approved the national e-Governance Plan (NeGP), comprising of 27 mission mode projects on May 18, 2006. In the year 2011, 4 Project Health, Education, PDS and Posts were introduced.

1. Central MMPs

- Banking
- Central Excise & Custom
- Income Tax
- Insurance
- MCA21
- Passport
- Immigration, Visa and Foreigners

- Registration and Tracking
- Pension
- e-office
- Posts
- UID

2. State MMPs

- Agriculture
- Commercial Taxes
- e-District
- Employment exchange
- Land Records
- Municipalities
- e-Panchayat
- Police
- Road Transport
- Treasuries Computerization
- PDS
- Education
- Health

3. Integrated MMPs

- CSC
- e-Biz
- e-courts
- e-procurements
- EDI for Trade
- National e-Governance service delivery gateway
- India Portal

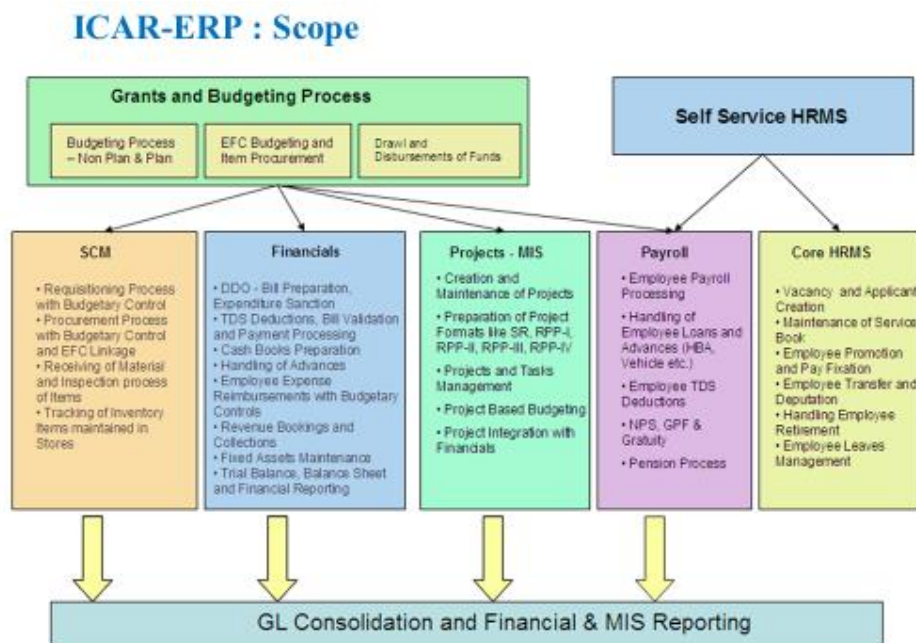
Major e-Governance Initiatives in ICAR

- ICAR-ERP
- ICAR Unified Communication Solution
- E-Office

- Personal Management System
- Foreign Visit Management System
- Agricultural Research Management System
- Land Record Management System
- Institute Ranking System
- KRISHI-ICAR Research Data Repository for Knowledge Management
- Kisan SARATHI
- Krishi Megh (ICT Infrastructure)

1. Enterprise Resource Planning (ERP)

- **Financial Management:** General ledger, Accounts Payable, Accounts Receivable, Cash Management, Fixed Assets Management, Budget Management and Grants.
- **Project Management:** Project Information, Costing, Project Documents, Contract Management and Collaboration of Project documents.
- **Supply Chain Management:** Purchase and Inventory Management.
- **Human Resource Management:** Employee information, HR policies, Leave Management, Performance and Appraisal System.
- **Payroll System:** Salary, GPF, Pension Payment, Retirement Benefit Calculation and Income tax calculation
- **Self Service HR:** Personnel & Professional Information, GPF, Leave, Salary etc.



2. ICAR Unified Communication Solution

Accessing webmail is as easy as using your web browser to visit a URL, and then entering your account name and password:

Use your web browser to go to to <https://mail.icar.gov.in>.

You should always use a URL that starts with <https://> (the HTTP Secure protocol).

The webmail login page appears:

- In the User Name text box, type the e-mail address for the account you want to access.
- In the Password text box, type the password for the e-mail account.
- Click Sign in.

Password Policy:

- While changing password, you need to input “icar\user id” in User Name field and your desired password in Password field
- Please change your password as per password policy mentioned below: -
 - Password should be of minimum 8 characters.
 - It must contain at least one special character, one numeric character & one capital letter.
 - Do not use your name/institute name in password.
 - Do not use your last 3 passwords.
 - Password will expire in 180 days.

Total User Count –

- AD Users: 25000+
- Mailbox Users: 24000+

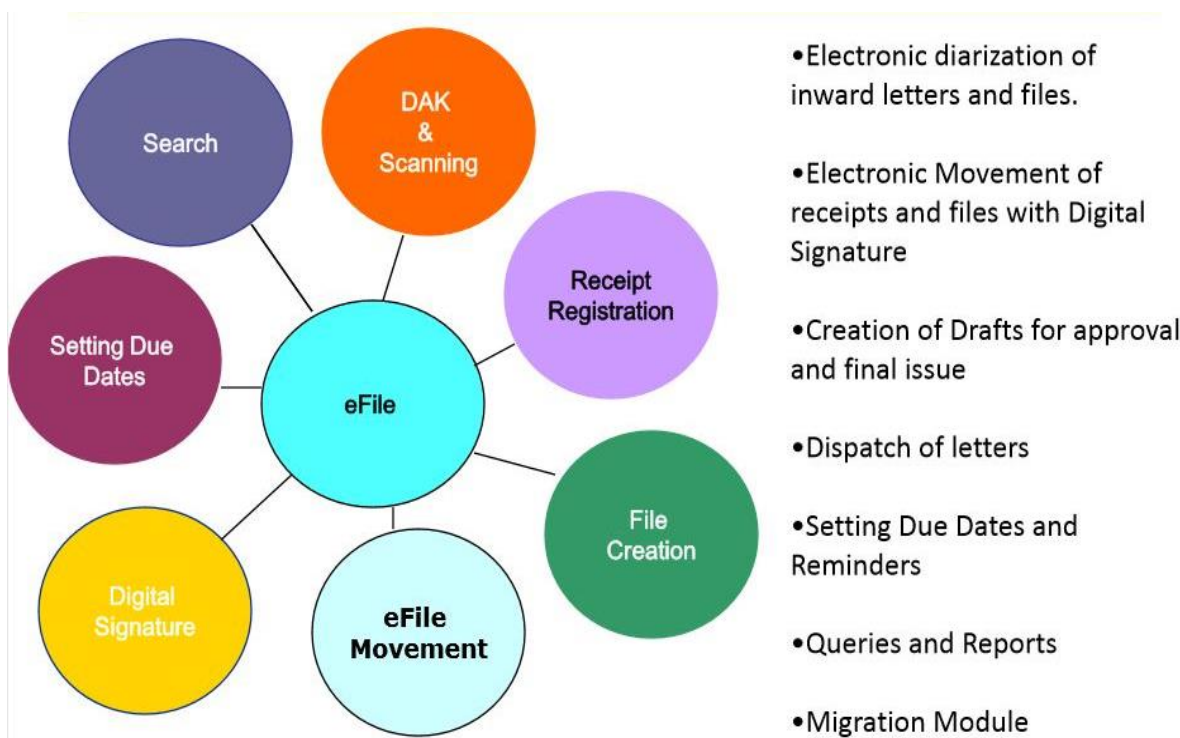
Website Hosting –

- Currently 400+ website/webapplication/webportal running from ICAR datacentre.

3. e-Office – A Digital Workplace Solution

- eFile is a workflow based system that replaces the existing manual handling of files with a more efficient electronic system.
- This system involves all stages, including the electronic diarization of inward correspondence, creation of files, movement of correspondences and files, electronic signing of noting & drafts using Digital Signature Certificates (DSC), eSign, and finally, the archival of records.

Receipts	Files
<ul style="list-style-type: none"> • Diarization – Electronic / Email / Physical • Acknowledgement Generation • Receipt to Receipt and File Attachment • VIP Letter Tracking • Address Book • Signing on remarks • Legends on priority • Advanced Search on metadata • Receipt Status Monitoring System • Closing of Receipts 	<ul style="list-style-type: none"> • File Creation – Electronic and Physical • Notings (Green and Yellow Note) • Correspondence • Draft for Approval (DFA) • Referencing • Digital Signatures on Noting and DFA • File to File and Receipt Attachment • Linking of File • Closing of File • Advanced Search on metadata
Dispatch	Reports
<ul style="list-style-type: none"> • Templates Selection • Digital Signatures • Advanced Search on metadata • Reminders and Follow-ups • Dispatch sent through email and post 	<ul style="list-style-type: none"> • MIS Reports <ul style="list-style-type: none"> ➤ File/Diary Register Report ➤ File/Diary Movement Report ➤ File/Diary Pendency Reports <p>..... many more</p>



4. Personal Management System

Management Information System has been designed, developed and implemented across ICAR. The system is accessible at <http://pms.icar.gov.in>

The system has following functionalities:

- Ability to add, modify, retire, transfer scientists belonging to various cadres such as scientist, senior scientist, principal scientist, RMP positions (HoDs, Director, ADG, DDG, DG). Service details of all scientists were updated by individual scientists/PME incharge. ERP ids of scientist were also mapped to their service details.
- Workflow based system for posting of newly appointed scientists from NAARM to various institutes. Workflow starts from online application of scientists, forwarded by Director NAARM, and then posting by Under Secretary, Personal after getting approval by competent authority. The first implementation of this module was executed successfully in March 2017.
- Content management facility has been incorporated in the PMS to facilitate management of Discipline, Institute, Designation, Cadre Type, SMD, State, Sanctioned Strength and Transfer Cycles.
- AIPR (Annual Immovable Property Return) upload functionality can be added.

- Workflow based Transfer application module consisting of 5 types of transfer cycle has been developed and is available for use. It includes transfer cycles depending upon the place of posting and duration at present place of posting.

5. Foreign Visit Management System

- Foreign Visit Management System of DARE-ICAR is an online system for managing all activities of foreign visits/training/fellowships undertaken by ICAR employees.
- The work flow process of foreign visit application is digitized in FVMS thereby reducing delays in foreign visit proposals.
- The system enables the employee to submit the application online which goes through the online approval process and the status of the application is displayed to the employee.
- Email notification about the approval of the application is also automated. The employee and concerned officials get the email notifications in their official email.
- It is benefited both applicant and DARE officers in processing of visit application more efficiently and transparently.
- System maintain repository on the foreign visit deputation report in searchable format.

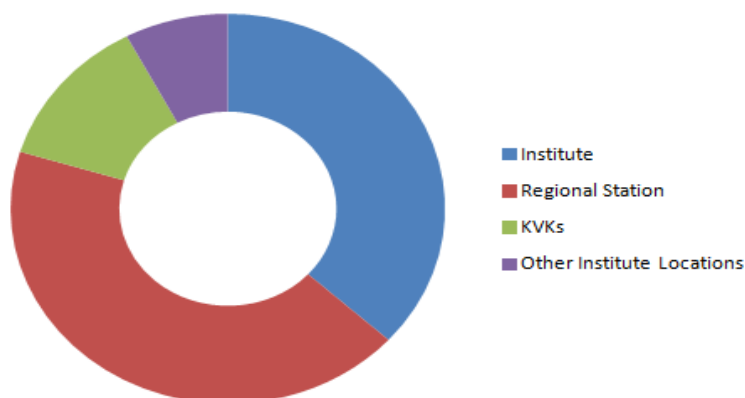
6. Agricultural Research Management System

- ARMS is an important tool developed and implemented in ICAR to monitor and assess scientific achievements.
- This system is used for information management and evaluation of scientific research of the Council based on the information submitted by scientists in this system.
- Achievements uploaded in the system monthly basis on completion. The information submitted by scientist is duly verified by Reporting Officer, PME In-charge and Reviewing Officer at various stages in a prescribed time period.
- Scientists able to connect the research they are undertaking to the priorities of the Government, Council, Region and Institute and these priorities are mapped by a proposed Standing Committee on Research in the Council.

- All scientists have required to enter the information into the system only once, this would save the scientist from the hassle to provide information many times for different requirements.
- System have provision to provide various information in report formats (Monthly progress report of Division/Institute/SMD/Annual report material of Institute).
- In future, information generated from this system also be linked with different dashboards of Government of India. So, the System could help in monitoring and management of agricultural research on real time basis.

7. Land Record Management System

- ICAR- Land Records Management System (LRMS) has been developed it can be accessed using the URL <https://lrms.icar.gov.in/>. This is an integrated system which provides land record information of all institutions along with their Regional Stations.
- The system keeps the online record as total land area, land utilization details (Farm area, Research Area, area under building, area under sports ground/park/green area, forest area, vacant land), Ownership description as per revenue record, Date of Possession as per revenue record, Date Acquisition, Free hold land/leased hold, Lease Period, Date Start for lease, Date of renewal of Lease etc.
- It also generates advisory and sends email to the Director of the institute, Head of Administration along with Director (Works) about expiry of lease where lease would expire within one year.
- The Total land available with all the ICAR institutes and the ICAR headquarter is more than 54 thousand acres.
- In total **236** institutes/ KVKs/ Regional stations have filled their land record data into the system, so far. The details number of institutes, KVKs and regional stations are given below:



- The information obtained from this system can be readily used for deciding the future policies for land management by council.

8. Institute Ranking System

- The Institute Ranking System framework outlines the indicators and methodology for ranking the relative performance of ICAR institutes.
- These indicators are broadly categorized into three groups i.e. (i) Institute Performance (ii) Recognitions of the Institute and (iii) Research Output of the Institute. Each of these three groups have been assigned differential weightages based on their importance i.e. 30%, 10% and 60% respectively for computing over all institute score.
- Further, this score is divided by the number of scientists in the institute for determining the relative ranking of the institute. Indicators for ranking of the institutes were finalised based on the discussion in various meetings at different levels.
- The url of the IRS: <https://irs.icar.gov.in/index.php>

9. KRISHI-ICAR Research Data Repository for Knowledge Management

- KRISHI - Agricultural Knowledge Resources and Information System Hub for Innovations, is an initiative of Indian Council of Agricultural Research (ICAR) to bring its knowledge resources to all stakeholders at one place.
- The portal is developed as a centralized data repository system of ICAR consisting of Technology, Data generated through Experiments/ Surveys/Observational studies, Geo-spatial data, Publications, Learning Resources etc.
- Photo Gallery Video and Audio Gallery Dashboard
- ICAR-IPR Repository Variety Information System

10. Kisan SARATHI - System of Agri-information Resources Auto-transmission and Technology Hub Interface

- The Kisan SARATHI has been customized/developed and implemented by ICAR-Indian Agricultural Statistical Research Institute in association with Digital India Corporation, Ministry of Electronic and communication Technology (MeitY), Government of India under an MoU.

- Initially the services have been started in four major states of India viz. Bihar, Madhya Pradesh, Maharashtra and Uttar Pradesh later on the services were extended to two more states viz. Andhra Pradesh and Telangana. Subsequently the services of Kisan Sarathi were made available to all the States and UTs in the month of June, 2022
- This is an on call advisory services for farmers of India, where any farmer can call or record their query in his own language - automatically directed to respective KVK/ATARI for query redressal. The Queries are being responded in the same language by the concerned KVK either online or by call later on based on the recorder queries of the farmer. This platform supports multilingual messaging system where bulk or individual SMS can be send to group of farmers based on either their location of the crops they cultivate. KVK use this facility to send advisories to the large chunk of farmer in a go. This system provides the services through IVR based service on toll free number- 1800-123-2175 and a short number 14426 to the farmers of all states.
- Presently, services of Kisan Sarathi are being provided by the more than three thousand Agriculture experts from 731 KVKs across the country to more than 1.68 crore registered farmers which covered more than 2.58 Lakhs villages.

11. Krishi Megh (ICT Infrastructure)

- The Krishi Megh infrastructure and services has been jointly developed by ICAR-Indian Agricultural Statistics Research Institute (IASRI) and ICAR-National Academy of Agricultural Research Management (NAARM) under the National Agricultural Higher Education (NAHEP) component-2 Project “Investment in ICAR leadership in higher education” project.
- NAHEP aims to develop ICT infrastructure for providing means for better governance and management, so that holistic model can be developed to raise the standard of Indian Agricultural Research and Education System (NARES)

KRISHI Megh is consist of ICAR Data Center , Artificial Intelligence and Cloud Computing & Disaster Recover Centre

I. ICAR Data Center (an ISO 27001:2013 and 20000:2011):

- ICAR-DC at ICAR-IASRI is providing IT Services efficiently to DARE, KVKs, ICAR and its institutes since September 2014.

