Introduction to Basic of programming

Course Benefits:

Participants will be exposed to the core basics of computer programming thus building a solid foundation for learning programming in any language which will in turn prepare one for a promising future in the field of computer programming.

Computer and Computer Science

Computer science is the study of computers and computing, including their theoretical and practical applications. It applies the principles of mathematics, engineering, and logic to a variety of functions, including:

- Algorithm formulation
- Software and hardware development
- Artificial intelligence
- Information theory
- The way humans interact with technology

A computer is defined as an electronic device designed for storing and processing data, typically in binary form.

Information Technology

The term Information Technology is defined as the study or use of systems (especially computers and telecommunications) for storing, retrieving, and sending information. It refers to the uses of computers, networking, software and other equipment to manage information. In the modern world information technology is integral to the success of business and most companies are equipped with computers, DBMS (Database Management Systems), servers(computers with high computing capacity) for storing, processing, retrieving and protecting information of the company

Information and Communication Technology

Information and Communication Technology (ICT) is an umbrella term that includes communication device or application, encompassing: radio, television, cellular phones, computer and network hardware and software, satellite systems, etc. and all the technical means for processing and communicating information . The terms encompasses both digital as well as pre-digital technologies, including paper-based writing. However, it is most often used to describe digital technologies including:

1. Components of data communication such as communication protocols, transmission techniques, communications equipment, transmission medium.

, as well as

2. available techniques for data storage and information processing.

The term has been coined out due to the convergence of information technology (IT) and telecommunication technology. In a nutshell ICT encompasses elements such as storage media to record information (whether paper, pen, magnetic disk/ tape, optical disks - CD/DVD, flash memory etc.); and also technology for broadcasting information - radio, television,; and any technology for communicating through voice and sound or images- microphone, camera, loudspeaker, telephone to cellular phones and the Internet .

Characteristics of Computers

The characteristics of a computer show the capability and the potential of the computer for processing data. This saves time, space, money, labors etc. And they answer the questions like
why computers are used and why have they become so popular? The following are items that characterize a computer

Speed: In general, no human being or any other device can compete to solve complex computation, faster than computer.

Accuracy: Since the Computer is programmed, so whatever input we give it, it gives result with high degree of accuracy. In other words, it does computation with high precision.

Storage : Computer has the capacity to store mass volume of data with appropriate format.

Diligence: Computer never feels bored and it can work for hours and hours without any break and creating error.

VERSATILITY: Computers are multipurpose. A Computer can perform completely different types of work at the same time.

Memory Efficient: It stores data for future use.

No IQ: Computer needs to be instructed to do its job.

No Feeling: Computer does not have emotions, knowledge, experience and feeling.

TYPES OF COMPUTERS

There are different types of Computers. Besides size and purpose of application, their differences also stem from variations in terms of how they function or method of operation.

Classification by the Method of Operation (Type)

On the basis of their method of operations Computers are further classified into three:

Analog Computers: Analog computers operate by measuring. They deal with continuous variables; as opposed to discrete values or numbers, rather, they operate by measuring physical magnitude such as pressure, temperature, voltage, current etc.

Examples:

a. Thermometer, Voltmeter, Speedometer

b. Gasoline pump – Contains an analog Computer that converts the flow of pumped fuel into two measurements the price of the delivered gas and the quantity of pumped fuel.

Note: They are special purpose computers, and they are known for limited accuracy.

Digital Computers: Digital computers deal with discrete variables; they operate by counting rather than measuring. They operate directly upon numbers (or digits) that represent numbers, letters, or other special symbols.

Examples:

Abacus
Desk & pocket computers
The general purpose computers

Note: Digital computers have higher accuracy and speed than the analog ones.

Hybrid Computers: The best features of analog and digital computers can be combined into a single device to form a hybrid computer. A hybrid computer processes the information by collecting input data with analog methods and

converting it into digital quantities; processes the digital values and then converts the output from digital to analog form.
Example: The following paragraph describes an example of the application of a hybrid
1. Computer In Healthcare
In hospital or healthcare centers analog devices can be used to measure a patient's heart function, temperature and other vital signs. These measurements can then be converted into numbers and supplied to a digital component in the system. This component is used to monitor the patient's vital signs and sends an immediate signal to the healthcare station if any abnormal readings are detected.

Classification By Purpose Of Application
Computers can be applied or used for different purposes. Based upon their application, they are classified as special purpose or general-purpose computers.
Special purpose computers: They are designed to solve a single type of problem, that is their components and functions are uniquely adapted to a specific situation involving specific application.
Example:

The public telephone box,
Traffic control system,
Ticket machines (used in grocery, super markets etc.),
Pocket calculators,
Counters etc.

Most analog computers are special purpose computers.

General-Purpose Computers: They are designed to solve variety of problems through the use of "store program concept". A program or set of instructions designed to solve a problem is read and stored into the memory and then executed by the computer one by one. The same computer can be applied to solve another set of problem using different program. General purpose computers are more flexible and versatile.
Examples:

Micro computers
Mini computers
Super computers etc.

Introduction to Computer Systems
Classification by Physical Size, Capacity and Performance
At this stage, by a computer, we mean a general-purpose digital computer. There is a wide variety of general purpose digital computers on the market place today, in terms of physical size, price, capacity, and performance. They are classified as follows by their capacity and size:
Super Computers: Super computers are the fastest, largest and most potential types of computer. They have speed of hundreds of millions of operation per second, a primary memory capacity of about 80 million characters, a secondary memory of capacity of about 20 times its primary memory.

They are multi-user systems in intercontinental range.
They can carry out enormously complex scientific calculations.

They are used to process huge amount of data and are commonly used in areas where there is a pressing need for big storage as well as performance capacity. Sectors where Super computers are in high demand include space technology centers, meteorology stations, astronomical observatories, intercontinental communications, airline organizations.

Mainframe computers: Smaller than supercomputers in size and capacity, lower in speed & memory capacity than the supercomputers. However they are multi-user systems and handle hundreds of users, usually used in large organizations.

Mini computers: Have relatively lower speed, can handle multi-users, are smaller in size than the mainframe computers. They use terminals for inputs and output. Mini computers are used in small organizations.

Micro computers: Micro computer (personal or desktop computer) is a computer whose CPU is a microprocessor. Microprocessor is a processor all of whose components are on a single integrated-circuit chip. Since its CPU is integrated in a single circuit, it can serve only a single user at a time. Most of home and personal office computers are microcomputers. The
relative performance and usage of personal computer is relatively increased with a very high rate.

Computer and related Terms

Computer Science is the study of Computers and Computational systems. It is a science concerned with the representation, storage, manipulation or processing and presentation of information. Like any other science, which uses some devices for the practical aspect, computer Science uses an electronic or digital device called Computer.

Like other sciences Computer Science has different fields of specialization or sub-disciplines. Major fields of study in Computer Science include artificial intelligence, computer systems and networks, security, database systems, human computer interaction, vision and graphics, numerical analysis, programming languages, software engineering, bio informatics and theory
of computing. A description for some of the disciplines is given below.

Technology: The sum of all scientific, practical, engineering, applied, and theoretical knowledge in the fields of natural, human, and social sciences. Technology, broadly defined, encompasses a range of disciplines and applications that enable humans to change their environment to satisfy their needs. It includes, but is not limited to, computing, information systems, networking, artificial intelligence, machine learning, biotechnology, and many other areas. In the digital age, technology often focuses on automating processes, optimizing resource usage, enhancing communication, and enabling collaboration. Additionally, technology often addresses pressing societal issues, such as poverty reduction, healthcare improvement, and environmental protection The development and application of technology involve contributions from a wide range of disciplines, including physics, chemistry, biology, economics, sociology, psychology, and engineering. Moreover, technology constantly evolves and interacts with society, culture, and politics, shaping our daily lives and transforming the way we work, learn, and live.