- The facility are built in a state-of-art Data Centre equipped with industry standard 960 Cores Compute, 6224 GB RAM, 400 TB Storage, Networking, Cyber Security, Power Precision Colling, Building Management System, Software Application and other related technologies.
- Currently more than 260 websites/web applications are running from ICAR Data Centre and 21709 Mailbox Users “icar.gov.in”.

II. Cloud Computing & Disaster Recover Centre

- Building clouding Computing with Disaster Recovery Centre (DRC) at NAARM, Hyderabad is expansion of ICAR-DC at ICAR-IASRI, New Delhi.
- It is built on cutting edge technology of industry standard of Data Centre Components of Smart Rack, 804 Cores & 2940 RAM Computing, 225 TB Storage, Hyper Converged Infrastructure (HCI), 100 Gig Structured, Scalable and Resilient Network Architecture Networking, Global Load Balancer and Cyber Security with Multilayer, Cross Hybrid, Advance Threating Protection, Deep Security, DLP, Next Generation Network and Web Application Firewall

KRISHI Megh ensures

- Availability and reliability of e-Governance and IT system in the field of agricultural research and education
- Resilient IT infrastructure for fostering innovation to enhance scientific research and development
- Ready to use platform for agri-entrepreneurs, educationists, researchers and extension professional.
- Immediate and authentic information disbursement to researchers, students, farmers and policy planners

Mobile Application Development

Md. Ashraful Haque and Chandan Kumar Deb

ICAR-Indian Agricultural Statistics Research Institute, New Delhi - 110 012
ashraful.haque@icar.gov.in

Introduction

Android is a mobile operating system. It is an open-source framework and is based on Linux. The Android framework helps us to develop advanced and user-friendly applications. The applications can be built using Java and Kotlin. The Android operating system has then gone through numerous releases by fixing bugs as well as adding additional features which make our life more comfortable and easier.

History of Android Technology

It is an operating system developed by Android Inc. and then overtaken by Google. Android Inc. was developed in Palo Alto California, in October 2003 by Andy Rubin, Rich Miner, Nick Sears, and Chris White.

, in 2007. The first version was released by Google and the commercial version was released in 2008 known as Android 1.0.

Besides, numeric names, Google has assigned code names to all Android versions. The following picture depicts all the versions and their code names.



Version

- The first version of Android 1.0, released on 23 September 2003.
- The second version, released on 9 Feb 2009.
- Then version 1.5 known as Cupcake, released on 27 April 2009.
- Version 1.6 known as Donut, released on 15 Sept 2009.
- Then on 26 Oct 2009 version 2.0-2.1 was released.
- Froyo version 2.2-2.2.3, released on 20 May 2010.
- On 6 Dec 2010 gingerbread version 2.3-2.3.7 was released.
- Honeycomb version 3.0-3.2.6, released on 22 Feb 2011.
- Ice cream version 4.0-4.0.4, released on 18 Oct 2011.
- Jelly bean version 4.1-4.3.1, released on 9 July 2012.
- Kit-kat version 4.4-4.4.4, released on 31 Oct 2013.
- Lollipop version 5.0-5.1..1, released on 12 Nov 2014.
- Marshmallow 5 version 6.0-6.0.1, released on Oct 2015.
- Nougat version 7.0-7.1.2, released on 22 Aug 2016.
- Oreo version 8.0-8.1, released on 21 Aug 2017.
- Version 9 also known as Pie, released on 6 Aug 2018.
- Version 10 also known as Android 10, released on 3 September 2019.

Why Learn Android Technology?

Android technology is not constrained to only cell phones, nowadays there are many devices in the market that use it as their operating system. Devices like television sets, tablets, Android auto cars, ebook-reader, and wristwatches use Android as the operating system. It is leading the global market. Most of the population uses Android devices. The applications we use every day has brought plenty of jobs available for Android developers in the market. As Android is open-source anyone can learn and build Android applications. Also, Android applications are compatible with a variety of devices. No doubt Android development is one of the enthusiastic and interesting jobs in this period of technology.

Android Features

As we know the Android has changed our lives, let's discuss some of the Android features.

1. Voice-search

This feature lets the user search by recording the voice message instead of typing it. Example- If we want to call XYZ person, we just have to speak and the call will be directed to the XYZ person, performing multi-tasking. With this feature, we can watch a video and also play games simultaneously.

2. Screen-capture

We can capture the screen using this feature.

3. Multiple Language Support

English is the default language but now we can use any local language. Also, Android supports multiple languages.

4. Gestures

With the help of gestures, we can use the phone without even touching it.

5. Tethering

With this feature, we can share internet connections through the wired./wireless hotspot.

6. Media Support

Android supports the following media H.263, H.264, MPEG-4, AMR, AMR-WB, AAC, HE-AAC, MP3, JPG, PNG, etc.

7. Storage

SQLite is an open-source relational database that is inbuilt in Android.

8. Auto-correct

This feature suggests words and corrects grammatical mistakes.

9. Sensors

Almost, every mobile phone has inbuilt sensors that sense the motion of the phone. Some of the inbuilt sensors are an accelerometer, heart rate, magnetic field sensor, gyroscope.

Android Architecture

When we talk about Android we know that it is an operating system and we use applications that are built on it. Now we need to understand exactly what is Android and what enables it to work the way it does. For that, we need to have an exact vision of its components and functionalities.

Here comes the need and use of Android Architecture. Android Architecture is a software stack of components that are required to build an Android Application. Android Architecture contains 5 major layers that an Architecture works in. Android Architecture explains the complete working of Android through these five layers only.

These five layers are:

System Applications

Java API Framework

Native Libraries

Android Runtime

Linux Kernel

Frameworks for Android Development

Let's discuss the tools required for application development. There are many tools available for Android development. Some of the best Android development tools are listed below:



Android Applications

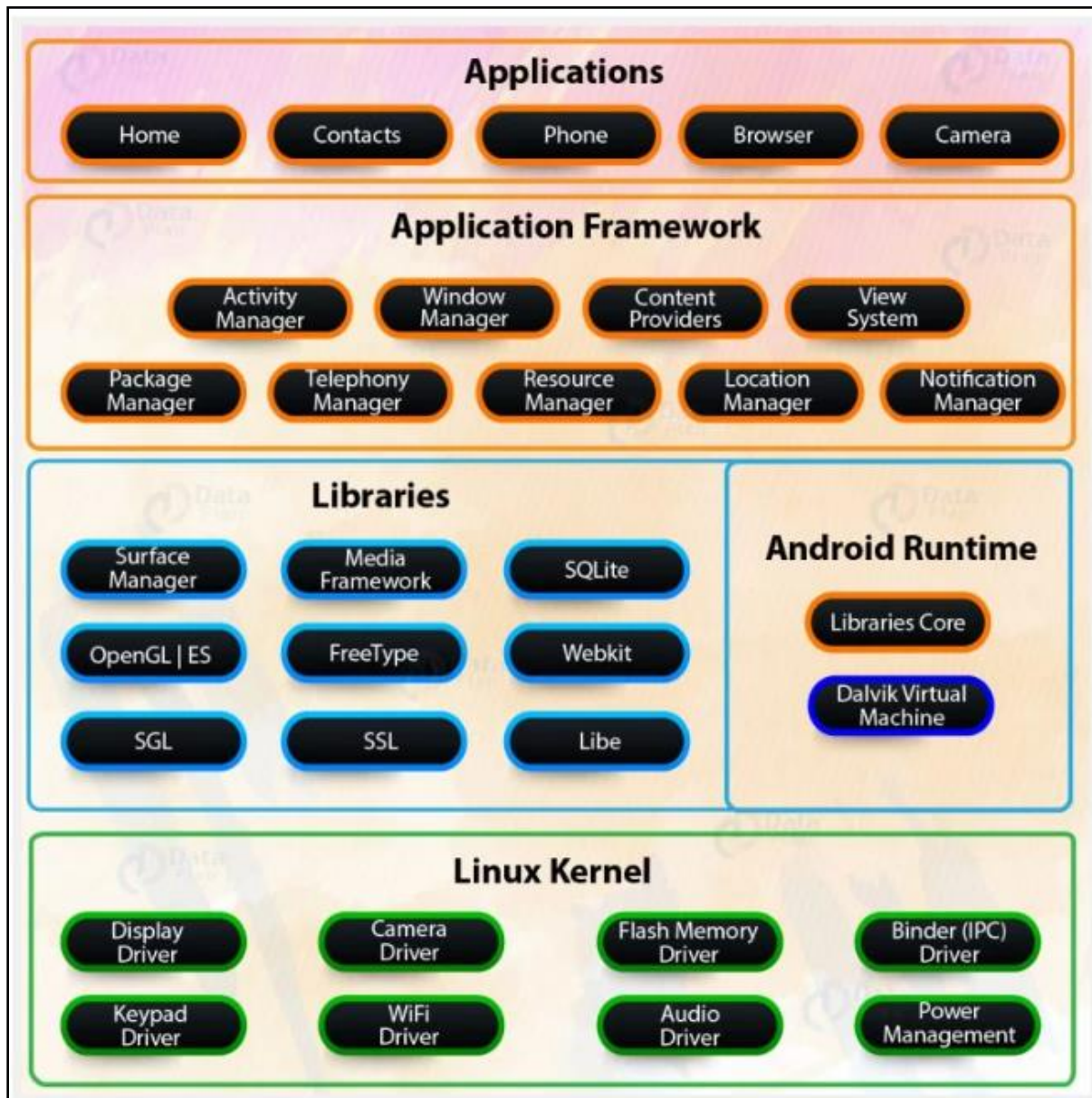
In this Android tutorial, we have discussed various types of applications that can be created using Android development which helps to save time and energy. Various types of applications are Music, Lifestyle, Social Media, Navigation, Finance, Weather, Travel etc.

Top Android Features

Following is the list of unique and best Android features:

Split-Screen, Dark Mode, Auto-correct, Gesture Control, Direct USB Access, Bluetooth, Multiple Language Support, Text To Speech Feature, Storage And Battery Swap, Customize Home Screen, Notification Dots, Snooze Notification, Fingerprint scanner, Alternate Keyboard, Infrared Transmission, Improved Files App, Google Assistant, Overheating USB Warnings.

Android Architecture



When we talk about Android applications, there is a wide range of languages available for Android app Development. Before getting to which one is the best, let us see some languages that you can consider for development. Following are few of the common languages for Android Application development:

Top 7 Programming Languages for Android App Development

- Java
- Kotlin
- C#
- Python
- C/C++
- Corona
- BASIC

Android Application Components

1. Activities
2. Services
3. Content Providers
4. Broadcast Receiver

Additional Components of Android Application

1. Intents
2. Widgets
3. Views
4. Notifications
5. Fragments
6. Layout XML Files
7. App APK files
8. Resources

What is Android Studio?

Android Studio is the Official IDE for Google's Operating System that is Android. It is built on JetBrains' IntelliJ IDEA software, especially for Android Application Development. This can be used on Windows, macOS, and also Linux based operating systems. It has risen as a replacement for Eclipse Android development tools for the native Android app development. Google announced Android Studio as the primary IDE for native android apps on 16th May 2013 at the Google I/O conference.

Java was the official language of Android Studio for Android app development, which was recently replaced by Kotlin on 7th May 2019.

Why is Android Studio used?

Android Studio is a development framework that has every tool required for Android Application Development. The purpose of developing it was to accelerate the development and help developers. It helps the developers to build the apps of the highest quality for every Android Device. It offers many tools to developers for coding, editing, designing, debugging, testing, and whatnot. Android Studio also has some features such as auto code completion and refactoring that makes the project development hasslefree. Android Studio is useful for Android application development if you're good with either Java or Kotlin.

What are Features of Android Studio?

We already know that Android Studio is based on [IntelliJ](#) that does all the things that Eclipse did with the ADT plug-in. Other than that, it has many things and many more new and exciting features that we will see. Without any delay let us check them out-

- It has Gradle-built support
- It lets debug the code easily and has quick fixes.
- There are lint tools that catch performance issues, usability, or version compatibility problems.
- It has a [Proguard tool](#) that is one of the best things in Android Studio.
- It has a template-based wizard that helps in creating common Android designs and layouts.
- This has a rich editor that allows us to drag and drop the UI components and check it very easily.
- It has a rich color preview editor that lets us add the color and the color name in the resource directory.

- It has a deep code analysis, which means it shows exactly where the bug is and also gives suggestions to correct it without any complexity.

Android Studio Installation

So guys, we know you are interested in Android Application development. To begin with application development in Android Studio, you must begin with its installation. Learn [Android installation in easy steps](#).

First Android App in Android Studio

Once you have Android Studio ready with you, now you can proceed to [create your first ever Android App](#). Before beginning with it, it would be good if you brush up the following skills-

- Object-Oriented Programming
- Java Programming
- Basics of extensible markup language

Android Studio for Android Applications

Android Studio has many exciting things that help and encourages developers to create many new applications. Let us see these things one by one in the following:

a. Code and debug Faster

As Android Studio is based on IntelliJ, it provides the fastest turnaround for its coding and workflow. It also has auto code completion that is of great use for the developers.

b. Easy and fast Run

It has an instant run feature that immediately runs the app using the codes and the resources provided. It is intelligent enough to understand the changes and show them immediately in the app.

c. Smart Code editor

Its code editor helps the developer in writing better code faster and more productive. For this, it offers advanced code completion, refactoring, and code analysis features. As you type, it provides suggestions from which you can easily choose the best fit.

d. Faster Emulator

The Android Emulator installs the apps faster than a real device and lets the developers test their app on various Android Virtual Devices. These devices can be smartphones, tablets, Android Wear, and even Android TV devices.

e. Great Build System

Android Studio offers rich dependency management and customizable build configurations for apps. The developers can configure the projects and include local as well as hosted libraries. The build variants that include different code and resources can also be defined.

f. Supports all Android devices

Android Studio IDE provides a unified environment to build apps for all Android devices. These devices are phones, tablets, Android Wear, and also TVs. It also has the facility of working in modules that allow developers to divide their projects into functional units that can be independently built, tested, and debug.

g. Templates and sample apps

Android Studio provides templates and sample applications that make it easy to add well-established patterns such as menu or navigation, etc. The developer can start with a code template or even an API in the editor. It also has this functionality where the code can be directly imported in the Android Studio and run.

h. Testing tools

Android Studio provides many tools that help in testing Android apps and their functional UI parts. UI test codes can be easily generated using Espresso Test Recorder, recording the interactions with the app on an emulator or a real device.

i. Keyboard shortcuts

Here are 10 shortcuts to help you through your Android Application Development:

- Ctrl + F – It will help you find words in your file.
- Ctrl + R – It will help you replace a word/ name.

- F5 – It will help you to copy.
- Ctrl+ (+/-) – It will let you zoom in or out.
- F11 – It will get the Android Studio in full-screen mode.
- Ctrl + B – It will open the XML file.
- Ctrl + N – It will help you find a class.
- Shift + F6 – It will help you to rename a class name or variable name.
- Ctrl + Shift+ N – It will help you find a file.
- Ctrl + I – It will help to implement methods.

Some other Tips

We would like to conclude this tutorial by providing you some tips and tricks. These will help you out throughout the journey of Android App development using Android Studio-

- You can change the theme as per your convenience as there is a light and a dark theme to use Android Studio.
- You can make a very fast search if you need to find the files or classes or symbols. It is very simple. You can simply press the Shift key two times, and there you go; you can search it.
- While writing the code, you would be given suggestions, pressing the tab key will type that automatically.
- You can also check the CPU, memory, and network usage of the applications in Android Studio. This can be done through the profile check feature in it.
- Also, if there is some issue with the name of the class or variable or method, you can easily change it too. You just need to right click and choose the refactor and then rename. And you are done.

What is Activity?

We know by now, that an activity is a single screen of an application that lets us see and interact to perform an activity. Usually, an application contains many screens and each screen extends **Activity()** class.

When we work on an application what we see is a UI on the screen which is an activity. Most of the applications that we use have many activities. Among all those activities, one is the **MainActivity()** & the rest are its **ChildActivities()**. Generally, the first page that appears on the screen when an application opens is referred to as MainActivity(). This main activity interacts with the child activities and lets the

user access them.

Android Activity Lifecycle

An activity can have four states, which are :

1. Running
2. Paused
3. Resumed
4. Stopped

The above are the four states that Android activity can achieve during its whole lifecycle.

1. Running State

An activity is in the **running** state if it's shown in the foreground of the users' screen. Activity is in the running state when the user is interacting with it.

2. Paused State

When an activity is not in the focus but is still alive for the user, it's in a **paused** state. The activity comes in this state when some other activity comes in with a higher position in the window.

3. Resumed State

It is when an activity goes from the paused state to the foreground that is an **active** state.

4. Stopped State

When an activity is no longer in the activity stack and **not visible** to the users.

1. onCreate()

The Android onCreate() method is called at the very start when an activity is created. An activity is created as soon as an application is opened. This method is used in order to create an Activity.

2. onStart()

The Android onStart() method is invoked as soon as the activity becomes visible to the users. This method is to start an activity. The activity comes on the forescreen of the users when this method is invoked.

3. onPause()

The Android onPause() method is invoked when the activity doesn't receive any user input and goes on hold. In the pause state, the activity is partially visible to the user. This is done when the user presses the back or home buttons. Once an activity is in the pause state, it can be followed by either **onResume()** or **onStopped()** callback method.

4. onRestart()

The Android onRestart() method is invoked when activity is about to start from the stop state. This method is to restart an activity that had been active some time back. When an activity restarts, it starts working from where it was paused.

5. onResume()

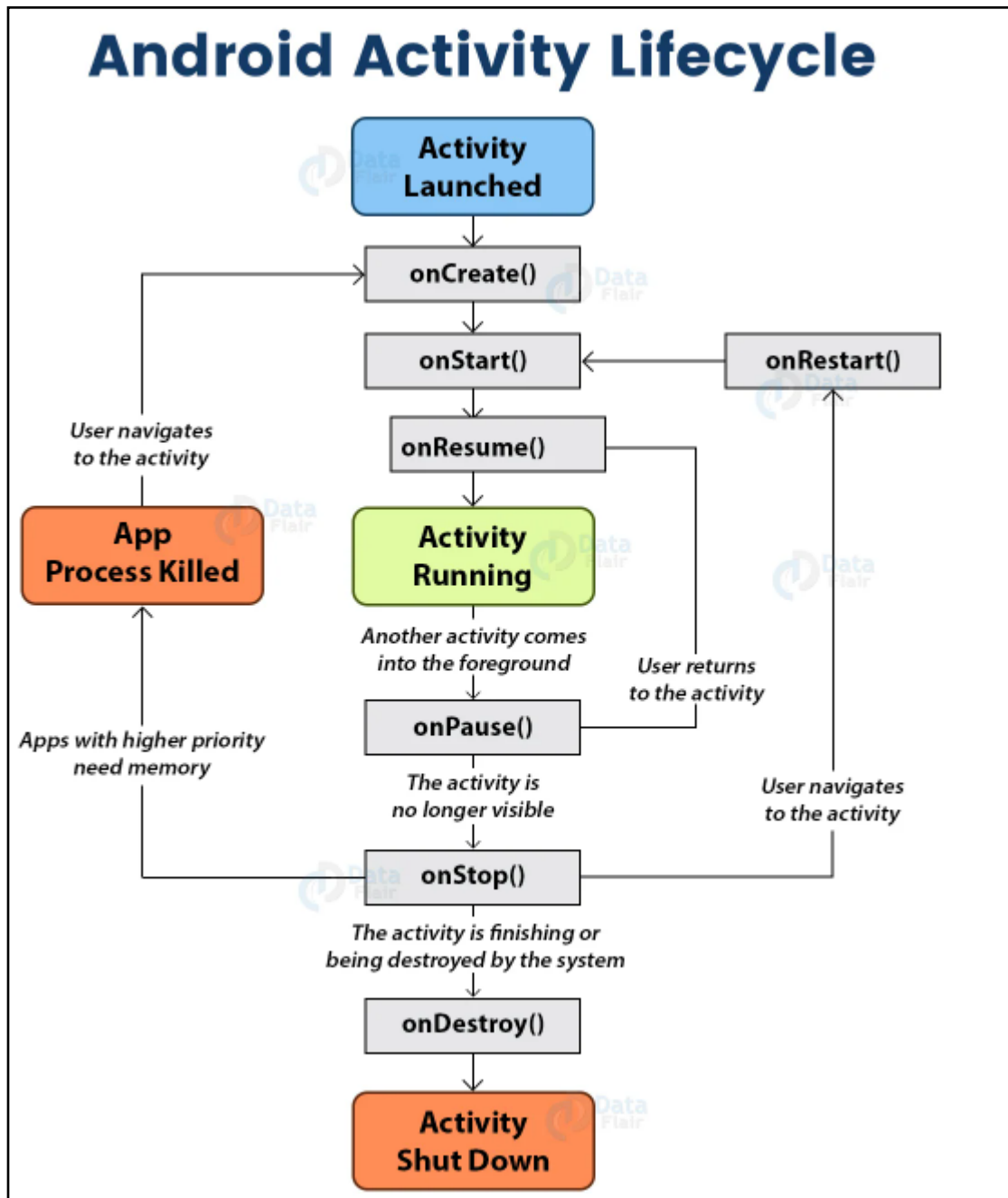
The Android onResume() method is invoked when the user starts interacting with the user. This callback method is followed by **onPause()**. Most of the functionalities of an application are implemented using **onResume()**.

6. onStop()

The Android onStop() method is invoked when the activity is no longer visible to the user. The reason for this state is either activity is getting destroyed or another existing activity comes back to **resume** state.

7. onDestroy()

The Android onDestroy() is the method that is called when an activity finishes and the user stops using it. It is the final callback method received by activity, as after this it is destroyed.



What are Android Services?

Android Services are the application components that run in the background. We can understand it as a process that doesn't need any direct user interaction. As they perform long-running processes without user intervention, they have no User Interface. They can be connected to other components and do **inter-process communication (IPC)**.

1. Foreground Services

Foreground services are those services that are visible to the users. The users can interact with them at ease and track what's happening. These services continue to run even when users are using other applications.

The perfect example of this is Music Player and Downloading.

2. Background Services

These services run in the background, such that the user can't see or access them. These are the tasks that don't need the user to know them.

Syncing and Storing data can be the best example.

3. Bound Services

Bound service runs as long as some other [application component](#) is bound to it. Many components can bind to one service at a time, but once they all unbind, the service will destroy.

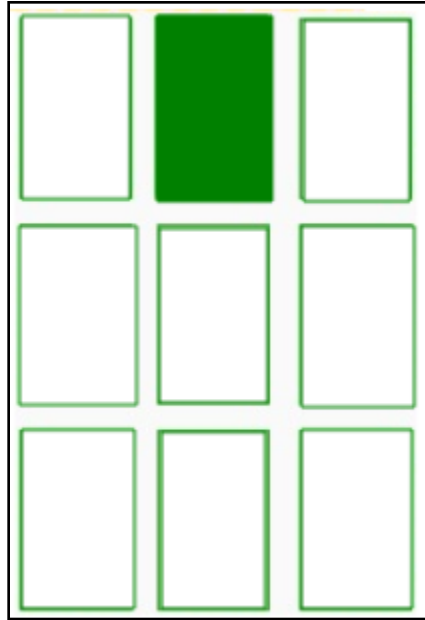
To bind an application component to the service, **bindService()** is used.

What is Android Broadcast Receiver?

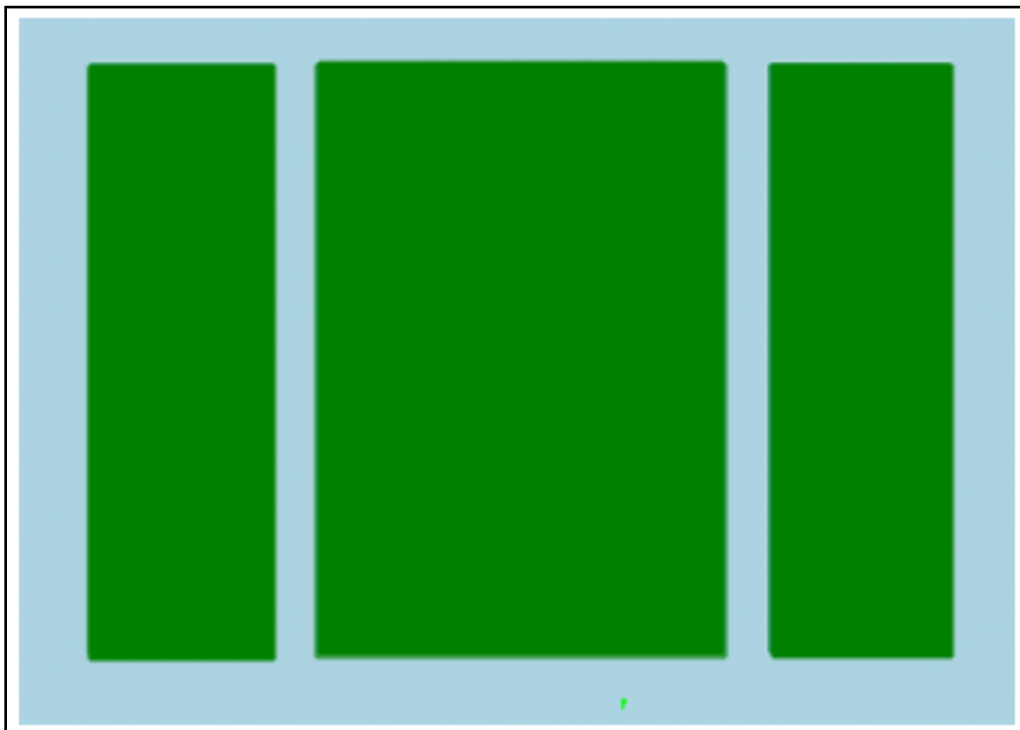
Android Broadcast Receiver is an Android component that is used to broadcast the messages to the system or other applications. The broadcast message is referred to as an Event or Intent. Broadcast receivers, unlike Activities, have no user interface. It's working is similar to that of a publish-subscribe design pattern. It's used for Asynchronous Inter-Process communication.

Layouts in Android UI Design

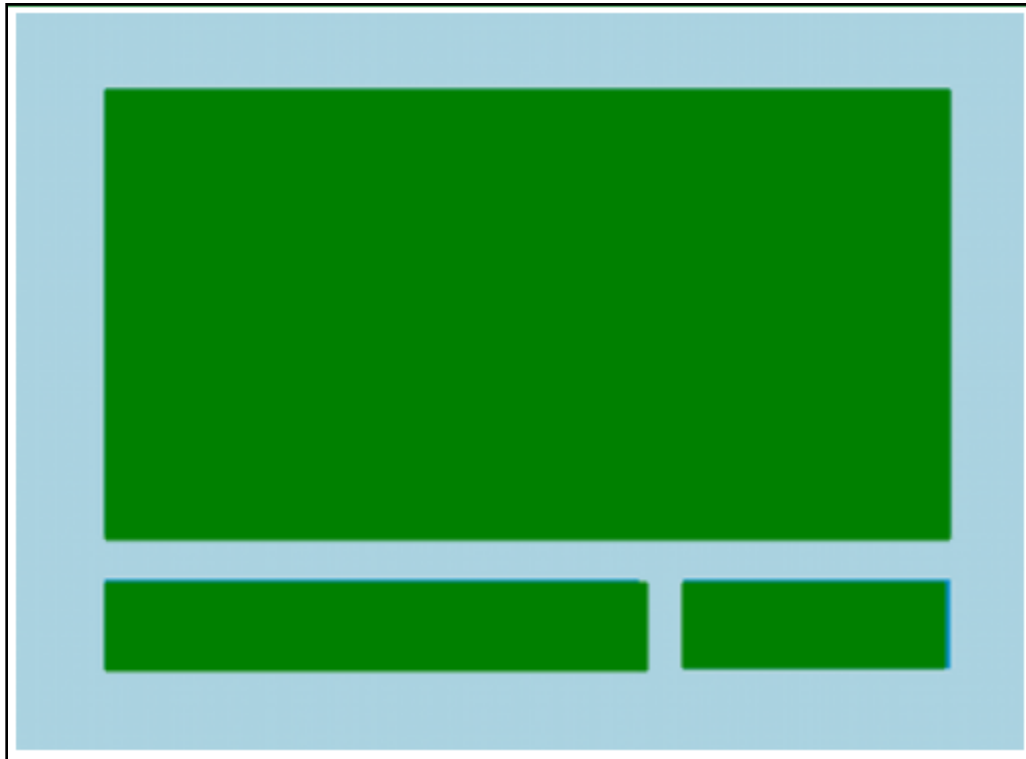
FrameLayout- It is the simplest of the Layout Managers that pins each child view within its frame. By default the position is the top-left corner, though the gravity attribute can be used to alter its locations. You can add multiple children stacks each new child on top of the one before, with each new View potentially obscuring the previous ones.



LinearLayout- A LinearLayout aligns each of the child View in either a vertical or a horizontal line. A vertical layout has a column of Views, whereas in a horizontal layout there is a row of Views. It supports a weight attribute for each child View that can control the relative size of each child View within the available space.



RelativeLayout- It is flexible than other native layouts as it lets us to define the position of each child View relative to the other views and the dimensions of the screen.



- **GridLayout-** It was introduced in Android 4.0 (API level 14), the Grid Layout used a rectangular grid of infinitely thin lines to lay out Views in a series of rows and columns. The Grid Layout is incredibly flexible and can be used to greatly simplify layouts and reduce or eliminate the complex nesting often required to construct UIs using the layouts described before.

Android UI Controls



1. TextView

TextView is a UI Component that displays the text to the user on their Display Screen.

2. EditText

EditText is a user interface control that allows the users to enter some text.

3. Button

This is a UI that is used to perform some action as soon as the user clicks on it.

4. ImageButton

It is the same as a Button but it's used to display an image on the button to perform an Action. In this, we need to give the source of the image so that the system can load it.

5. ToggleButton

The toggle button displays the ON/OFF states of a button with a light indicator.

6. RadioButton

Radio button in Android is the one that has only two possible states, that are either checked or unchecked. Initially, it is in the unchecked state, once it's checked it can't be unchecked.

7. RadioGroup

It's a group of Radio buttons that are alike. In this, only one of all the buttons can be chosen.

8. CheckBox

A CheckBox is the UI control that has two states that are either checked or unchecked. If we have a group of CheckBox, we can select as many as we want, unlike RadioGroup.

9. ProgressBar

In Android, we have a progress bar that shows the progress of some action that is happening like pasting a file to some location. A progress bar can be in two modes:

1. Spinner

The Spinner in Android is a User Interface that is used to select a particular option from a list of given options. Spinner in Android is the same as dropdown in [HTML](#). It provides us with a faster way to select an option of our choice. When we click on the down arrow, it shows a list of values from which we can select one. By default, the first value would be shown as the currently selected Value.

2. TimePicker

Time picker is a UI component that works as an intermediate to select a time of the day. The time chosen by it is shown either in 24 hrs format or in 12hrs format using AM and PM.

It gives a virtual Clock/watch to select it. This virtual clock makes it easy to choose the time.

3. DatePicker

Like we have time picker, we have a date picker as UI control too. In this, the System shows a virtual calendar to the users to choose the day.

This enables the user to choose a particular date using either a calendar or a dropdown. These both are made to make it easier for the user to pick up a date and a time.

4. SeekBar

In Android, Seekbar is an extended Progress bar. A seekbar comes with a pointer that is draggable throughout the bar either in left or right. This pointer helps to set the progress as well. This helps the user to choose a particular range of values.

5. RatingBar

A rating bar in Android is an extended version of a seekbar. It is used to give the rating by touching it. In the rating bar, a user can rate at a scale of 5 with a difference of 0.5. Its rating is in Stars. The user needs to tap/click the stars.

6. AlertDialog

Alert Dialog Box is a UI that gives the users an Alert or Warning of something. It appears on the screen in a small window. Once it comes, the user needs to decide or choose an option that it shows.

For example, when you enter the wrong password for email id.

7. Switch

In Android, a switch is a two-state UI element that holds either ON or OFF state. ON generally means Yes and OFF means No. By default, a switch is in the OFF state. A user can change its state many times.

8. AutoCompleteTextView

AutoCompleteTextView is an extension of EditText. In this UI element, the user is provided with a few suggestions of some values/texts. The value can be chosen by the user while filling AutoCompleteTextView.