Software engineering: It is concerned about the development of a better quality software by applying scientific and basic engineering principles. Software engineering is a professional engineering discipline that applies the principles of

engineering, scientific method, and mathematical logic to the design, analysis, testing, and evaluation of software.In the context of software engineering, 'software' refers to a system or set of programs, while 'engineering' suggests the application of scientific principles and techniques. This field aims to produce software that is reliable, maintainable, efficient, and effective. Some of the common tasks in software engineering include requirements analysis, design, coding, testing, debugging, documentation, configuration management, and project management. Software engineers use a variety of tools, languages, and platforms to achieve these tasks, and they often collaborate with other professionals, such as software developers, database administrators, and system administrators. In addition to traditional software development tools and techniques, software engineers increasingly rely on modern technology trends like agile methodologies, version control systems, automated testing tools, and artificial intelligence. These advancements help software engineers streamline their workflows, improve software quality, and enhance the overall productivity and efficiency of the software development process. Software engineering plays a crucial role in modern society, as it directly influences the way we communicate, interact with computers, and perform various tasks. By providing well-engineered, reliable, and efficient software, software engineers contribute significantly to the growth and advancement of technology and society as a whole.

Computer Network: It is concerned with connecting computers for the purpose of sharing resources. A computer network is a collection of hardware, software, and people, all connected or able to connect and communicate. The purpose of a computer network is to facilitate data sharing and collaboration among its members. Computer networks can be classified into different types based on their geographical area, architecture, or specific function. For example, local area networks (LANs) connect computers within a small geographical area, such as a home or office, while wide area networks (WANs) span larger areas, such as an entire city or region. Additionally, computer networks can be categorized into client-server networks, peer-to-peer networks, and other types based on their architecture and specific use cases. Computer networks have revolutionized the way we work, live, and communicate. They enable efficient and secure data sharing and collaboration among individuals, organizations, and devices. Additionally, computer networks provide a robust and scalable infrastructure for the development and deployment of advanced computing services, such as cloud computing and the Internet of Things (IoT). Computer networks rely on various protocols and services to facilitate communication and collaboration. Examples of common protocols include the Transmission Control Protocol (TCP) and the Internet Protocol (IP), which govern how data is transmitted and addressed across a network. Services such as the Domain Name System (DNS) and the Simple Mail Transfer Protocol (SMTP) are also integral to the functioning of computer networks. Moreover, the World Wide Web (WWW) and the Internet provide the platform for a vast array of online services and applications, enabling users to access information and perform various tasks over the internet. As the digital world continues to evolve, computer networks and the underlying technologies and services that enable them must also evolve. This includes advancements in network architecture, protocols, services, and security measures. The development and adoption of emerging technologies, such as

artificial intelligence, blockchain, and 5G networks, will play a crucial role in shaping the future of computer networks.

Binary Digit: a single digit number in base-2, in other words, either a 1 or a zero. It is the smallest unit by which a computer data is represented.

Byte: A set of Bits that represent a single character.

Programming: Programming refers to a technological process for telling a computer which tasks to perform in order to solve problems. You can think of programming as a collaboration between humans and computers, in which humans create instructions for a computer to follow (code) in a language computers can understand.

Database and Database Systems: A database is a structured set of data that is stored electronically. It is a way of organizing, managing, and storing large amounts of data in an efficient manner. Databases are essential for most types of software, including those for websites, apps, and other software applications.

A database management system (DBMS) is a software that enables the creation, maintenance, and management of databases. Examples of DBMS include MySQL, Microsoft SQL Server, Oracle Database, and MongoDB.

In a database, data is organized into tables. Each table has rows and columns, where each row represents a specific data entry, and each column represents a specific field of that data entry. Relationships between tables are defined by keys.

Some key concepts related to databases include:

Database: A collection of data that is organized into tables.

Table: A collection of related data organized into rows and columns.

Column: A single field in a table that represents a specific attribute of each row.

Row: A single data entry in a table that represents an individual data point.

Primary Key: A unique identifier for each row in a table.

Foreign Key: A field in a table that is a primary key in another table, used to establish a relationship between the two tables.

Relationship: A connection between two or more tables that represents how data in one table relates to data in another table.

Here's a simple example of a database schema:

Customers Table:

CustomerID	Name	Email
1	John	john@example.com
2	Jane	jane@example.com

Orders Table:

OrderID	Product	Quantity	CustomerID
1	Laptop	1	1
2	Smartphone	2	2

In this example, the 'Customers' table and the 'Orders' table are related by the 'CustomerID' field, which is the primary key in the 'Customers' table and a foreign key in the 'Orders. Learn more on databases with a DBMS software like MYSQL.

Artificial Intelligence: It concerned with the means by which Computers can perform tasks that would be characterized as intelligent if performed by human beings.
Artificial Intelligence sometimes called AI stands refers to the technology and methods used to enable computers to simulate human intelligence in performing tasks. This includes tasks such as visual perception, speech recognition, decision-making, and learning.
AI involves using algorithms, statistical analysis, and computational models to extract knowledge from large amounts of structured and unstructured data. It has a wide range of applications, including:
1. Search engines: These utilize algorithms to find relevant information from the internet based on user queries.
2. Social media analysis: This involves using AI algorithms to analyze and interpret data from social media platforms to identify patterns, trends, and insights.
Healthcare diagnostics: AI algorithms can be used to analyze medical images and identify potential issues.
3. Self-driving cars: AI-powered systems analyze sensor data from cars to make decisions about steering, braking, and acceleration.
4. Financial fraud detection: AI can be used to analyze large amounts of financial data to identify patterns that may indicate fraudulent activity.
AI also includes more advanced techniques, such as machine learning, deep learning, and natural language processing. These methods allow computers to learn from and make decisions based on the data they have been exposed to, without explicit programming for every task.

Register: A register is a small piece of storage located within the CPU itself. It holds values that the CPU uses for performing calculations and other operations.There are different types of registers in a CPU, each serving a specific purpose. For example, there are general-purpose registers that can hold data and instructions for any type of operation. Additionally, there are special-purpose registers, such as the program counter (PC), which holds the address of the next instruction to be executed, or the stack pointer (SP), which points to the top of the stack. The registers provide fast access to data and instructions for the CPU. They are also the smallest and simplest memory available to a CPU. However, the number of registers in a CPU is typically limited. This is why efficient code generation and instruction scheduling techniques are essential to optimize the use of these limited resources.
Here's a simple example of a register usage:
CPU performs an addition operation and stores the result in a register (let's call it register A). The CPU then needs to multiply this result by another value. Instead of loading the result from register A into another register, the CPU can directly use register A to perform the multiplication. By doing this, the CPU avoids wasting time on data transfers and can execute instructions more quickly.

BASIC COMPUTING (LAPTOP AND DESKTOP COMPUTERS)

Bus: a distinct set of conductors carrying data and control signals within a computer system, to which pieces of equipment may be connected in parallel.

File: A file is a contiguous set of bytes used to store data. In computer systems, it serves as a method of organizing, storing, and accessing digital data.

There are different types of files based on their purpose and structure:

1. Text Files: These files contain text and are saved with a .txt extension.

2. Image Files: These files contain digital images and are saved with extensions like .jpg, .png, .gif, etc.

3. Audio Files: These files contain digital audio recordings and are saved with extensions like .mp3, .wav, etc.

4. Video Files: These files contain digital video recordings and are saved with extensions like .mp4, .mkv, .avi, etc.

5. Executable Files: These files contain programs and applications that can be executed by the computer. They are saved with extensions like .exe for Windows, .app for macOS, and .elf for Linux.

6. Document Files: These files contain formatted text and other multimedia elements. They are saved with extensions like .docx for Microsoft Word, .pdf for Portable Document Format, etc.

7. Database Files: These files contain structured data that can be accessed and managed by a database management system. They are saved with extensions like .db, .sql, etc.

Each file is associated with a unique identifier, called a file name. This file name is typically organized into two parts: the base name and the file extension.