IASRI-IT Applications for NARES

Dr. Sudeep

ICAR-Indian Agricultural Statistics Research Institute, New Delhi - 110 012

sudeep@icar.gov.in

The National Agricultural Research and Education and Extension System (NAREES) of India is one of the largest agricultural education systems with 4 Deemed-to-be Universities, 3 Central Agricultural Universities, 4 Central Universities with Agricultural Faculties, 65 State Agricultural Universities (SAUs), 114 institutions (including Agricultural Technology Application Research Institute- ATARI) and 731 Krishi Vigyan Kendra (KVK) under the aegis of Indian Council of Agricultural Research. These universities and institutions offer a wide range of courses, including UG, PG, and Doctoral programs in agriculture, horticulture, animal husbandry, fisheries, and other related fields and also conduct research in agricultural sciences, provide extension services to farmers and other stakeholders.

The Indian Council of Agricultural Research (ICAR), established in 1929, is the apex body responsible for coordinating and promoting agricultural education and research in India. ICAR plays a pivotal role in formulating policies, setting standards, and developing curricula to enhance agricultural education across the country. ICAR's mission is to "promote agricultural research, education, and extension for sustainable development." It coordinates with various agricultural universities and institutions to ensure the dissemination of quality education to aspiring agricultural professionals.

The Changing World of Education and the Need for New Initiatives in Agricultural Education

Education, as a whole, has undergone a significant transformation in recent years. The integration of technology, interactive learning platforms, and real-world experiences are enhancing the effectiveness of education, promoting critical thinking, problem-solving skills, and innovation.

The evolution of digital education has been a remarkable journey. From its humble beginnings as a mere concept, it has rapidly transformed into a global phenomenon that has revolutionized the way knowledge is imparted and acquired.

Recognizing the pressing need for transformation, agricultural education has embraced the digital revolution fascinatingly.

Embracing Change: ICAR's Commitment to Innovation

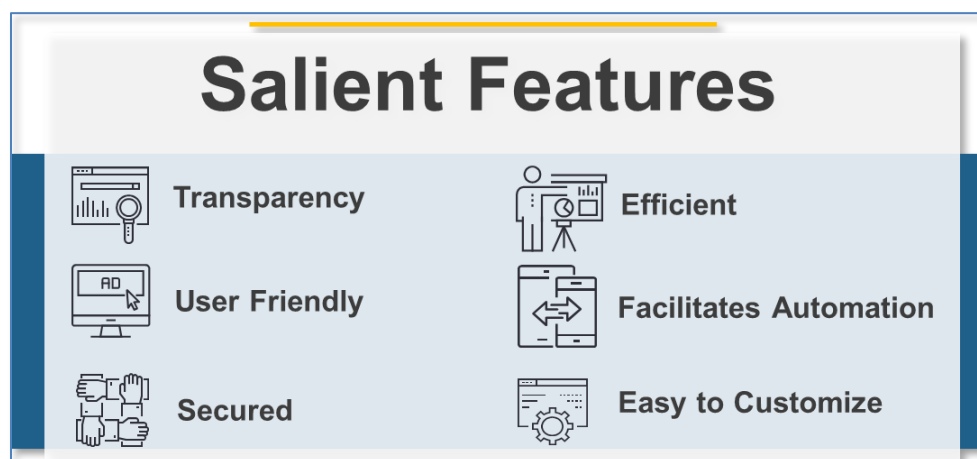
Recognizing the potential of strategic interventions in revolutionizing agricultural education, ICAR has

embraced the digital revolution in education. This commitment stemmed from the understanding that strategic interventions aided by technology can bridge gaps, foster inclusive education, and enable the efficient dissemination of knowledge existing in agricultural education. By leveraging digital tools and platforms, ICAR aims to ensure that agricultural education remains relevant, dynamic, and resilient on the face of adversities. To keep the system dynamic and resilient, ICAR launched RAES (Resilient Agricultural Education System) under NAHEP (National Agricultural Higher Education Project) which is a robust three-tiered digital framework that has been put in place to strengthen digital infrastructure, enhance digital capacity and create robust, relevant digital content for system wide consumption. Through this initiative, ICAR is paving the way for a more inclusive and transformative learning experience for students and educators.

Digital Infrastructure

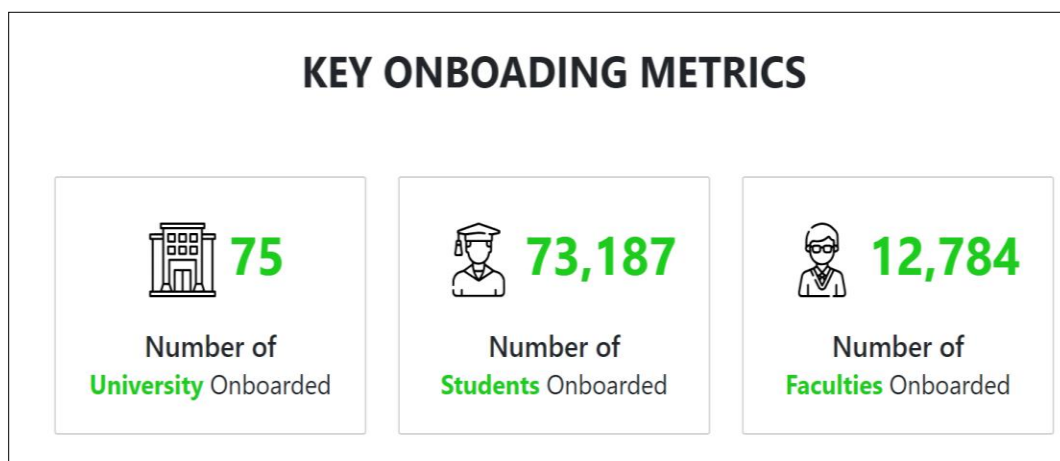
Digital infrastructure is revolutionizing the education system in the field of agricultural education by providing a platform for transformative learning experiences. Digital infrastructure is the key enabler and foundation of Resilient Agriculture Education System (RAES) to achieve the end goal of providing an exceptional learning experience to various stakeholders through enabling a robust digital eco-system. Under digital infrastructure, ICAR has launched various initiative:

Academic Management System (AMS): The Academic Management System is an integrated digital platform that streamlines administrative processes, student management, and facilitates effective communication between universities, colleges, and ICAR. Agricultural Universities and Institutions have not only welcomed this initiative but have also successfully completed their day-to-day tasks with it. A total of 63 universities are using the AMS system for their regular day to day administrative tasks. This also includes the 417 Registered Institutes/Colleges, 62,784 registered students, 7911 Registered faculty members under the universities.



Krishi Megh: The Krishi Megh was launched to meet the growing IT needs of the NARES system with the employment of applications such as e-Office, ICAR-ERP, Education Alumni Portal, e-Courses etc. More than 80 applications and websites are hosted on the Krishi Megh, which acts as a digital backbone for digital infrastructure.

Blended Learning Platform (BLP): The Blended Learning Platform facilitates a hybrid learning approach. It combines traditional classroom instruction with online modules and resources, allowing students to learn at their own pace, access quality content, and engage in interactive learning experiences. The platform is being implemented in all agricultural universities to equip students with the necessary knowledge and competencies to excel in the agricultural sector while promoting innovation and preparing them for the challenges of the digital age.



Agricultural Education Portal: The Agricultural Education Portal provides students, educators, and researchers with access to a wide range of resources, including information about agricultural universities, institutes, schemes, digital initiatives, e-learning modules and collaborative tools related to agricultural education.

ICAR has taken a holistic approach in developing educational initiatives that encompass various aspects. These initiatives include organizing hackathons through the Kritagya Hackathon Portal, establishing a network for alumni engagement known as KVC ALNET, and implementing AI-based disease identification in crops through AI-DISC. Alongside these efforts, ICAR's digital infrastructure initiatives are focused on providing stakeholders with the finest digital facilities available through effective utilization of digital resources. This approach ensures that the goal of delivering top-notch digital services to users of the portal is successfully achieved. It covers 3 Central AUs, 4 ICAR deemed universities, 63 State AUs, 4 Central University with Agriculture Faculty, 2,48,443 Unique Student ID generated.

Digital Content

ICAR has taken the lead in introducing numerous pioneering initiatives aimed at elevating agricultural education and establishing an enriching learning atmosphere for students. These ICAR initiatives, falling under the umbrella of digital content, are strategically designed to harness the power of technology, foster collaborative partnerships, and grant access to state-of-the-art resources.

Some of the key initiatives under Digital Content by ICAR include:

E-Learning Portal: The E-Learning Portal provides students and faculty with access to a diverse range of online courses aligned with the agricultural curriculum. The dynamic system of the portal ensures high quality content creation by the renowned faculty of SAUs and deemed universities. The platform is becoming popular amongst students as they are accessing content online. The institution offers 74 postgraduate courses that have been downloaded 33,923 times, and 163 undergraduate courses that have been downloaded 41,432 times.

Agri-DIKSHA: The Agri-DIKSHA is an online repository of agricultural educational resources. Under Agri-Diksha, Lectures are delivered across multiple disciplines through video recordings, video repositories, interactive and personalized adaptive learning, online assessments etc. The platform showcases input from 74 contributing universities, consisting of 2,575 resources in the public library section. Additionally, a total of 7,108 videos have been created, amounting to a total duration of 5,067 hours.

E-Krishi Shiksha: The e-Krishi Shiksha portal, is a digital platform dedicated to agricultural education. It provides online courses, video lectures, e-books, and interactive learning resources to students and educators. The portal has successfully facilitated access to quality agricultural education, reaching a wide audience across the country and abroad.

Virtual Reality Experience Labs: ICAR has established Virtual Reality Experience Labs to provide immersive learning experiences. VR Labs are the integration of immersive technology with research and education. A total of 74 Virtual Reality Experience Labs has been established in the country by ICAR with about 2066 students have used it since its launch.

A-Krishi-Agricultural Knowledge Resources and Information system hub for Innovations: ICAR Research Data Repository for Knowledge Management

Dr. Anshu Srivastav

ICAR-Indian Agricultural Statistics Research Institute, New Delhi - 110 012

Anshu.Bharadwaj@icar.gov.in

Introduction

In an era of digital world, it has been realized that digital preservation is most important for effectively using the information created particularly in the digital form. Digital preservation means securing long-term storage of information in digital form. The same is also applicable to research outputs in physical form such as seeds, culture, etc. There are several challenges in digital preservations as it is expected to preserve them for long period, even beyond the data creating organizations. Moreover, the ever evolving technological changes (for e.g. there is no guarantee that a file stored in a specific format can be read in future as there are no compatible hardware/software available) and fragility of storage medium make digital preservation further complicated. To address this issue, Consultative Committee for Space Data Systems (CSDS) has proposed a reference model called “Open Archival Information System (OAIS)” which has been accepted as a standard by International Standard Organization (ISO) – ISO 14721:2012. The original model was developed by CSDS to preserve Space data but it is more general to cater to the needs of other subjects and now it has been accepted as a reference model by many organizations across the world for creating knowledge repositories.

The modern research in all disciplines has become increasingly dependent on strong collaboration, sharing of resources and information flow. This is particularly true in the case of Agricultural Research. Indian Council of Agricultural Research (ICAR) is the major source of funding for all agricultural research being carried out in the country. It is therefore important to develop an Agricultural Knowledge Repository which can improve and manage Agricultural Research in the country.

ICT in agriculture is an emerging field focusing on the enhancement of agricultural and rural development. It involves the conceptualization, design, development, evaluation, and application of innovative ways to disseminate the agricultural information in the rural domain, with a primary focus on agriculture. The main activities of the agriculture are crop cultivation, water management, fertilizer application, pest management, harvesting, post-harvest handling etc. All stakeholders involved in agricultural development

activities need information and knowledge about these activities to manage them efficiently. Any system applied for getting information and knowledge for making decisions in any organization should deliver accurate, complete, concise information in time or on time. The information provided by the system must be in user-friendly form, easy to access, cost-effective and well protected from unauthorized accesses.

The stored information can be quickly extracted by a large number of users simultaneously. IT based Knowledge System provide new opportunities to improve the livelihood of the rural mass. In this proposed system, technology database, research data repository, Geo-portal will be developed. The evolved system would be helpful to farmers, researcher, extension workers and stakeholders.

Further, the demand for web based spatial data applications and services are increasing rapidly. The availability of accurate and timely information enables and rationalizes the decision-making processes of planners and managers in a cost effective and time efficient manner. Availability of spatial datasets in standard and readily useable format in Geoportal helps in developing the plans and policies for optimum utilization and management of agricultural resources in the country. A wide area of web-based applications initiated the requirement to disseminate spatial data to the end-users by the use of Geoportals. Geoportal will act as a knowledge gateway to visualize, access, query and disseminate information on agricultural resources for the users. Development of systematic databases in GIS framework eliminate redundancies, duplication of efforts and enforce consistency, standards, sharable protocols to build a cross-domain knowledge base for effective utilization of limited natural resources in the country.

A lot of efforts and ICT initiatives have been made in Indian Council of Agricultural Research in the past.

Background and Motivation

It is a well-known fact that scientific research depends on collaboration, sharing of resources and free flow of information. This has been recognized worldwide and data repositories have been planned and are being created. The research work carried out by all the Institutes under the aegis of Indian Council of Agricultural Research (ICAR) as well as in State Agricultural University in the area of Agriculture and Allied sciences are funded directly or indirectly by ICAR/Government of India. Thus, it is imperative to have a central data repository to hold all the data generated in Agricultural Research. Such a data repository with proper meta-data tagging, sharing would surely improve the output of Agricultural Research.

Realizing the importance of data management, the committee constituted for data management in ICAR institutes. The Governing Body of Indian Council of Agricultural Research (ICAR) has approved the guidelines for internal evaluation and forwarding research papers to scientific journals and data management in ICAR Institutes on 12th March 2014. The guidelines provide directions to achieve the

highest standards and practices in publishing and managing data in ICAR institutes. It is also expected that these guidelines would help the researchers and research managers in ensuring reliability of published results, proper credit sharing and safety of data for future use.

The committee which delved into the issues of research data management, while making the guidelines, has recommended ICAR to take up creation of a centralized data repository to streamline the research data management in ICAR institutes. The committee has given guiding principles for Data Management in ICAR. These guidelines have been prepared under following headings.

- ✓ Definition of Data
- ✓ Data Ownership
- ✓ Data Collection and Recording
- ✓ Data Storage and Security
- ✓ Data Access and Sharing
- ✓ Data Retention
- ✓ General Recommendations

The guidelines are a comprehensive set of standards for effective management of data in the Institutes and are intended to increase awareness of maintaining data integrity.

The committee has also suggested to implement the recommendations in a phased manner possibly to start with one AICRP/Institute from each subject matter division in the first phase may be taken up and the other institutes may be covered in the subsequent phases. This activity is a way forward to operationalize the committee's recommendation of establishing data repository.

Implementing Data Management in ICAR

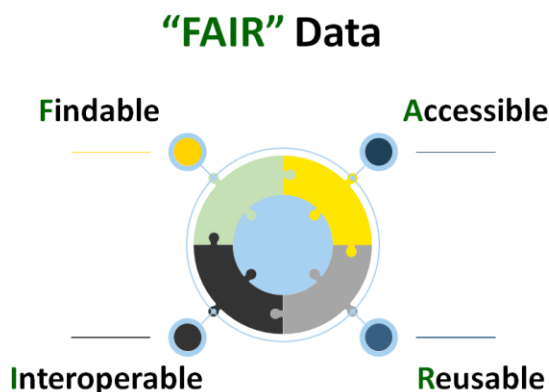
The committee which delved on the issues of research data management, while making the guidelines, has recommended ICAR to take up creation of a centralized data repository to streamline the research data management in ICAR institutes. As per these guidelines of Data Management Policy in ICAR Institutes, the major emphasis needs to be laid on Creation of Research Data Repositories of Actual Trials/experimental level data and Standardization of Analysis. Based on this digitized data, an Information System for Planning and Analysis of Experiments would be developed, and data may be shared among scientists as per the data sharing policy. Technology Repository; Geo-Portal; Publication Repository; Observational Data Repository; etc. added in the scope of the work as per Action Points for

DARE/ICAR from Cabinet Secretariat, Point No. 5(v).

The implementation task has been entrusted under the chairmanship of DDG(NRM) with DDG(Engineering) as Co-Chair. Other Subject Matter Divisions of ICAR and nodal officers were identified for coordination. A core team of scientists from ICAR-IASRI, New Delhi; ICAR-NAARM, Hyderabad; ICAR-NBSS&LUP, Nagpur; ICAR-DKMA, New Delhi and ICAR- IARI, New Delhi are at present involved in these activities and has started working on creating basic framework and identify ways and means of implementing the data management policy in ICAR. Accordingly, a portal named as **Agricultural KRISHI-Knowledge Based Resources Information Systems Hub for Innovations in Agriculture** was initiated as ICAR Research Data Repository for Knowledge Management and is available at <http://krishi.icar.gov.in>. This portal would consist of six main repositories viz. (i) Technology Repository; (ii) Unit Level Experimental Data Repository; (iii) Survey Data Repository; (iv) Observational Data Repository; (v) Publication Repository and (iv) ICAR Geo-portal. Other data repositories that will be included in this portal are Learning Resources Repository, Historical Repository; etc.

ICAR Research Data Management aims at:

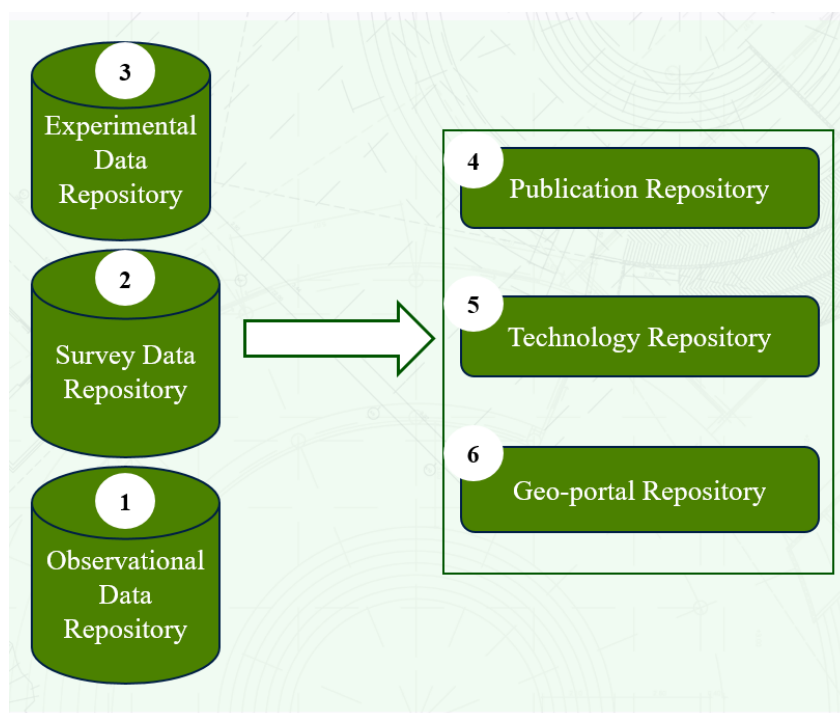
- Bring data/outputs at a centralized place for single window access
- Principle of **FAIR**
- Transparent – provide information on data availability to all.
- Open - legitimate purposes
- Promote Open Data/sharing data Culture among researchers.



5. Objectives

- i) To develop knowledge repositories related to proven technologies, and publications created by ICAR/NARS institutes.
- ii) To develop research data repositories and standardization of analysis of unit level experimental/survey data including observational data.
- iii) To create an Agricultural Geo-portal for strengthening spatial data analysis.
- iv) To develop an Agricultural Knowledge Portal to provide access and manage knowledge repositories and create tools for efficient data management and statistical data analysis.

The following six knowledge repositories have been created.



- **Technology Repository** consists of technologies generated by ICAR institutions in the areas of Crop Improvement, Natural Resource Management, Fisheries, Veterinary, Dairy, Animal sciences, Horticulture, Engineering and Social Sciences. The technologies include only those which were developed and tested. Information is also available on commercialized technologies. The repository provides an easy way to search and locate appropriate technologies based on regional, subjects, institutions, and related resources such as reports, success stories, media resources/contents, etc. This repository is made available under the aegis of open access policy of ICAR for viewing. Additions/modifications to the items in the repository are restricted only to authorized persons.
- **Publication Repository** consists of available all types of publications including reports, bulletins, research papers, web resources, media resources etc. from all ICAR institutions. However, due to IPR related issues, some of the resources may not be available in open access.

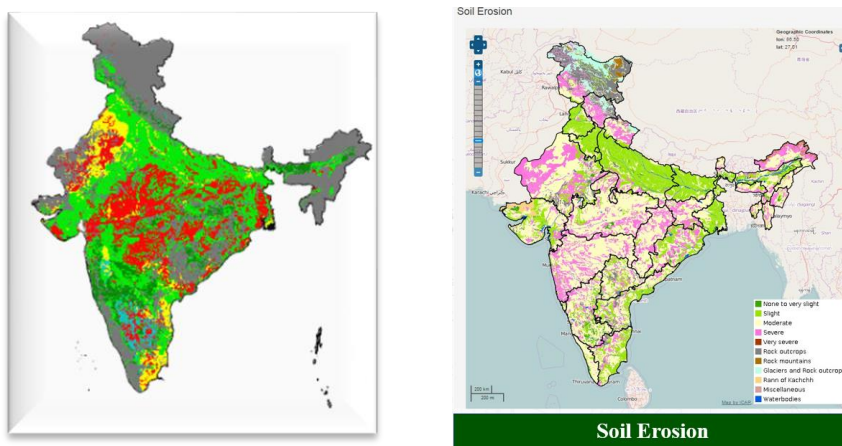
The following types of knowledge repositories have also been created.

- **Experimental Data Repository** consists of data generated through experiments conducted by researchers in different ICAR institutions. All experimental data consist of location and

identification details, subject level details, unit level data etc. Only limited information is available for general users. The access to this repository is restricted as per the ICAR data sharing policy.

- **Survey Data Repository** consists of data collected through surveys conducted by researchers in different ICAR institutions. All survey data consists of location and identification details, subject level details, unit level data etc. However, to protect the privacy of the respondents, personal details will not be made available. Only limited information is available for general users. The access to this repository is restricted as per the ICAR data sharing policy.
- **Observational Studies Data Repository** consists of data collected through observation studies including meteorology, land-use pattern collected by researchers in different ICAR institutions. Most of the information will be made available under the aegis of ICAR data sharing policy.

ICAR Geoportal: To standardize and develop metadata on agricultural resource data for design and development of ICAR Geoportal and applications, ICAR Geoportal has been developed.



Approach of Geoportal:

- Identify legacy datasets.
- Identify data formats including Metadata formats.
- Identify the available data models.
- Metadata development
- Standardize the datasets.
- Design and develop prototype Geoportal.
- Develop the protocols to store and process database.
- Develop modules for analyses, query and visualize the data.
- Develop data interoperable mechanisms.
- Develop applications and services in Geoportal.

Features of Geoportal:

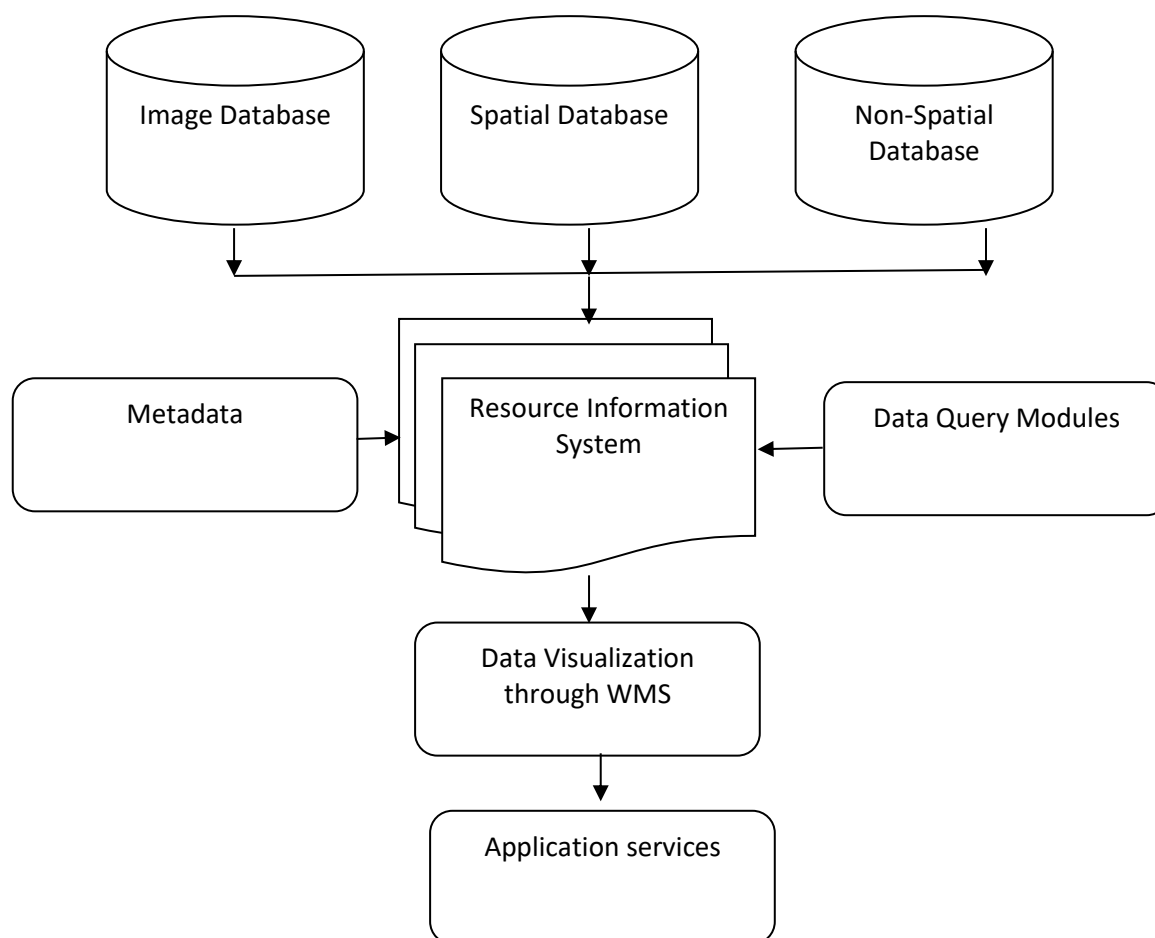


Fig. Simple design of Geoportal

An Agricultural Knowledge Portal to provide access and manage knowledge repositories and create tools for efficient data management and statistical data analysis has also been created.

Integration of Knowledge Repository and Geo-Portal

- ICAR-repository will need to interact with knowledge repository and geo-portal.
- Existing websites and web applications of ICAR institutes
- These applications/repositories use different platforms and technologies. Integration techniques should be used to integrate the knowledge repository and geo-portal.

Standardize Master Data

Master data should be standardized. ICAR repository Portal would comply with defined standard. Compliance and governance processes should check adherence to defined standards. Content publishing workflow should enforce usages right master data values as part of publishing process.

Standardize Content Publishing Workflow

Content publishing workflow should be standardized web enabled automated process. Definition of

metadata should be integrated into publishing workflow. Standardize Content Management System. Content management system should provide defined set of standard functionality.

Unique Content Identifier

All content should be assigned a unique ID. Content repository should have web based access. All published content should be accessible using a URI.

Centralized Metadata Repository

A Centralized Metadata Repository should be established, which will store metadata of all content published on ICAR repository Portal using an automated process.

Centralized Metadata Repository stores metadata in normalized form.

Metadata Updates

When new content gets published, its metadata should be propagated to Centralized Metadata Repository. Similarly metadata may get changed after publishing of content. In such case also changes should be propagated to Centralized Metadata Repository. Propagation of metadata should be managed using an automated process.

KRISHI Visibility

Visibility

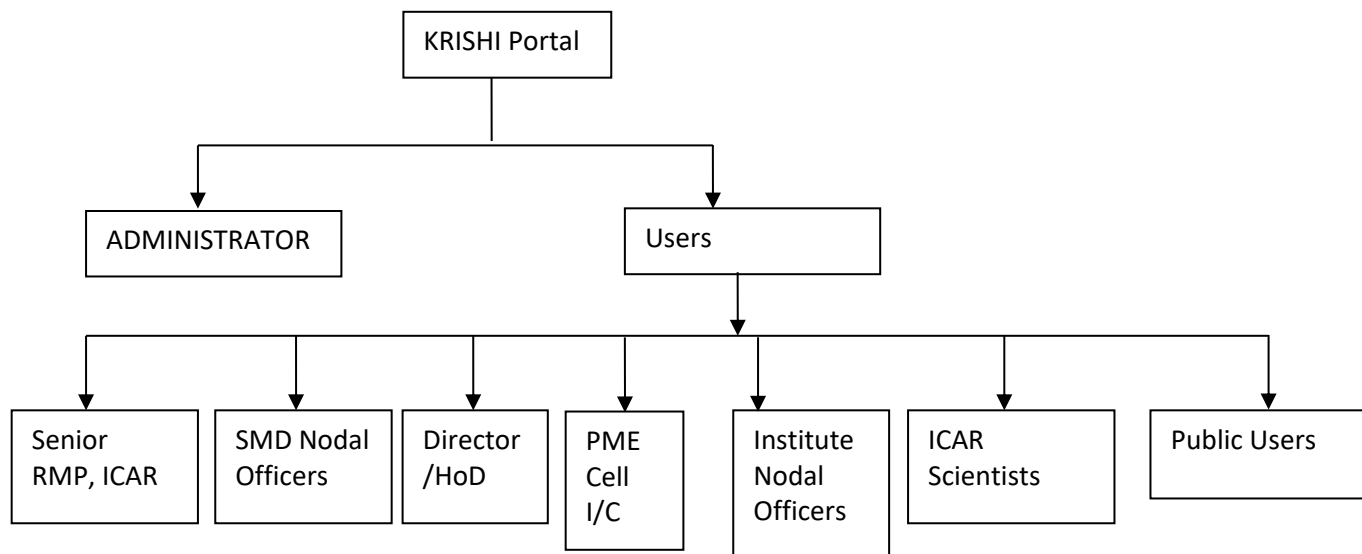
- ▶ **Indexing of KRISHI Publication and Data Inventory:** Google Scholar, **Base** (Bielefeld Academic Search Engine) and **OpenDOAR** (Directory of Open Access Repositories)
- ▶ **GFAR** (The Global Forum for Agricultural Research)
- ▶ Open Government Data Platform (**data.gov.in**) (Several datasets are shared through Webservices)

Integration with ICAR institutes Applications

General Interface for Transaction ICAR Services

ICAR-repository Portal would work as a single source of information for all ICAR institutes content and provide front end or start point for all web applications provided by ICAR institutes. It is required that functionality of web applications provided by the institutes is made accessible on ICAR repository Portal.

User Level



Administrator -

Administrator enables to access all portal pages. Administrator will have full control of the portal. It will monitor all the transactions on portal and generate all type of reports. Admin will assign the role to the users to create, edit, and upload the information on the portal.

The Authenticated-User

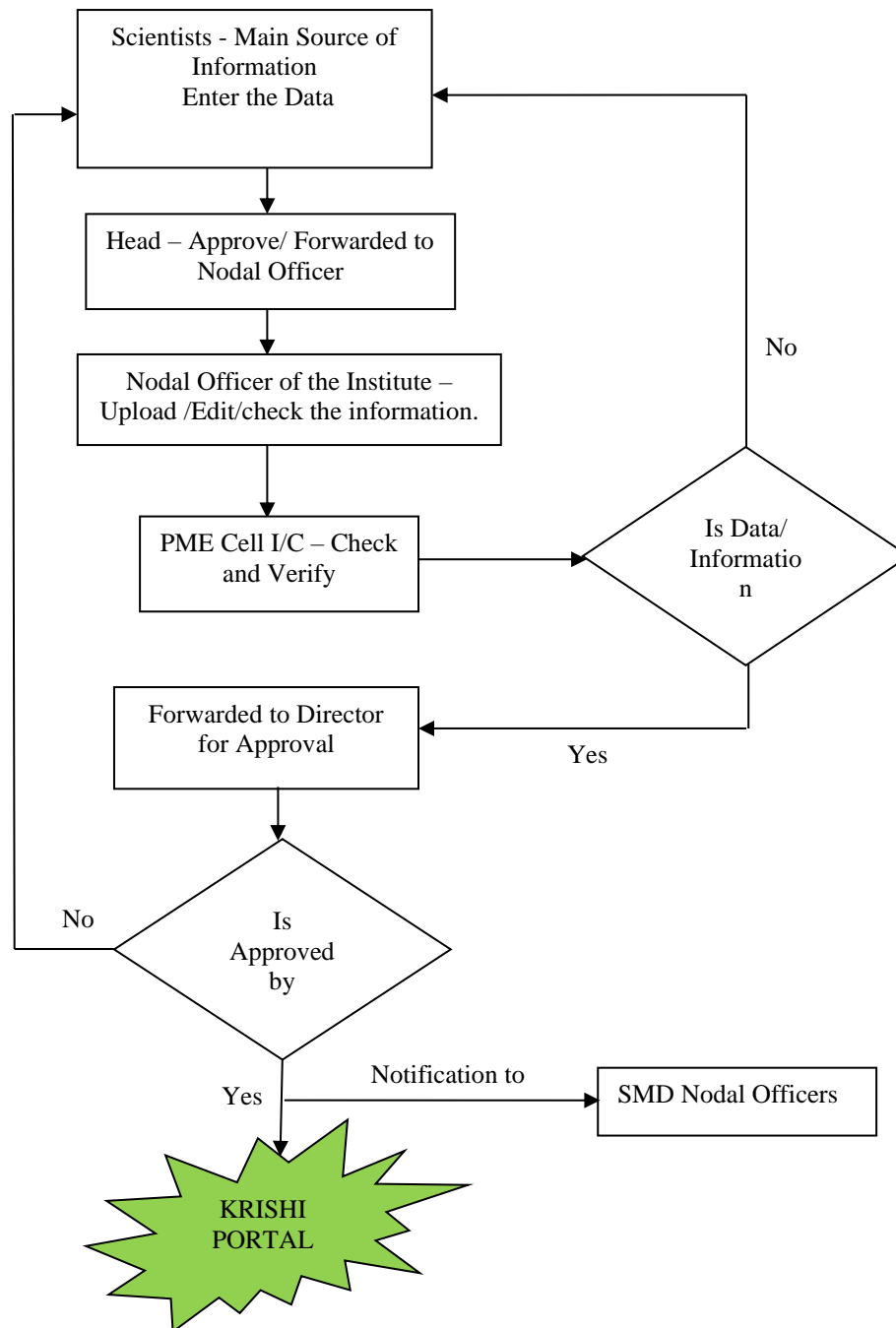
All ICAR personnel will be authenticated user and automatically assigned already MIS/FMS user id-password to view and upload the information.