The organization and management of files is typically performed by the computer's operating system and file system. This includes creating, modifying, deleting, and accessing files, as well as managing the allocation and deallocation of disk space.

Folder: A folder, also known as a directory, is a hierarchical data structure in a file system that serves two primary purposes:

- Organizing and categorizing files.
- Providing a way to locate and access files.

A folder can contain an arbitrary number of files and/or other folders, forming a directory tree or file hierarchy. Each file and folder within a folder has a unique name, known as a file path, that is used to identify it.

Folders play a crucial role in organizing a computer's data and are fundamental to using operating systems like Windows, Mac OS, and Linux. They help users manage their files and directories more efficiently.

Hardware: The computer hardware refers to the physical components of a computer system. Hardware, or "hardware equipment," is the physical aspect of computing devices. It consists of tangible items that are used to interact with and perform

computational tasks. These devices are often integrated into consumer electronics and office machines. The hardware in a computer system includes components such as:

The Central Processing Unit (CPU), which serves as the "brain" of the computer and is responsible for executing instructions.

The computer's memory, which stores data and instructions while the computer is running.

The input and output devices, such as keyboards, mice, and monitors, which enable interaction with the computer.

The computer's storage devices, such as hard drives, solid-state drives, and USB flash drives, which store data and programs.

Additionally, **network devices**, such as routers, switches, and firewalls, can also be considered part of the computer hardware, as they facilitate communication between different devices.

Computer Memory

The computer has a memory that can hold both data and also the program processing that data. In modern computers this memory is known as RAM(Random Access Memory). It is a temporary storage space on a computer or mobile device that is used to hold data and instructions for immediate access by the processor. It enables the computer to execute tasks and applications smoothly by quickly accessing the necessary data from the memory rather than from the slower storage devices like the hard drive.

In most modern computers, the primary type of memory is called Random Access Memory (RAM). This is a volatile memory, which means it loses its data when the power is turned off.

Here are some types of memory in a computer system:

RAM: This is a temporary memory where your computer stores and uses data and programs while it is running. It is a crucial part of the system as it determines how much of your data the computer can keep readily available for processing.

ROM: This is a type of non-volatile memory that stores essential programs and data used to boot up and start your computer. ROM is also known as firmware or BIOS.

Cache Memory: This is a small and fast memory, typically located on the motherboard, which is used to store frequently accessed data. It improves the speed of data retrieval.

Virtual Memory: This is a concept in which your computer's operating system creates an illusion of a larger, contiguous block of memory (RAM) by using part of your hard drive's space.

In conclusion, computer memory is an essential part of a computer system, as it plays a vital role in managing data and processing speed. It includes various types of memory, such as RAM, ROM, cache memory, and virtual memory, each serving a unique purpose in the computer's overall functionality.

Input / Output

Input devices and output devices are hardware components that enable the computer to interact with the user and the outside world.

Input devices: Input devices, also known as peripheral input devices, are the means by which a user or an external device provides data or commands to the computer.
These devices include:
1. Keyboards: The most common type of input device, keyboards are used to type in letters, numbers, and commands.
2. Mice: A pointing device, a mouse allows users to select options, navigate the screen, and provide input through clicks and movements.
3. Microphones: These devices are used to record sound. For example, when making a Skype call or recording audio using an application like Audacity.
4. Digital Pens: A type of handheld device used for capturing and storing handwritten notes, images, and drawings digitally.
5. Scanners: Devices that digitize printed documents, allowing users to upload scanned images to a computer.
6. Barcode Readers: Used to read and interpret barcodes, commonly found on products, for inventory and other management purposes.

Output Devices: Output devices, also known as peripheral output devices, are the means by which a computer provides data or results to a user or an external device.
These devices include:

Monitors: Display screens that present visual information to the user, such as the screen saver, the Windows desktop, or the contents of a document being edited.

Printers: Hardware devices that create tangible copies of digital documents by depositing ink or toner onto paper.

Speakers and Headphones: These devices are used to reproduce audio. For example, when playing music, watching a video, or listening to an online podcast.

Projectors: These devices are used to project a large image onto a screen, such as in a movie theater or during a PowerPoint presentation.

Digital Signage Displays: A type of output device used for displaying digital content on screens in public spaces, such as shopping malls, airports, and restaur

The ALU

The Arithmetic/ Logic Unit(ALU) performs mathematical operations((+, -, x, /, …) and logic operations (=, <, >, and, or, not, ...). The ALU is a sub-component of the CPU(Central Processing Unit). The ALU is composed of:

- Circuits responsible for performing arithmetic/logic operations,
- Registers responsible for storing intermediate computational results, and
- Bus that connects the two.

T**he Control Unit**

The control unit is responsible for managing the process of moving data and program into and out of memory. It is also responsible for carrying out (executing) program instructions - one at a time. This includes the idea of a ‘register’ to hold intermediate computational values.

Secondary Storage devices: Secondary storage devices, also known as secondary memory or external memory, are a type of computer data storage that provide data storage capabilities for a computer or server. They are not part of the computer's primary memory, such as random access memory (RAM) or cache memory.
Some common secondary storage devices include:

Hard Disk Drives (HDD): These are the most common type of secondary storage device. They consist of a rotating magnetic disk with a fixed head that reads and writes data. HDDs are widely used for their large storage capacity and relatively low cost.

Solid State Drives (SSD): SSDs are a type of secondary storage device that uses flash memory instead of magnetic disks. They offer faster data access times compared to HDDs due to their lack of moving parts. SSDs are becoming increasingly popular for their speed and reliability.

Network Attached Storage (NAS) Devices: These are dedicated storage devices that provide file and data storage over a network. NAS devices can be used for tasks such as backing up data, providing shared storage for a network, and implementing centralized file management.

External Storage Devices: These are secondary storage devices that are physically connected to a computer via an external interface, such as USB, FireWire, or eSATA. Examples of external storage devices include USB flash drives, external hard drives, and network-attached storage (NAS) devices.

Tape Libraries and Magnetic Tapes: These are older types of secondary storage devices that use magnetic tapes as their storage medium. They are primarily used for long-term data archiving and backup.

Secondary storage devices provide data storage capabilities for a computer or server. They are not part of the computer's primary memory and can be physically attached to the computer or accessed over a network. Examples of secondary storage devices include hard disk drives, solid state drives, network attached storage devices, and external storage devices.

Software

The finite set of instructions (steps) that a computer system follows to perform a given task is called a program. For any program to be executed first it should be loaded in the memory. A Software is therefore defined as a collection of programs that directs a computer to accomplish a given task

Types Software

Software is divided into two types:

- System Software, and
- Application Software.

System software is designed to facilitate the work of the computer's hardware. The following are activities a system software is responsible for:

- It is responsible for organizing and managing the computer's hardware;
- It acts as intermediary or interface between the user and the computer hardware
- It makes what it looks to be a complex hardware more users friendly.

System software is further divided into:

Operating system
Language software
Operating System

Operating system coordinates the activity between the user and the computer. An operating system performs these functions.

1. Controlling operations (control program)
2. Coordinates, or supervises the activity of the computer system.

3. Decides where programs and data should be stored in the computer memory.
4. Handles communications among the computer components, applications software and the user.
5. Controls the saving and retrieving of files to and from disks in the disk drive.
6. It performs all its controlling tasks without the involvement or awareness of the user.
7. Input/output Management
8. The I/O manager coordinates the computers communication with outside world, flow of data to the display screen and other output devices (printers/ plotters) and from the keyboard or other input devices.
9. Handles the flow of data to and from the disk drives (file management).
10. Handles the process of preparing a disk for use, the copying, renaming, erasing task of a file.
11. Command Processing (command Interpreter)
12. It interprets the commands or what you enter using the keyboard or other input devices Computer System

Language Software

Language software is a generic name consisting of various programs that serve as editors and translators to develop programs using programming languages.

Programming languages are broadly classified into:

a. High-level
b. Low-level

Examples of high-level programming languages include C, C++ Java, FORTRAN, etc

Tools such as compilers and interpreters are used to convert programs written in high level languages into machine or low-level language or object code.