- (i) ICAR Scientists will be the primarily data resource person. Assigned the role only to upload, edit and view data/Information.
- (ii) **Institute Nodal Officers** - The Authenticated-User role will be given to Institute Nodal Officer. Nodal Officer logged in and contributes/upload and verifies the research data and technology data.
- (iii) **PME Cell In charge** – Assigned the role to examine the data and reassign to the respective Institute Director for final approval.
- (iv) **Director** – Director will act as data controller of the respective Institute. Assigned the role to approve the data/information for final submission to the Krishi Portal. They will also customize the reports as desired by the respective institute. They will also be able to view the data of the other institutes.
- (v) **SMD Nodal Officer** – Assigned the role to monitor the transactions in respective SMDs with relation to uploading of data. They will also customize the reports as desired by the respective division.
- (vi) **DG, DDG and ADG will have the right to view data/Information and** also customize the reports as desired by the respective division.

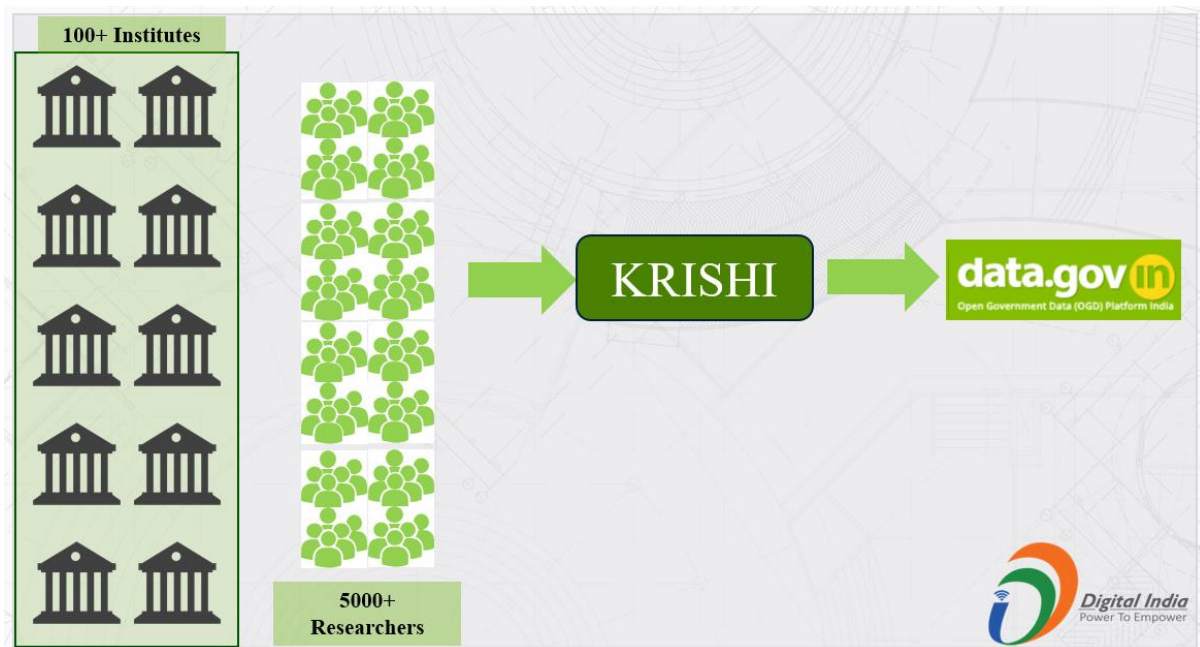
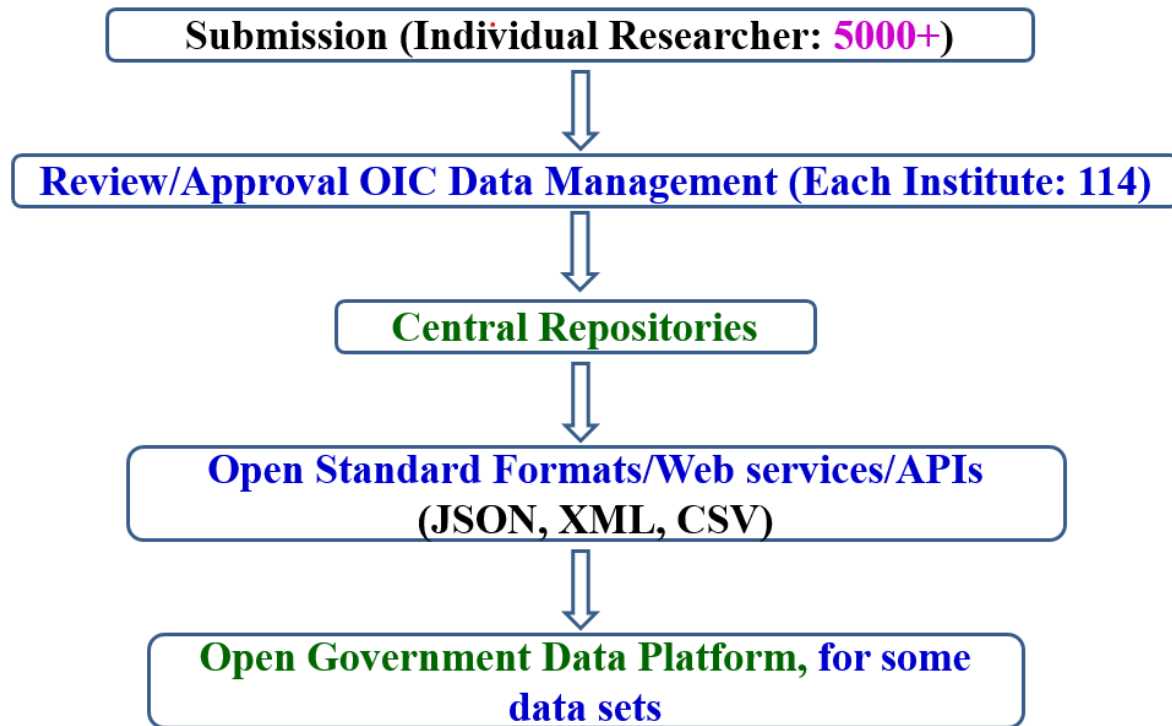
Public user

Any user with access to KRISHI Portal who is not registered as ICAR personnel is a public-user. These users will have access to the open-source information available on the portal.

Approval Mechanism of Data/Information



Further the data workflow can be understood by the following figure:



Web Portal Functions

The major functions of the ICAR Repository Portal are as listed below-

1. Content Management System – metadata, content repository (text, images, video, graphs)

2. Expert Advisory and Knowledge Management Database – Ask the expert and knowledge repository.
3. Identity and Access Management – User registration and access control
4. User database
5. Identity and Access Management for Registered and non-registered users, Internal users, External Users, etc.
6. Links for applications – ICAR institutes web applications/websites, agricultural technology, pesticides, seeds, soil health, crops, farm machinery, training and, forecasted weather and agro-met advisory, fisheries, Livestock Management.
7. Links for State Agricultural Portals
8. Collaboration Tools – blogs and discussion forum
9. Search and indexes
10. Directories
11. Frequently Asked Questions (FAQs)
12. Feedback
13. Helps

Features of Portal

The main focus of the ICAR repository will be disseminated the agricultural information among the farmers, extension workers, students and researchers. The main features of the proposed ICAR web portal are:

- (i) Single window access of agricultural knowledge for farmers and other stakeholders
- (ii) Graphical interface easy to use- design of the web pages for common look and feel. The user would be able to access the desired document in maximum 3 clicks.
- (iii) Easy to navigate, search and browse- Site map and breadcrumbs for navigation guidance.
- (iv) Interface for the updating of information and service delivery for ICAR institutes/AICRP
- (v) Multi-Lingual support – for farmers to access information in his own language.
- (vi) A Search feature for users to find the relevant information or service on portal.
- (vii) Extension training aids materials (Audios/Videos/ pictures)

Information Accessible for Public

The main purpose of the project has to disseminate agricultural knowledge to the farmers and general public. The technology may be kept under open access. But for keeping the record, who will be accessing

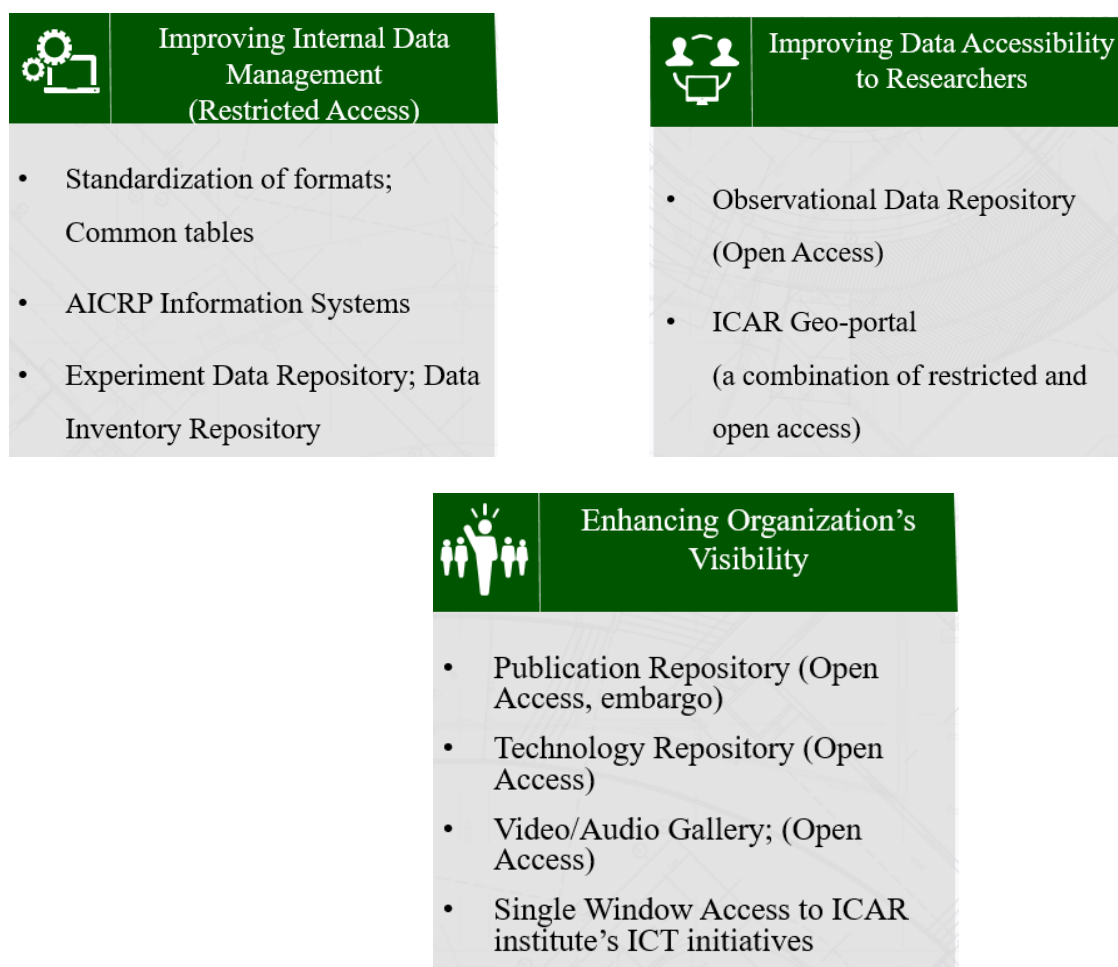
the system, a simple registration form is required for registration. User personnel information will not be shared for other purposes.

Information Accessible for Researchers

ICAR Repository project involves scientists, students and other stakeholders who will be accessing various resources of the project online. Secured access of these resources is of utmost important. At the same time users expects system to protect integrity and confidentiality of their identity information and ensure safety of their transaction. Hence ICAR-Repository needs a comprehensive Identity and Access Management (IAM) System that will provide secure access to its resources in integrated and secured manner and at the same protect privacy of the users.

User's identities are at the core of any operation. Anyone desires to retrieve the knowledge needs to prove his identity. Identities need to be managed to facilitate the right access to the right resources. Identity and Access Management of ICAR repository will provide consistent, efficient, and secure method to manage identities both internally and externally.

KRISHI works on the following :



- KRISHI is based on the philosophy to enable the interested users to find and use them either manually or obtain through web services.
- Mainly built on open-source standards, the KRISHI portal can interoperate with any other Information Systems.
- The portal also makes sure that intellectual and commercial interests of the council are safe-guarded.
- In the present form, KRISHI focuses on researchers both from within and outside ICAR system to make use of resources collected.

Cross Learning and Impact of KRISHI

- Has been awarded the Gold Icon Award in Open Data Championship Category for ICAR Research Data Management Initiative under Digital India Awards 2020 of MEITY, Govt. of India. Conferred by Honourable President of India Sh Ramnath Kovind ji on December 30, 2020
- First LDAP Authentication 2016: Paved the way for other systems in ICAR like E-office, ARMS, FVMS and other systems of the Council.
- Postgresql 2016: For ICAR Publication Repository ~ capacity building of handling Enterprise version of PostGreSql ~ has helped in faster implementation of E-office that also requires PostGreSql Enterprise version of database.
- KISAN 1.0 and KISAN 2.0: The data from ICAR Mobile App Gallery provided through Webservices.
- Research Papers Published: Several journals require making available data in open domain. At least three papers have been published where the URL of data availability has been shown as ICAR Data Inventory Repository
- MasterVocab and Webservices: Developed master vocabulary to be used across applications for interoperability and mechanism of sharing through webservices.

Various applications of KRISHI are:

Applications

- ▶ **At Research Level:** Identifying the genotypes/phenotypes suitable for biotic and abiotic stresses, management practices that are resilient to climate change and ensure sustainable production.
- ▶ **At Farm level:** Crop health and soil health analysis through remote sensing and IoT, farmer advisory services for input control and market demand prediction - Precision farming.
- ▶ Mega Environment Mapping of Crops
- ▶ **Location specific advisories:** Generation of sowing schedules and contingency plans; Predicting Phenology and suggest agronomic measures; Crop-specific soil test based fertilizer recommendation.
- ▶ Agro-met advisory
- ▶ **Policy:** Crop Planning... *What to grow?*

AI-DISC: Artificial Intelligence Based Disease Identification System for Crops

Chandan Kumar Deb

ICAR-Indian Agricultural Statistics Research Institute, New Delhi - 110 012

Chandan.deb@icar.gov.in

What is AI-DISC ?