Low-level languages are expressed in binary digits, which is the only language the computer can understand.

Application Software

A software designed for a specific purpose is known as application software. Examples of application software include:

1. Word-processing software such as Microsoft Word
2. Spreadsheet software such as Microsoft Excel,
3. Presentation software such as Microsoft PowerPoint,

The Internet

The early Internet was proposed, designed and implemented by American research institutes, universities, and telecommunication companies. Many believe the Internet today is the initial prototype of what is often called the National Information Infrastructure. It is a widespread information infrastructure with rich history that entails many aspects - technological, organizational, and community. It has a tremendous impact upon society with its influence transcending beyond the technical fields of computer communications.

Browser: a software that is used to access various kinds of Internet resources.

Protocol: Rules that computer use to exchange data and communicate

World Wide Web: The World Wide Web (www, W3) is an information space where documents and other web resources are identified by unique identifiers which are known as uniform resource locators (or URIs), and can be accessed via the Internet.

Website: a location connected to the Internet that maintains one or more pages on the World Wide Web

Application/Purposes of the Internet

Since its emergence the Internet has become popular and it has been used for several purposes. It has revolutionized the computer and communications world like nothing before. Through the help of the World Wide Web and websites, the internet has become very useful in many ways for everyone. Today the Internet has brought a globe in a single room. Right from news across the corner of the world, wealth of knowledge to shopping, purchasing the tickets of your favorite movie-everything is at your fingertips.

The following are some of the common uses of the Internet

1. **Communication such as Email, Skype,whatsapp, Facebook etc:** By using the Internet people can communicate in a fraction of seconds with a person who is sitting in the other part of the world.
2. **Information or resource sharing:** The Internet is known for being one of the most important gateways for resource repositories which store or house different types of data or information in different formats. The Internet and the World Wide Web has made it easy for anyone to access these data or information. People can browse the Internet in search of information related to healthcare advises, weather, etc. easily.
3. **Business medium:** Over the years, the internet has proved itself that it is not just another technological tool, it has become increasingly a business platform too. World trade has seen a big boom with the help of the internet, as it has become easier for buyers and sellers to communicate and also to advertise their business. It has become a common practice for businesses to use online classified websites to buy or sell or advertise their products or services. Classified websites save a lot of money and time so this is chosen as medium by most of people to advertise their products.
4. **Social Networking:** Today social networking websites have become an important part of the online community. More and more people as well as companies are joining social networking sites such as Facebook and Twitter for personal as well as business purposes. It has now become a common reality for companies as well as individuals to use social medias or networking websites to build their business brand or profile
5. **Online Shopping:** In today's busy life most of us are interested to shop online. In some parts of the world, nowadays almost anything can be purchased via the internet. In countries like the United States most of consumers prefer to shop from home.
6. **Entertainment:** via the Internet we can find all forms of entertainment from watching films to playing games online. Almost anyone can find the right kind of entertainment for themselves. When people browse the Internet, there are numerous things that can be found. Music, hobbies, news and more can be found and shared over the Internet.
7. **E-Commerce:** Ecommerce is the concept used for any type of commercial maneuvering, or business deals that involves the transfer of information across the globe via the Internet. It has become a phenomenon associated with any kind of online shopping.

8. **Services:** Many services are now provided on the internet such as online banking, job seeking, purchasing itinerary tickets, and guidance services on array of topics in the every aspect of life, and hotel reservations and bills paying. Many governments in the developed world have transformed their services into electronic and as a result citizens interact with governments online.
9. **Job Search:** Internet makes life easy for both employers and job seekers as there are plenty of job sites which connects employers and job seekers.
10. **Networking**: People are connecting with others via the Internet and build their network.

Search Engines: Search Engines are software tools that search documents for specified keywords and returns a list of the documents where the keywords were found. A search engine is a general class of programs, however, the term is often used to specifically describe systems like Google, Bing and Yahoo search that enable users to search for documents on the World Wide Web. Search engines are special sites on the Web that are designed to help people find information stored on other sites. There are differences in the ways various search engines work, but they all perform three basic tasks:
1. They search based on words
2. They keep an index of the words they find, along with where they find them.
3. They allow users to look for words or combinations of words found in that index.

Millions of people around the world use search engines almost on regular basis. The benefits of search engines to both individuals and society is huge. They have made the web a very critical resource repository of our time. They are the most popular and widely used tool to find information online.

The Internet of Things (IoT)

The term Internet of Things(IoT) has no single universal definition. The term Internet of Things generally refers to scenarios where network connectivity and computing capability extends to objects, sensors and everyday items not normally considered computers, allowing these devices to generate, exchange and consume data with minimal human intervention. It can also be defined as the network of physical objects—devices, vehicles, buildings and other items embedded with electronics ,software, sensors, and network connectivity that enables these objects to collect and exchange data. The Internet of Things allows objects to be sensed and controlled remotely across existing network infrastructure, creating opportunities for more direct integration of the physical world into computer-based systems, and resulting in improved efficiency, accuracy and economic benefit. When IoT is augmented with sensors and actuators, the technology becomes an instance of the more general class of cyber-physical systems, which also encompasses technologies such as smart grids, smart homes, intelligent transportation and smart cities. Each thing is uniquely identifiable through its embedded computing system but is able to interoperate within the existing Internet infrastructure.

What do you mean by programming?

Programming is the process of creating a set of instructions that tell a computer how to perform a task. We can program using a variety of computer programming languages, such as JavaScript, Python, and C++.
Types of Programming
1. Procedural programming
2. Functional programming languages. …
3. Object-oriented programming languages. …
4. Scripting languages. …
5. Logic programming languages.

Types of programming
1. Procedural programming languages
A procedural language follows a sequence of statements or commands in order to achieve a desired output. Each series of steps is called a procedure, and a program written in one of these languages will have one or more procedures within it. Common examples of procedural languages include:
C and C++, Java, Pascal, BASIC

2. Functional programming languages
Rather than focusing on the execution of statements, functional languages focus on the output of mathematical functions and evaluations. Each function–a reusable module of code–performs a specific task and returns a result. The result will vary depending on what data you input into the function. Some popular functional programming languages include:
Scala, Erlang, Haskell, Elixir, F#

3. Object-oriented programming languages
This type of language treats a program as a group of objects composed of data and program elements, known as attributes and methods. Objects can be reused within a program or in other programs. This makes it a popular language type for complex programs, as code is easier to reuse and scale. Some common object-oriented programming (OOP) languages include:
Java, Python, PHP, C++, Ruby

4. Scripting languages
Programmers use scripting languages to automate repetitive tasks, manage dynamic web content, or support processes in larger applications. Some common scripting languages include:
PHP, Ruby, Python, bash, Perl, Node.js

5. Logic programming languages
Instead of telling a computer what to do, a logic programming language expresses a series of facts and rules to instruct the computer on how to make decisions. Some examples of logic languages include:
Prolog, Absys, Datalog, Alma-0

Other ways to classify programming languages

You'll find many more ways to categorize languages beyond the five listed above. Let's take a closer look at there other ways you can think about programming languages:

1. Front-end and . back-end languages

Front-end languages are primarily concerned with the 'user' aspect of the software. The front end deals with all of the text, colors, buttons, images, and navigation that the user will face when navigating your website or application. Anyone with a background in graphic design or art may be more inspired to begin learning one of the front-end languages.

Some examples of front-end programming languages include:

HTML , CSS , JavaScript , React

2. **Back-end languages** deal with storage and manipulation of the server side of software. This is the part of the software that the user does not directly come into contact with but supports their experience behind the scenes. This includes data architecture, scripting, and communication between applications and underlying databases.

Anyone with experience in mathematics or engineering may find more interest in back-end development.

Some examples of back-end programming languages include:

JavaScript, PHP, Java, Python, Ruby, C#

3. High-level vs. low-level languages

The biggest factor that differentiates **high and low-level** programming languages is whether the language is meant to be easily understood by a human programmer or a computer. Low-level languages are machine-friendly, which makes them highly efficient in terms of memory usage but difficult to understand without the help of an assembler. Since they're not very people-friendly, they're also not widely used anymore. Examples include machine code and assembly languages.

High-level languages, on the other hand, are less memory efficient but much more human friendly. This makes them easier to write, understand, maintain, and debug. Most popular programming languages in use today are considered high-level languages.

4. Interpreted vs. compiled languages

The distinction between interpreted and compiled languages has to do with how they convert high-level code and make it readable by a computer. With interpreted languages, code goes through a program called an interpreter, which reads and executes the code line by line. This tends to make these languages more flexible and platform independent.

Examples of interpreted languages include:

Python, JavaScript, PHP, Ruby

Compiled languages go through a build step where the entire program is converted into machine code. This makes it faster to execute, but it also means that you have to compile or "build" the program again anytime you need to make a change.

Examples of compiled languages include:

C, C++, and C#, Rust, Erlang

How To Choose a Programming Language

Familiarity: The most important factor is familiarity. If you already know a programming language, stick with it unless there are specific reasons to change.

Project Requirements: If you're working on a specific project, you should consider using a language that best fits the project's needs. Some languages are more suited to certain tasks, such as Python for data analysis, or C++ for high-performance computing.

Job market: The language you choose should be relevant to the job market in your area. For example, if you're applying for a job in a certain field, you may want to choose a language commonly used in that field.

Performance: Some languages are better suited for specific tasks due to their performance capabilities. If your project requires high performance, you may want to choose a language like C++ or Rust.

Ease of Use: Choose a language that you find easy to use and work with. This will ultimately improve your productivity and make the development process more enjoyable.

Availability of Resources: A programming language's popularity and the number of resources available for learning and using it can impact its usefulness. It's generally easier to find resources and support for popular languages.

Personal Preferences: Lastly, it's important to consider your personal preferences when choosing a programming language. Different languages cater to different programming paradigms and styles, so you may find some languages more appealing than others based on your own programming philosophy

Applications of Programming

Artificial Intelligence (AI) and Machine Learning (ML): AI is designed to function intelligently, learn from experiences, and perform tasks that usually require human intelligence. ML algorithms use data to predict outcomes.

Big Data Analysis: Big data refers to massive, complex datasets that can be too large or complex for traditional data processing applications. Programming languages like Python and Scala are used for big data analysis.

Mobile Application Development: Mobile app development is the process of creating an application that can run on mobile devices, such as smartphones and tablets. This includes creating an app's interface, designing the user experience, and implementing the functionality using languages like Java, Kotlin, Swift, or Objective-C.

Web Development: Web development is the process of creating web pages, websites, and web applications using various languages and technologies like HTML, CSS, JavaScript, Python, PHP, and Ruby.

Game Development: Game development involves the use of programming languages like C++, C#, or Lua to create game mechanics, user interfaces, and AI.

Internet of Things (IoT) and Embedded Systems: IoT devices, like smart thermostats or smart watches, typically have embedded systems that use a programming language like C or C++.

Cloud Computing: Cloud computing platforms and services, such as AWS, Azure, and Google Cloud, use various programming languages to provide scalable, on-demand computing resources.
Data Science and Statistics: Data scientists use programming languages like Python and R to analyze large datasets, develop predictive models, and draw conclusions from complex data.
Block chain Technology: Block chain is a distributed data storage technology that allows secure and transparent transactions. It often involves smart contracts written in programming languages like Solidity.
Cryptocurrency and Cyber-security: Programming languages like Python, C++, and Java are used in the development of cryptocurrencies, as well as for cyber-security applications, including creating antivirus software.
These are just a few examples of the wide range of applications for programming.

CODING
Coding creates a set of instructions for computers to follow. These instructions determine what actions a computer can and cannot take. Coding allows programmers to build programs, such as websites and apps. Computer programmers can also tell computers how to process data in better, faster ways.
The Difference Between Programming And Coding
The terms programming and coding are often used interchangeably, but they actually have different meanings. Here is the difference between programming and coding:
Programming:
1. Programming is the process of creating a set of instructions, known as a program, to tell a computer how to perform a specific task. This involves using a programming language to define algorithms, variables, functions, and other structures necessary for the program.
2. Programming involves higher-level problem-solving, creativity, and decision-making skills. It involves transforming an idea or problem into a specific solution that can be implemented by a computer.
Coding:
1. Coding, on the other hand, is the process of converting the programming language code into machine code or bytecode, which can be executed by a computer.
2. Coding involves the detailed translation of algorithms, data structures, and control structures from the programming language into binary instructions that the computer can understand and execute.
3. Coding is more focused on the implementation of the solution, translating the higher-level concepts into a form that can be executed by the computer.
In summary, programming is the broader concept of creating a program, while coding is a specific part of the process that involves translating the program into machine-readable code. Both programming and coding are essential steps in the software development process.

How to Start Programming

1. Figure out why you want to learn to code.
2. Choose which coding language you want to learn first.
3. Take courses.
4. Use tools that make learning to code easier.
5. Check out how other people code.
6. Complete coding projects.

What is Pseudo code

Pseudo code is a detailed yet readable description of what a computer program or algorithm should do. It is written in a formal yet readable style that uses a natural syntax and formatting so it can be easily understood by programmers and others involved in the development process.

Algorithm

In computer programming terms, an algorithm is a set of well-defined instructions to solve a particular problem. It takes a set of input(s) and produces the desired output.

For example: ***An algorithm to add two numbers:***

1. Take two number inputs
2. Add numbers using the + operator
3. Display the result

Qualities of a Good Algorithm

- Input and output should be defined precisely.
- Each step in the algorithm should be clear and unambiguous.
- Algorithms should be most effective among many different ways to solve a problem.
- An algorithm shouldn't include computer code. Instead, the algorithm should be written in such a way that it can be used in different programming languages.

Algorithm Examples

Algorithm to add two numbers

```
Step 1: Start
Step 2: Declare variables num1, num2 and sum.
Step 3: Read values num1 and num2.
Step 4: Add num1 and num2 and assign the result to sum.
        sum←num1+num2
Step 5: Display sum
Step 6: Stop
```

Algorithm to find the largest among three numbers

```
Step 1: Start
Step 2: Declare variables a,b and c.
Step 3: Read variables a,b and c.
Step 4: If a > b
           If a > c
              Display a is the largest number.
           Else
              Display c is the largest number.
        Else
           If b > c
              Display b is the largest number.
           Else
              Display c is the greatest number.
Step 5: Stop
```

Algorithm to find all the roots of the quadratic equation

```
Step 1: Start
Step 2: Declare variables a, b, c, D, x1, x2, rp and ip;
Step 3: Calculate discriminant
         D ← b2-4ac
Step 4: If D ≥ 0
              r1 ← (-b+√D)/2a
              r2 ← (-b-√D)/2a
              Display r1 and r2 as roots.
        Else
              Calculate real part and imaginary part
              rp ← -b/2a
              ip ← √(-D)/2a
              Display rp+j(ip) and rp-j(ip) as roots
Step 5: Stop
```

Algorithm to find the factorial

```
Step 1: Start
Step 2: Declare variables n, factorial and i.
Step 3: Initialize variables
          factorial ← 1
          i ← 1
Step 4: Read value of n
Step 5: Repeat the steps until i = n
     5.1: factorial ← factorial*i
     5.2: i ← i+1
Step 6: Display factorial
Step 7: Stop
```

Algorithm to check prime number

```
Step 1: Start
Step 2: Declare variables n, i, flag.
Step 3: Initialize variables
        flag ← 1
        i ← 2
Step 4: Read n from the user.
Step 5: Repeat the steps until i=(n/2)
     5.1 If remainder of n÷i equals 0
            flag ← 0
            Go to step 6
     5.2 i ← i+1
Step 6: If flag = 0
           Display n is not prime
        else
           Display n is prime
Step 7: Stop
```

THE PURPOSE OF ALGORITHM

The purpose of an algorithm is to solve a problem by defining a step-by-step process or sequence of operations. This sequence is intended to be clear, precise, and unambiguous. The main objective of an algorithm is to automate the process of problem-solving.