- **AI-DISC** (Artificial Intelligence based Disease Identification System for Crops) is an AI-enabled android mobile application for automatic identification of diseases through image
- Developed under **NAHEP Component 2** and **NASF Project** (Artificial Intelligence based mobile app for identification and advisory of maize diseases and Insect Pests)
- Images and advisories have been collected from **ICAR-IIMR**, **ICAR-IARI** and **11 SAUs**
- Imagebase and Knowledgebase has been maintained in **NIBPP** (National Image Base for Plant Protection)
- Deployed and Hosted on **Krishi-Megh** Cloud Infrastructure
- Provides real time crop protection solution using **AI** and **Image-processing**
- Expert forum of AI-DISC facilitates Expert-Users interaction through chat for complex multifactor plant protection problems

Simple Steps to Identify Crop Diseases

- Download AI-DISC android mobile app
(https://play.google.com/store/apps/details?id=com.ai.ai_disc)
- Upload images with visible symptoms
- Get the disease and advisory automatically

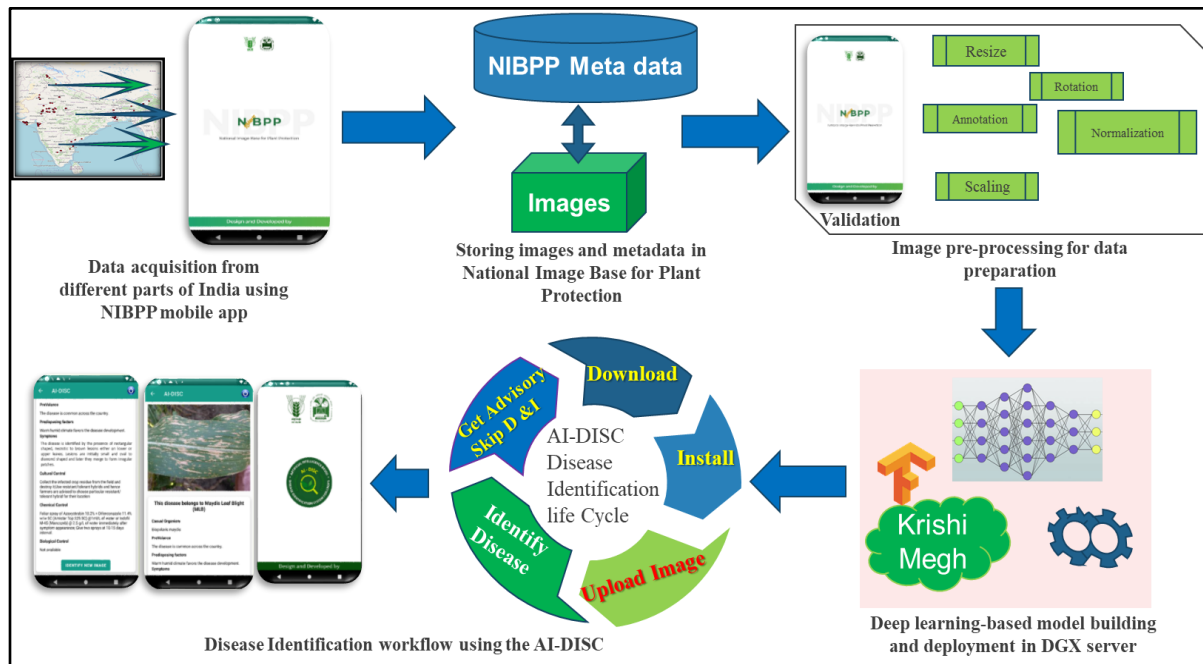


Figure: Developmental journey of AI-DISC

Models

- Deep learning models over 19 crops (Rice, Wheat, Maize, Tomato, Mustard, Cotton, Brinjal, Apple, Peach, Kinnow, Mandarin, Assam Lemon, Chickpea, Green gram, Cluster bean, Moth bean, Chilli, Coriander etc.)
- Trained over 1.5 lakh images
- Developed and deployed in NVIDIA GPU server

Features

- AI-enabled disease identification within fraction of seconds
- Expert consultations facility via text/video chat

Types of Accounts in AI-DISC

Multiple Type of users with different privileges

Extension Worker/ Farmer

Image based disease identification

Report location wise disease occurrence

Domain Expert Account

Provide domain advisory to the users through chat and video call

Administrator

Overall control through NIBPP

- Real time reporting system for disease infestation across India

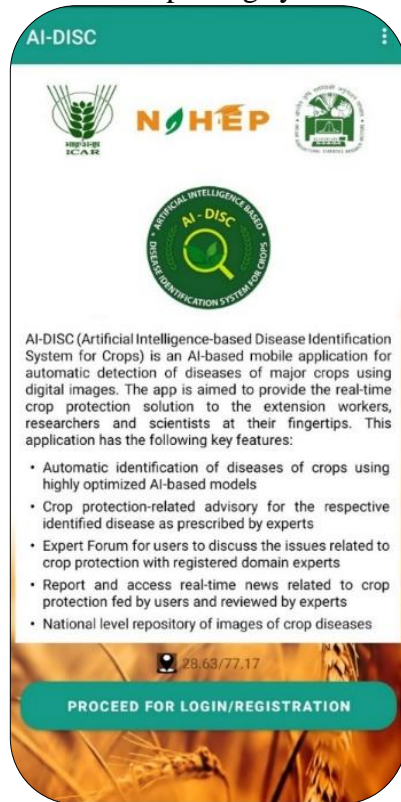


Figure: AI-DISC Mobile Home Screen

Facilities available in AI-DISC: Identification

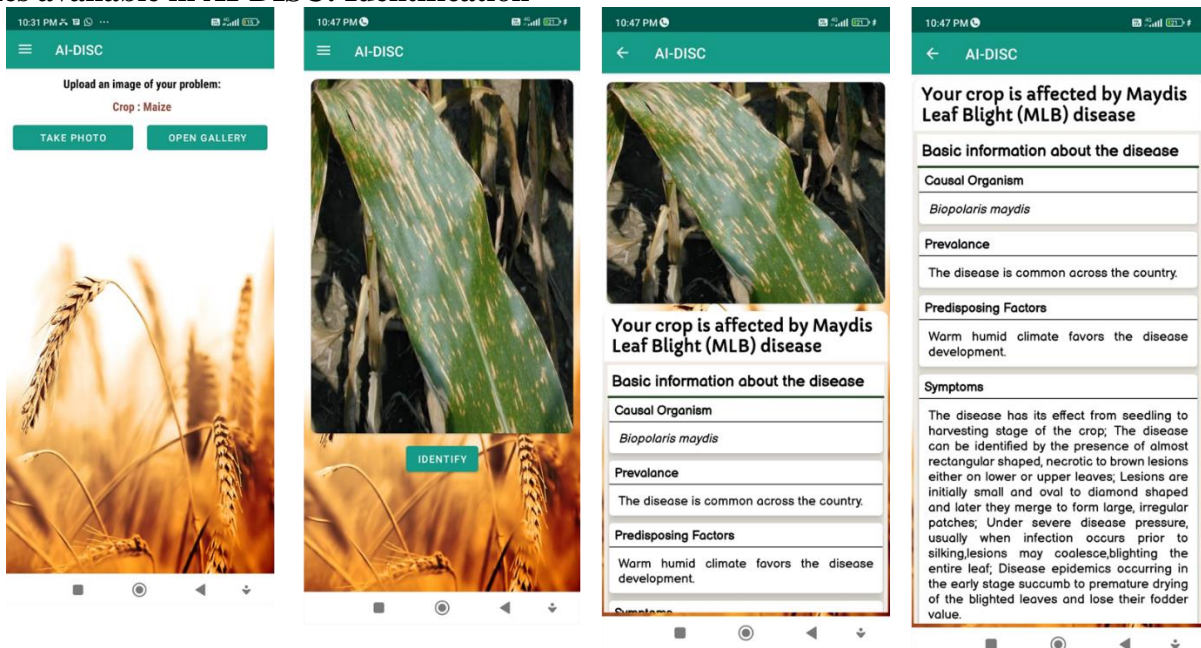


Figure: Identification Module of AI-DISC

Facilities available in AI-DISC: Reporting

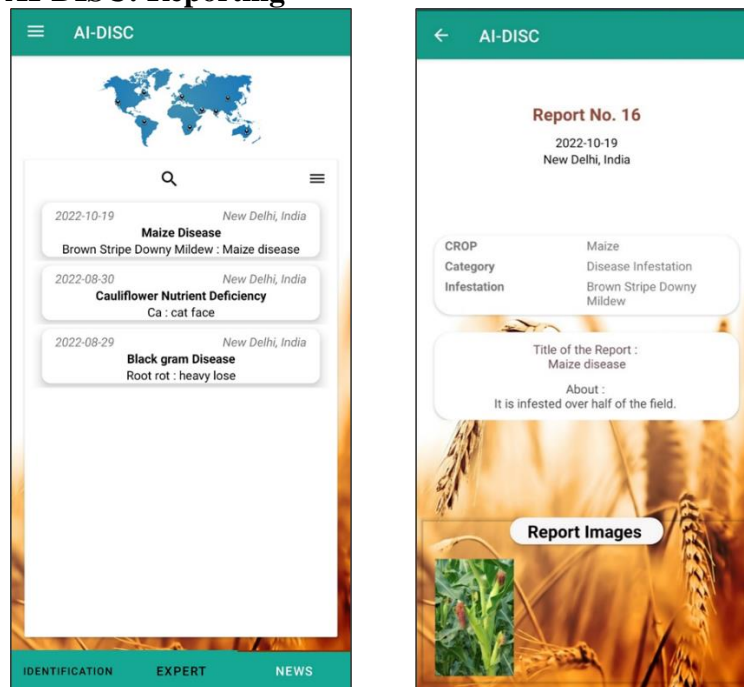


Figure: Reporting Module of AI-DISC

Report

Location wise report of plant diseases and related information along with images

Facilities available in AI-DISC: Expert Forum

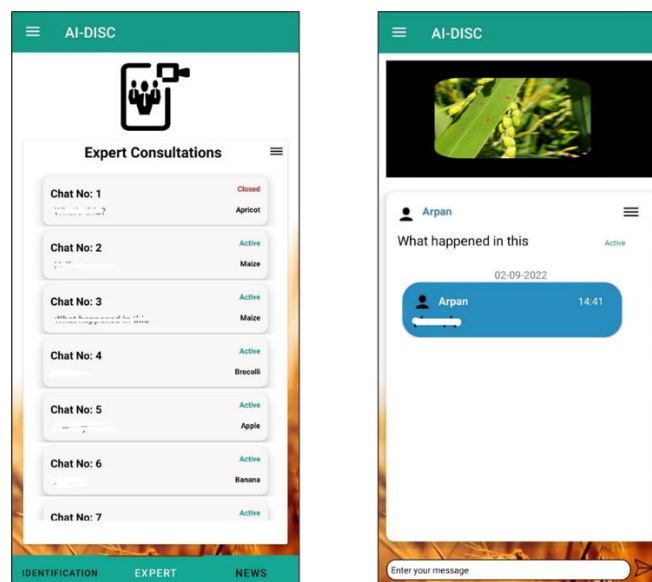


Figure: Expert forum of AI-DISC

Expert Forum

Query facility for plant protection problems

Expert-User Chat

Technology Used

App Development

Java

Android SDK

MS SQL Server

Model Programming

Python 3.6 and above

Packages

Tensorflow, Keras, Scikit-learn, Numpy, Pandas, Matplotlib

Model Train and Deploy

NVIDIA DGX GPU Clusters

Configuration

System: **NVidia DGX Server**

Operating system: **Ubuntu 18.04.3 LTS**

CPU processor: **Intel(R) Xeon(R) CPU E5-2698 v4 @ 2.20 GHz**

Graphics processor unit (GPU): **Tesla V100-SXM2- 32 GB**

RAM: **528 GB**

Deep learning framework: **PyTorch, TensorFlow**

Deep learning environment: **Jupyter Notebook**

Programming language: **Python**

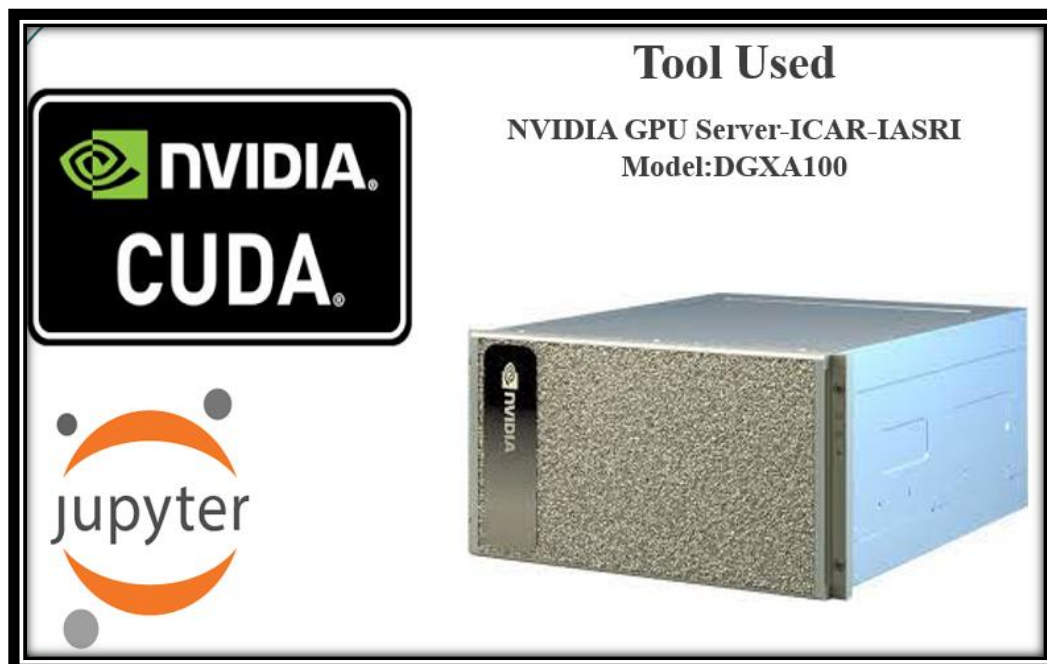


Figure: Tool Used for model development

National Agricultural Research and Education System- Blended Learning Platform (NARES-BLP)

Dr. Sapna Nigam and Dr Sudeep Marwaha

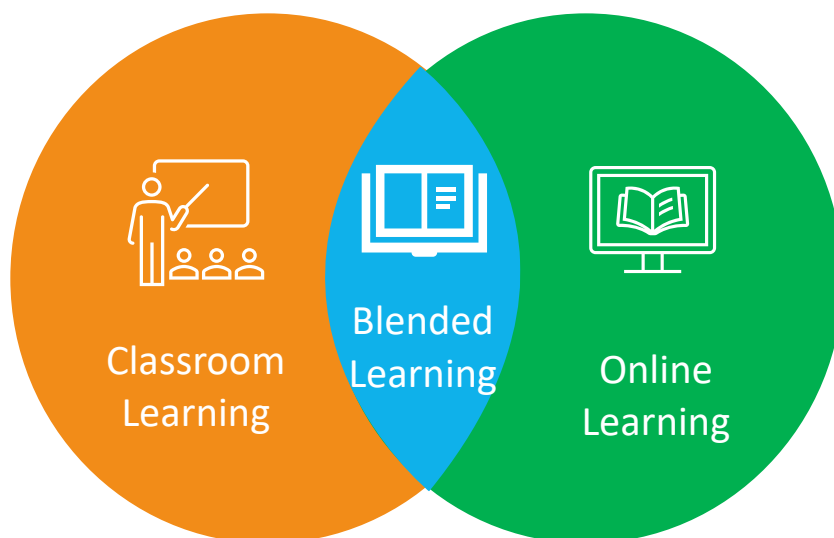
ICAR-Indian Agricultural Statistics Research Institute, New Delhi - 110 012

Sapna.nigam@icar.gov.in

Resilient Agricultural Education System (RAES) is a development initiative under NAHEP undertaken to strengthen the Digital Infrastructure of Agricultural Universities, enhance the quality of Digital Content in the field of Agriculture, and facilitate Digital Capacity Building to address the digital divide among stakeholders. A blended Learning Ecosystem provides centralization of all the amenities that learners and administrators require for knowledge retention and increased engagement. Its key differentiator is some top-of-the-line functionalities such as in-built Video Conferencing and Digital Whiteboard for collaboration, Instant Messaging, eLearning, and Online Assessment tool.