An algorithm should have the following characteristics:

Definiteness: The algorithm should clearly define the input and the desired output. It should specify what operations to perform and the order in which they should be performed.

Feasibility: The algorithm should be realistic and feasible to implement. It should not require impossible resources or be overly complex.

Finite Existence: The algorithm should not contain an infinite number of operations. Each operation should be well-defined and finite.

Deterministic: The algorithm should produce the same output for the same input. It should not depend on random variables or external factors.

Usefulness: The algorithm should solve a problem that is practically useful. It should have real-world applications.

Examples of algorithms include searching algorithms (like linear search and binary search), sorting algorithms (like bubble sort and quick sort), and data compression algorithms (like Huffman coding). These algorithms help us efficiently process and analyze data

DATA OUTPUT

An output device is any piece of computer hardware that converts information into a human-perceptible form or, historically, into a physical machine-readable form for use with other non-computerized equipment. It can be text, graphics, tactile, audio, or video.

Functions of a computer

What are the main functions of computer?

The functionality of any computer mainly includes the following tasks;

- Taking input data,
- Processing the data,

- Returning the results, and
- Storing the data

Variables

Variables are data simply data containers. Data could be a unit or several units of bytes. A variable is used to store these units of bytes which can be used as many times as required. Variables makes mathematics operations and calculations easier as a unit value can stored in a variable and the variables used in place of primitive values, we can have a variable y and assign a value 10 thus the expression y=10; which means y+y = 20, y is a variable with a value 10 stored in it and can be used many times

DATA

The quantities, characters, or symbols on which operations are performed by a computer, which may be stored and transmitted in the form of electrical signals and recorded on magnetic, optical, or mechanical recording media.

Types of Data

Quantitative

Quantitative data is data that can be counted or measured in numerical values. The two main types of quantitative data are discrete data and continuous data. Height in feet, age in years, and weight in pounds are examples of quantitative data.

Quantitative data is always numerical. It can be put in a database and analyzed using mathematical and statistical methodologies.

Discrete Data: Data that can only take certain values is called discrete data or discrete values. This is data that can be counted and has a limited number of values. It usually comes in the form of whole numbers or integers. These values must fit into certain categories and can't be broken into smaller parts.

The number of each type of treatment a salon needs to schedule for the week, the number of children attending a nursery each day or the profit a business makes each month are all examples of discrete data. This type of data is often represented using tally charts, bar charts or pie charts.

Continuous data: Continuous data is data that can take any value. Height, weight, temperature and length are all examples of continuous data. Some continuous data will change over time; the weight of a baby in its first year or the temperature in a room throughout the day.

Qualitative Data

Qualitative data is data that cannot be counted, measured or easily expressed using numbers. It is collected from text, audio and images and shared through data visualization tools, such as word clouds, timelines, graph databases, concept maps and info graphics.

Qualitative data analysis tries to answer questions about what actions people take and what motivates them to take those actions. Collecting and working with this kind of data can be time-consuming, because it requires reflection on the part of the analyst. Someone who works with qualitative data is called a qualitative researcher or qualitative analyst.

Quantitative and Qualitative Data

Typically, quantitative data is structured, while qualitative data is unstructured.

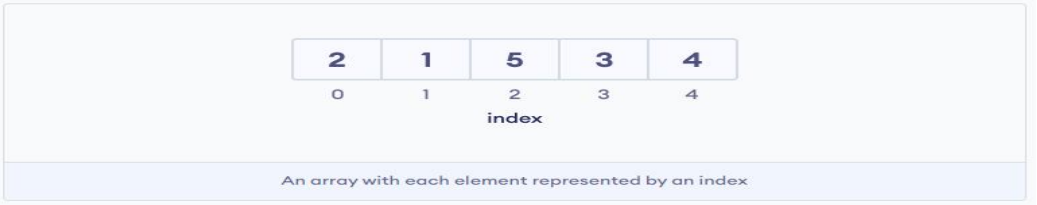
The information gained from performing both quantitative and qualitative data analysis can be complementary, but the goals for examining each type of data and the tools required to gain insight from each are different.
A quantitative data analyst seeks to answer objective questions about an event. In contrast, a qualitative researcher would seek to answer subjective questions about the meaning people assigned to the same event.

Data structure
A data structure is a storage that is used to store and organize data. It is a way of arranging data on a computer so that it can be accessed and updated efficiently.

Data structure is not only used for organizing the data. It is also used for processing, retrieving, and storing data. There are different basic and advanced types of data structures that are used in almost every program or software system that has been developed. So we must have good knowledge about data structures.

Types of data structures
Linear data structures: linear data structures, the elements are arranged in sequence one after the other. Since elements are arranged in particular order, they are easy to implement.
However, when the complexity of the program increases, the linear data structures might not be the best choice because of operational complexities.

Examples of Linear data structures
Arrays - In an array, elements in memory are arranged in continuous memory. All the elements of an array are of the same type. And, the type of elements that can be stored in the form of arrays is determined by the programming language. [2,1,5,4,3].



An array with each element represented by an index

Stack - In stack data structure, elements are stored in the LIFO principle. That is, the last element stored in a stack will be removed first.
It works just like a pile of plates where the last plate kept on the pile will be removed first.

In a stack, operations can be perform only from one end (top here).

Queue Data Structure: Unlike stack, the queue data structure works in the FIFO principle where first element stored in the queue will be removed first.
It works just like a queue of people in the ticket counter where first person on the queue will get the ticket first.

In a queue, addition and removal are performed from separate ends.

Linked List Data Structure: linked list data structure, data elements are connected through a series of nodes. And, each node contains the data items and address to the next node.

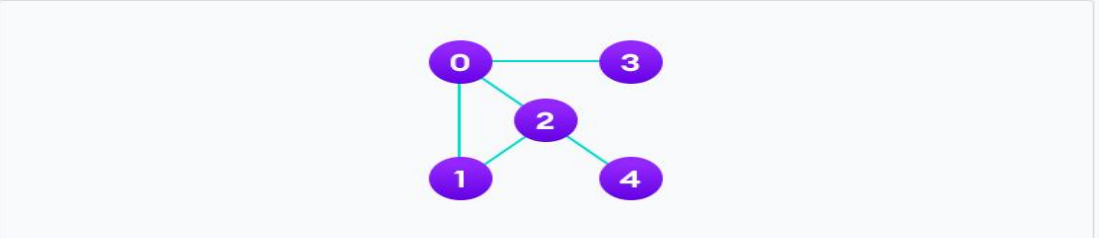
A linked list

Non Linear Data Structures

Unlike linear data structures, elements in non-linear data structures are not in any sequence. Instead they are arranged in a hierarchical manner where one element will be connected to one or more elements.

Non-linear data structures are further divided into graph and tree based data structures.

1. **Graph Data Structure:** In graph data structure, each node is called vertex and each vertex is connected to other vertices through edges.

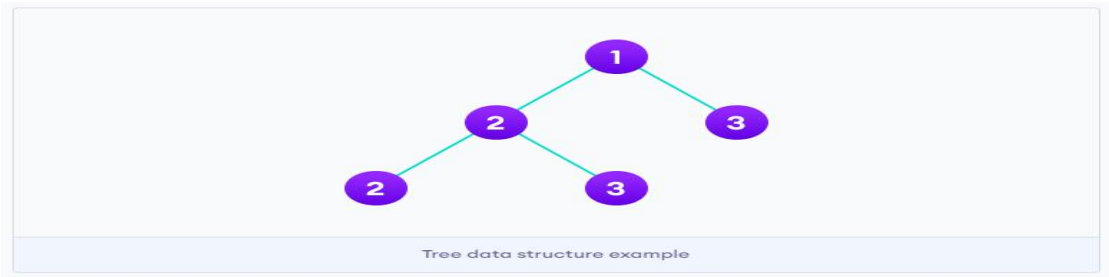


Graph data structure example

Graph Based Data Structures:

- Spanning Tree and Minimum Spanning Tree
- Strongly Connected Components
- Adjacency Matrix
- Adjacency List

Trees Data Structure: Similar to a graph, a tree is also a collection of vertices and edges. However, in tree data structure, there can only be one edge between two vertices.



Tree data structure example

Tree based Data Structure

- Binary Tree
- Binary Search Tree
- AVL Tree
- B-Tree
- B+ Tree
- Red-Black Tree

Linear Vs Non-linear Data Structures

Now that we know about linear and non-linear data structures, let's see the major differences between them.

The data items are arranged in sequential order, one after the other.	The data items are arranged in sequential order, one after the other.
All the items are present on the single layer.	The data items are present at different layers.
It can be traversed on a single run. That is, if we start from the first element, we can traverse all the elements sequentially in a single pass.	It requires multiple runs. That is, if we start from the first element it might not be possible to traverse all the elements in a single pass.
The memory utilization is not efficient.	Different structures utilize memory in different efficient ways depending on the need.
The time complexity increase with the data size.	Time complexity remains the same.
Example: Arrays, Stack, Queue	Example: Tree, Graph, Map

Why Data Structure?

Knowledge about data structures help you understand the working of each data structure. And, based on that you can select the right data structures for your project.This helps you write memory and time efficient code.

Introduction to Loop

A loop is a programming construct that allows a set of instructions to be repeatedly executed. The condition to repeat the instructions is given in the loop construct. The repetition continues until the condition is no longer satisfied.

In other words, a loop is used to repeat a set of instructions or code segments, often when you do not know the exact number of times that the loop will need to execute.

Here are a few examples of loops in different programming languages:

Python:

```
for i in range(5):
print(i)
```

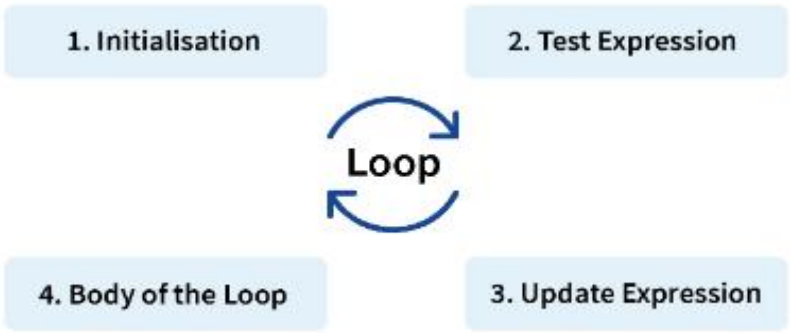
JavaScript:

```
for (var i = 0; i < 5; i++) {
console.log(i);
}
```

Java:

```
for (int i = 0; i < 5; i++) {
System.out.println(i);
}
```

In each of these examples, the loop iterates 5 times. In Python, range(5) generates a sequence of numbers from 0 to 4. In JavaScript and Java, the loop starts at 0 and increments by 1 each time until it reaches 4. The loop prints the value of the variable i at each iteration.



Types of loops

For Loop: In a for loop, the number of iterations is determined before the loop begins. This is commonly used when the number of iterations is known beforehand.

JavaScript:

```
for (var i = 0; i < 5; i++) {
console.log(i);
}
```

While Loop: A while loop is a loop that continues to execute as long as a certain condition is true. The number of iterations is not known beforehand.

Python:

```
i = 0
while i < 5: 3
```

print(i) 4 i += 1

Do-While Loop: A do-while loop is a loop that guarantees its body to be executed at least once. The loop is then repeated as long as the condition is true. However, it's important to note that the do-while loop is not available in all programming languages.

C:

```
#include <stdio.h>
int main() {
int i = 0;
do{
    printf("%d\n", i);
    i++;
} while (i < 5);
return 0;
}
```

NOTE: **For loops ** are used to iterate over a sequence of values, such as a list or a range of numbers.**While loops** are used to repeat a block of code as long as a condition is true.**Do-while loops** are similar to while loops, but the block of code is executed at least once, even if the condition is false.

When And Why Loops

1. Loops are used in programming when a block of code needs to be executed multiple times.

2. To Perform repetitive tasks: Loops are used to execute the same code again and again until a specific condition is met.

3. To Reduce code duplication: Without loops, you would need to write the same code again and again for different variables or iterations. However, using loops can simplify the code and reduce errors.

4. To Control flow: Loops are used to control the flow of execution in a program. By using loops, we can efficiently perform complex tasks by repeating the same code multiple times.

5. To Save Time: Loops allow us to write code more efficiently. Rather than writing separate code for each operation, loops help us to write concise and readable code.

6. To Make Code More Readable: Loops make the code easier to understand by other developers or even by ourselves in the future. By using loops, we can create code that follows a logical structure, which makes it easier to read and maintain.

When and why to use loops depends on the specific problem or task you are working on. In some cases, it may be more efficient to use loops to iterate over a sequence or repeat a code block multiple times. In other cases, it may be more appropriate to use conditional statements or functions to achieve the desired outcome.

FUNCTION

In programming, a function is a block of organized and reusable code that performs a single-related action or operation. It is like a procedure in other programming languages. Functions play a vital role in writing clean, modular, and efficient code. Functions are used to group a series of related statements together and then execute the grouped statements whenever necessary. They make it easier to maintain, update, and reuse the code.

A function can take inputs (called parameters) and may return a value based on the execution of the code block. The return value of a function can be stored in a variable and used in the program.

Functions can be called from anywhere in the code, which allows for maximum code re-usability. This makes programming in higher-level languages like Python or JavaScript easier and more efficient.

Here is an example of a function in JavaScript:

JavaScript: Function Definition

```
Function greet(name){
Console.log(`Hello ${name}! Happy to see you`);
}
```

Function Execution:

```
greet('John Doe');
```

Output = Hello John Doe! Happy to see you

In this example, the greet function takes one parameter called name. The function simply prints a greeting message using the name parameter. The function is then called with different values for the name parameter, resulting in different greeting messages.

Types of Functions

In programming, functions can be broadly categorized into three types:

Procedural Functions: These functions focus on the process or the steps to be executed. They typically follow a series of statements, and may contain loops, conditional statements, or subroutine calls. In some languages, like Python or Java, all functions are procedural in nature.

Procedural Functions in C :

```
void print_number(int num) {
        printf("%d\n", num);
        }
```

Object-Oriented Functions: In Object-Oriented Programming (OOP), functions are typically called methods and are associated with objects or classes. Methods have access to the object's state (data) and can modify it. In languages like Java, C++, or Python, functions can be methods of classes.

Object-Oriented Function in Java:

```
class Printer {
        void printNumber(int num) {
        System.out.println(num);
        }
```

```
}
```

Functional Functions: In Functional Programming (FP), functions are first-class citizens, which means they can be assigned to variables, passed as arguments to other functions, and returned as values from other functions. These functions typically perform a single task or calculation and have no side effects. Languages like Haskell, Lisp, or Erlang support functional programming.

Functional Function (in Haskell):

```
printNumber :: Int -> IO ()
printNumber num = putStrLn (show num)
```

Package and Modules

In programming, a package is a collection of classes and interfaces or (files) that provide functionalities for specific purposes. Packages can be used by different programs or scripts, allowing for code re-usability and maintainability.

In many programming languages, packages are grouped under a single directory, with the directory's name representing the package name. Each file within the directory is considered a module, and they all share the same name space, allowing them to interact with each other easily.

Here's an example of how to define a package in Python:

Python:

```
# filename: mypackage/__init__.py
def my_function():
  print("This is a function in the mypackage package.")
```

In this example, mypackage is the package, and the __init__.py file is used to define the package's content and functionality.