1. What is Blended Learning?

Blended learning integrates computer-assisted online activities with traditional face-to-face teaching methodology (Chalk & Talk)



2. Why Blended Learning?

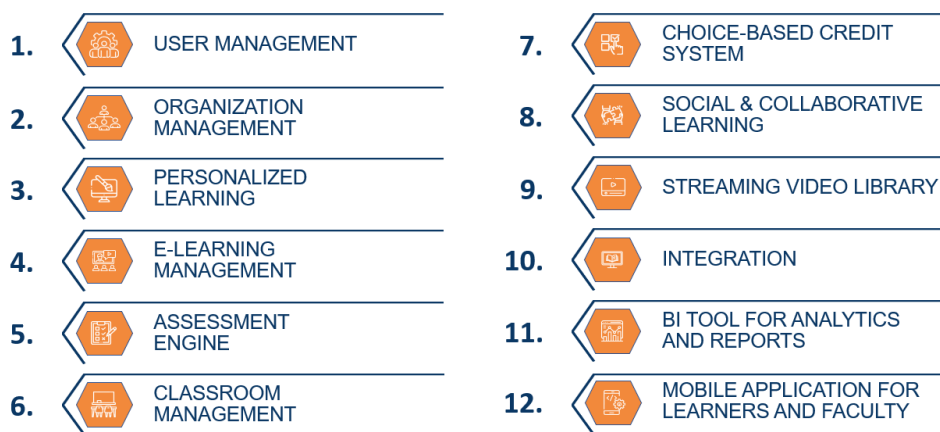
- Enables effective self-paced learning through flipped classroom model.

- Facilitates sustainable rotation model-learners and instructor can transition smoothly between self-paced learning and face-to-face classroom environment.
- Introduces learners to the personalized learning experience with customized learning paths to improve their academic outcomes.

3. Introducing NARES – BLP (Blended Learning Platform)

- Centralized teaching solution that is aimed at enriching the teaching-learning and monitoring process across agricultural universities in India.
- A platform where faculty can upload course content, and learners can access it
- Supports both face-to-face and virtual learning.
- Facilitates communication, tracks progress, and keeps records of all important data of students.
- Provides omni-channel learning experience by enabling uninterrupted user experience on all digital devices.

4. Key Elements of NARES - BLP



5. Some of the Use Cases of NARES - BLP

Delivery of Undergraduate and Postgraduate (PG) courses through Blended Learning Platforms (BLP) for all Agriculture Universities and their constituent colleges can be facilitated in the following ways:

- Faculty can utilize the BLP to enhance regular teaching by uploading video lectures, assignments, lecture notes, PowerPoint slides, etc. Students can submit their assignments on the BLP, faculty members can also conduct online or computer-based examinations.

- The University can utilize the BLP to extend teaching to students across multiple colleges, with a single senior faculty member overseeing the class while junior faculty or teaching associates manage the sessions in other colleges.
- During challenging times such as a pandemic, Thunderstorm etc., faculty members can teach their students in online mode using the video conferencing facility provided by the BLP. Allied activities such as attendance can also be done through BLP.
- In line with the guidelines of the Sixth Dean Committee, universities can Leverage the BLP to offer online courses for students, allowing them to complete 20% of their courses in an online mode. This can be particularly useful in providing flexibility and accessibility to students.
- The BLP can serve as a valuable platform for the delivery of reference course materials for Undergraduate (UG) and Postgraduate (PG) programs, and these materials can be made publicly accessible.
- BLP can facilitate the delivery of training programs, such as summer/winter schools, CAFT (Capacity Building in Agricultural Education) training, or any other training programs, to students, industry professionals, and farmers from different universities and participants.
- BLP can be leveraged in implementing Academic Bank of Credits (ABC) as per National Education Policy (NEP) 2020, enabling seamless credit transfer across courses, institutions, and educational programs.

6. How does NARES–BLP benefit the faculties?

- Can optimize teaching time by leveraging online resources for content delivery, freeing up classroom time for interactive discussions, hands-on activities, and personalized guidance.
- Offers analytics and data tracking features that enable teachers to monitor student progress, identify areas of improvement, and adjust their teaching strategies accordingly.
- Blended learning platforms often provide opportunities for teachers to connect, collaborate, and share resources with peers, fostering a supportive professional community.
- Can promote student engagement through interactive multimedia content, collaborative projects, and real-world applications by a mix of online and in-person activities.
- Can continuously evaluate and refine their teaching methods, making evidence/data-based adjustments to optimize student learning outcomes.

7. How does NARES–BLP benefit the Students?

- Can have the flexibility to access and engage with learning materials and activities at their own pace and convenience, accommodating different learning styles and preferences
- Each student can receive appropriate support enabled by educators who differentiate instruction by providing targeted interventions or extensions based on student performance and progress
- Online assessments and interactive activities provide immediate feedback to students, allowing them to track their progress, identify areas for improvement, and reinforce their understanding of concepts.
- Can foster teamwork, communication skills, and the ability to work effectively with peers, both in-person and through virtual platform by leveraging collaborative projects and online discussions
- Students can benefit from the expertise of both their classroom teacher and external subject matter experts through virtual guest lectures, online forums, or video conferencing sessions.

Networking Tools and Cyber Security

Subhash Chand

ICAR-Indian Agricultural Statistics Research Institute, New Delhi - 110 012

Subhash.chand@icar.gov.in

1. Introduction

In today's interconnected digital world, where data breaches and cyber-attacks are becoming more prevalent, the need for robust cyber security measures is paramount. Networking tools play a crucial role in safeguarding networks, systems, and sensitive information from malicious activities. By utilizing various networking tools, organizations can strengthen their defense mechanisms and proactively detect, prevent, and respond to cyber threats. This article explores the significance of networking tools in the realm of cyber security, the common tools used, their implementation and management, best practices, emerging trends, challenges, and the future outlook in this dynamic field. Understanding the role and effectiveness of networking tools is essential for establishing a secure and resilient cyber landscape.

Cybersecurity has become an increasingly critical concern in today's interconnected digital world. With the rise in cyber threats and sophisticated attack techniques, organizations and individuals must prioritize the detection and prevention of cyber-attacks to safeguard their sensitive information and digital assets. This article explores the fundamental concepts of cyber security detection and prevention, shedding light on the common threats and attack vectors faced by individuals and businesses. It highlights the importance of early detection and proactive prevention measures, delves into various detection techniques, and provides insights into implementing robust cybersecurity strategies. Additionally, this article explores the role of artificial intelligence and machine learning in enhancing cybersecurity efforts, effective incident response and recovery techniques, as well as emerging trends and future challenges in the field of cyber security detection and prevention.

2. Introduction to Networking Tools and Cyber Security

Definition of Networking Tools

Networking tools are essential software or hardware components that facilitate the communication and

exchange of data between devices in a network. These tools enable the smooth flow of information, improve efficiency, and promote collaboration among connected devices.

Importance of Cyber Security

Cyber security is the practice of safeguarding digital systems, networks, and data from unauthorized access, theft, and damage. In today's interconnected world, where cyber threats are becoming increasingly sophisticated, protecting sensitive information has become paramount. Cyber security measures are crucial to prevent unauthorized access, data breaches, and other cyber-attacks, ensuring the confidentiality, integrity, and availability of digital assets.

The role of detection and prevention in cybersecurity

When it comes to cybersecurity, detection and prevention are two sides of the same coin. Detection refers to the ability to identify and recognize cyber threats and attacks, while prevention involves taking proactive measures to stop those threats in their tracks. Together, detection and prevention form the front line of defense against cyber-attacks, helping to keep our digital lives safe and secure.

3. Importance of Networking Tools in Cyber Security

Understanding the Role of Networking Tools

Networking tools play a pivotal role in cyber security by providing the necessary infrastructure and mechanisms to protect networks and data. They help establish secure connections, monitor network traffic, detect potential threats, and prevent unauthorized access to sensitive information.

Enhancing Threat Detection and Prevention

Networking tools bolster cyber security efforts by enhancing threat detection and prevention mechanisms. They enable the monitoring of network traffic for suspicious activities, such as unauthorized access attempts or unusual data transfers. By analyzing network behavior, these tools can identify potential security breaches and alert administrators to take immediate action, preventing significant damage.

4. Common Networking Tools for Cyber Security

Firewalls

Firewalls act as a security barrier between an internal network and external networks by monitoring and controlling incoming and outgoing network traffic. They examine data packets and apply predefined security rules to allow or block specific connections, protecting against unauthorized access and malware.

Intrusion Detection Systems (IDS)

Intrusion Detection Systems (IDS) monitor network traffic and identify suspicious or potentially malicious activities. They analyze network packets, log information, and compare it against known attack patterns. IDS can provide real-time alerts, enabling prompt responses to mitigate potential threats and prevent further damage.

Virtual Private Networks (VPNs)

Virtual Private Networks (VPNs) establish secure and encrypted connections over insecure networks, such as the internet. By creating a secure tunnel, VPNs ensure the confidentiality and integrity of data transmitted between devices, protecting sensitive information from unauthorized interception. They are particularly useful when accessing networks remotely or connecting to public Wi-Fi hotspots.

5. Implementing and Managing Networking Tools for Cyber Security

Assessing Network Vulnerabilities

Before implementing networking tools, it is crucial to assess network vulnerabilities and identify potential weak points. This assessment helps determine the specific security requirements and select the most suitable tools to address the network's unique needs.

Selecting and Deploying Networking Tools

Once the vulnerabilities are identified, it is important to carefully select and deploy the appropriate networking tools. Consider factors such as compatibility, ease of use, scalability, and the ability to integrate with existing network infrastructure. Proper planning and implementation ensure that the networking tools effectively fortify the network's security.

Monitoring and Maintenance of Networking Tools

Networking tools require consistent monitoring and maintenance to ensure optimal functionality and effectiveness. Regular updates, patch management, and monitoring for potential vulnerabilities or performance issues are essential. Additionally, periodic audits and testing can identify any gaps in security and enable timely adjustments to further enhance the network's resilience against cyber threats.

Best Practices for Network Security with the Help of Tools

When it comes to network security, staying on top of regular updates and patches is like regularly flossing your teeth—it's not the most exciting task, but it's necessary for the health of your system. Just like how you wouldn't leave a cavity untreated, you shouldn't leave security vulnerabilities unpatched. So, make sure you keep those updates coming!

Another essential practice is to implement strong access controls and user authentication. Think of it as having a bouncer at the entrance of a trendy nightclub—only the right people with the right credentials get in. By enforcing strict access controls, you minimize the risk of unauthorized individuals gaining access to your network. It's like putting a velvet rope around your digital club.

##Emerging Trends in Networking Tools and Cyber Security

Artificial intelligence and machine learning aren't just for sci-fi movies anymore—they're making waves in the field of cyber security. These technologies can analyze vast amounts of data and identify patterns that humans might miss. It's like having Sherlock Holmes on your security team, except without the fancy hat and British accent.

Cloud-based networking security solutions are also gaining popularity. They provide convenient and scalable security services without the need for heavy on-premises hardware. It's like having your security

system hosted in the cloud—no need to worry about changing the batteries in your smoke detectors!

##Challenges and Risks in Using Networking Tools for Cyber Security

Networking tools for cyber security can bring their fair share of challenges and risks. One such obstacle is the complexity and integration challenges that arise when dealing with multiple tools. It's like juggling chainsaws while riding a unicycle—it takes skill and finesse to ensure everything works harmoniously.

Another issue to watch out for is false positives and negatives. Sometimes, security tools can get a little overzealous and flag harmless activities as threats. On the other hand, they can also miss actual threats, slipping through the cracks like a ninja in the night. It's like having a smoke detector that randomly goes off when you're just trying to make popcorn—but then stays silent when there's an actual fire.

##Conclusion and Future Outlook for Networking Tools and Cyber Security

In the ever-changing landscape of cyber security, continuous innovation and adaptation are key. Hackers are constantly evolving their tactics, so we must stay one step ahead. Networking tools play a vital role in our defense, but we can't rely solely on current solutions. We need to keep pushing the boundaries, exploring new technologies, and thinking outside the box. It's like a game of cat and mouse, except the mouse has access to the internet and the cat has an arsenal of hacking tools.

So, let's embrace the power of networking tools and cyber security, while keeping our sense of humor intact. After all, if we can't laugh at a failed security update or a false alarm, what can we laugh at? Stay secure, my friends!

6. Anticipating and overcoming future challenges in cybersecurity

The future of cybersecurity will undoubtedly bring new challenges. From the rise of Internet of Things (IoT) devices to the increasing complexity of cloud environments, anticipating and preparing for these challenges will be crucial. Continuous education, training, and collaboration with industry experts will help you navigate these uncharted waters and keep your digital assets secure. In conclusion, cyber security detection and prevention are essential components in safeguarding against the ever-growing threats posed

by cyber criminals. By understanding common attack vectors, implementing early detection measures, and utilizing robust prevention strategies, individuals and organizations can better protect their digital assets and sensitive information. The integration of artificial intelligence and machine learning further enhances the effectiveness of cyber security efforts. However, it is crucial to remain vigilant and anticipate emerging trends and challenges in the cyber security landscape. By staying informed and proactive, we can continue to strengthen our defenses and ensure a safer digital environment for all.

7. Conclusion and Future Outlook for Networking Tools and Cyber Security

In conclusion, networking tools play a crucial role in enhancing cyber security by providing effective threat detection and prevention mechanisms. By implementing and managing these tools, organizations can build robust defenses to safeguard their networks and valuable data. However, it is important to stay updated with emerging trends and continuously adapt to the evolving threat landscape. As technology advances, networking tools will continue to evolve, incorporating innovations such as artificial intelligence and cloud-based solutions. By embracing these advancements and addressing the challenges and risks associated with networking tools, we can pave the way for a more secure digital future. By prioritizing the implementation of networking tools and adhering to best practices, we can defend against cyber threats and protect our digital assets.

8. FAQ

1. What are networking tools in the context of cyber security?

Networking tools in cyber security refer to software applications, hardware devices, and technologies designed to secure networks and protect them from unauthorized access, intrusion attempts, and malicious activities. These tools include firewalls, intrusion detection systems, virtual private networks, and other solutions that enhance network security.

2. How do networking tools contribute to cyber security?

Networking tools play a significant role in cyber security by providing mechanisms for threat detection, prevention, and response. These tools help in monitoring network traffic, detecting anomalies, blocking suspicious activities, and establishing secure communication channels. Networking tools also assist in identifying vulnerabilities and implementing necessary security measures to protect against potential attacks.

3. What are some common networking tools used in cyber security?

Some common networking tools used in cyber security include firewalls, which monitor and control incoming and outgoing network traffic, intrusion detection systems (IDS), which detect and alert about potential intrusion attempts, and virtual private networks (VPNs), which establish secure and encrypted connections for remote access. Other examples include network monitoring tools, vulnerability scanners, and antivirus software.

4. What are the best practices for utilizing networking tools in network security?

When utilizing networking tools for network security, it is essential to regularly update and patch the tools to ensure they are equipped with the latest security features. Strong access controls and user authentication mechanisms should be implemented to prevent unauthorized access. It is also important to monitor and maintain the networking tools effectively to identify and mitigate any potential security risks. Lastly, organizations should stay informed about emerging trends and advancements in networking tools to adapt to the evolving threat landscape.

5. Why is early detection important in cyber security?

Early detection plays a crucial role in cyber security as it allows organizations and individuals to identify potential threats before they escalate and cause significant damage. By detecting and addressing cyber threats at their initial stages, it becomes easier to prevent unauthorized access, data breaches, and other malicious activities.

6. How can artificial intelligence and machine learning enhance cyber security?

Artificial intelligence (AI) and machine learning (ML) technologies offer a range of benefits in cyber security. AI can analyze vast amounts of data and identify patterns and anomalies that may indicate a cyber threat. ML algorithms can continuously learn from data, enabling them to detect and respond to new and evolving threats more effectively. These technologies enhance the speed and accuracy of threat detection and help organizations stay ahead of cyber criminals.

7. What are some effective incident response and recovery techniques?

Effective incident response and recovery techniques involve having a well-defined incident response plan in place. This plan should include steps for identifying, containing, eradicating, and recovering from a cyber-attack. It is crucial to have a designated incident response team, clear communication channels, and regular practice drills to ensure a prompt and effective response when a security incident occurs.

8. What are some emerging trends and future challenges in cyber security detection and prevention?

As technology advances, cyber criminals continue to develop new and sophisticated attack techniques.

Some emerging trends in cyber security include the increasing use of artificial intelligence by both attackers and defenders, the rise of ransomware attacks, and the growing threat to Internet of Things (IoT) devices. Future challenges will likely include addressing the shortage of skilled cyber security professionals, ensuring the security of cloud-based services, and tackling the complexities of securing interconnected systems in an increasingly digital world.