In conclusion, a package in programming is a collection of related modules that can be used by different programs or scripts, providing code re-usability and maintainability.

Types of Packages

Built-in or Standard Libraries:

These packages are part of the programming language itself. They provide a set of common functionalities, such as file handling, regular expressions, networking, etc. Built-in packages are automatically available for use in your programs once you have the language installed.

Here's an example of how to use the math package in Python:

Python:

```
import math
print(math.sqrt(9))
```

Third-Party or External Libraries:

These packages are developed by the programming community or the programmer and are not part of the standard language features. They provide additional functionalities and capabilities beyond what the built-in packages offer.

Here's an example of how to use the requests package in Python, which is not a built-in package:

First, you need to install the requests package using pip:

Python:

```
import requests
response = requests.get('https://api.github.com')
print(response.json())
```

Third-party packages are usually hosted on repositories like PyPI for Python, NPM for Node.js, or Maven Central for Java, and can be easily installed using package managers like pip, npm, or Maven.

Modules

A module is a distinct assembly of components that can be easily added, removed or replaced in a larger system. Generally, a module is not functional on its own.
Modules in programming are reusable, independent units of code that can be easily integrated into a larger application or another module. They help to promote code re-usability, organization, and readability.
Each module is designed to handle a specific task or a group of related tasks. It encapsulates related functions, classes, variables, and other elements within its name space. This isolation prevents the module's internals from being accessed or modified by external code.

Here are some key benefits of using modules:

1. Encapsulation: Modules hide the internal implementation details, making it easier to change or refactor code.
2. Namespacing: Each module has its own namespace, avoiding naming conflicts with other modules or the global namespace.
3. Code organization: Modules promote clean and organized code, making it easier to maintain and understand.
4. Re-usability: Modules can be reused across multiple projects, reducing the need to rewrite the same code multiple times.

For example, in Python, you can define a module like this:

Python:

```
# my_module.py
def my_function():
print("This is a function in my_module.")
```

Then, you can use this module in another file:

```
# main.py
import my_module
my_module.my_function()
```

prints "This is a function in my_module."

Modules in different programming languages might have different names and features, but they generally follow the same principles.

Types of Modules

1. Regular Modules: These are the basic modules in a program. They are imported and executed as a part of the main program.

2. Packages**:** A package is a collection of related modules in a sub-directory. This allows a developer to group together multiple modules in a way that is easier to understand and navigate.
3. Namespace Packages: Namespace packages are a way to break a single package across multiple directories. This allows developers to extend a package across multiple directories.
4. Compiled Modules: These are modules that have been compiled from another language. The Python compiler, for example, can compile C, C++, or Java code into a .pyd file that can be imported as a module.
5. Dynamic Loaded Modules: These are modules that are loaded and executed at runtime. They can be used to add new functionality to a program during runtime.
6. Special Purpose Modules: These are modules that provide unique or specialized functionality. Examples include modules that handle reading and writing files, interacting with the user through a graphical interface, or communicating with external services.
7. Third-party Modules: These are modules developed by third-party developers and organizations that are not part of the Python standard library. They provide additional functionality that may not be available in the standard library.
8. Implicitly Imported Modules: Some modules are implicitly imported whenever a program is run. For example, in Python, the 'main' module is always implicitly imported.
The type of module a developer chooses to use depends on the specific needs of their program and the features that module provides.

Polymorphism
Polymorphism is a principle in object-oriented programming (OOP) that allows objects of different classes to be treated as objects of a common super-class. It promotes flexibility, extensibility, and code re-usability in the software development process.
The concept of polymorphism includes the ability of a single function or method to handle different types of data. It can also involve a single property that can be set to different values.
Here is an example of Polymorphism in Vanilla JavaScript:

```
        class Animal {
                constructor(name) {
                        this.name = name;
                    }
                sound() {
                throw new Error('sound() must be implemented by subclass');
                }
        }

class Dog extends Animal {
 constructor(name) {
   super(name);
 }
```

```
 sound() {
   return 'Woof!';
 }
}

class Cat extends Animal {
 constructor(name) {
   super(name);
 }

 sound() {
   return 'Meow!';
 }
}

// Using polymorphism
let animal = new Animal('Fluffy');
animal.sound(); // throws an error because the base class doesn't implement sound()

animal = new Dog('Buddy');
animal.sound(); // outputs 'Woof!'

animal = new Cat('Fluffy');
animal.sound(); // outputs 'Meow!'
```

In this example, Animal is the base class and Dog and Cat are subclasses.
Both Dog and Cat classes implement the sound method.
The animal variable can hold instances of the Dog or Cat class. This is polymorphism in action, as we can treat objects of different classes as instances of the base class (Animal in this case).
Please note that JavaScript, unlike other statically-typed OOP languages, doesn't have built-in support for abstract base classes. Therefore, to ensure that subclasses implement certain methods, you need to use a different approach, such as throwing an error in the base class's method, as shown in the example.

BUGS AND ERRORS

Bug: A bug, also known as a software bug, defect, or glitch, is an error, flaw, or unintended behavior within a computer program or system. It can cause the program to produce incorrect or unexpected results. Bugs are often reported and fixed by developers using a process called debugging.
In some cases, the term "bug" may be used more loosely to describe a feature or behavior of a system that some users do not like, regardless of whether it is actually a bug in the technical sense of the term.
Error: An error, also known as an exception or an exceptional event, is an anomalous condition or an unexpected result during the execution of a program. This occurs when the program encounters an invalid data, an invalid operation, or an unanticipated external event.

For example, if you are trying to open a file that does not exist, you would encounter a FileNotFound error. If you try to divide a number by zero, you would encounter a ***Division-By-Zero*** error.
Errors are often classified into two categories: **runtime errors** and **logical errors**.
Runtime errors occur during the execution of a program, while **logical errors** occur before the program is even run.
For instance, a program that contains a syntax error will not even run, while a program that divides by zero will encounter a runtime error.

	Error	Bug
Definition	A mistake or a mistake in a program or algorithm.	A mistake in the logic or the algorithm itself that leads to a faulty or incorrect output.
Cause	Minor issues like typographical errors, syntax errors, or logical errors in a small portion of the code.	More significant issues like flawed logic, algorithm design, or inappropriate use of software.
Resolution	Finding the error and fixing it, often involves modifying or adding lines of code.	Finding the bug and fixing it, usually involves revising or replacing the logic or algorithm, or sometimes changing the data.
Impact	Minor or major depending on the type of error and its location in the code.	Can have significant impacts on the overall functionality and correctness of the program.
Prevention/Mitigation	Ensuring code correctness by thorough testing, debugging, and peer review.	Developing robust and accurate logic, algorithm, and overall software architecture.

NOTE: While this differentiation can be useful in certain contexts, the terms error and bug are often used interchangeably in software development.

Debugging

Debugging is the process of correcting bugs that are found within the computer program. A bug is an error, flaw, or unintended behavior in a program or system that causes it to produce incorrect or unexpected results, or to behave in unpredictable ways.

Steps Involved in Debugging:

1. Identifying the bug: The first step is to determine that there is a bug present. This is often done by running the program and observing the incorrect output or unexpected behavior.
2. Isolating the bug: Once the bug is identified, the next step is to narrow down the area of the code where the bug is located. This is typically done by using a process of

elimination and strategically adding debugging code to the program to help identify the source of the problem.
3. Fixing the bug: Once the bug is isolated, the final step is to apply a fix to correct the problem. This may involve modifying or replacing the existing code, or even adding new code to enhance the program's functionality.
4. Testing the fix: After the bug has been fixed, it is important to thoroughly test the program to ensure that the fix was successful and that no new bugs were introduced. This is typically done using a combination of manual testing and automated testing tools.
5. Repeat the process: In complex systems, the process of debugging often needs to be repeated multiple times. As more bugs are found and fixed, the program's functionality improves over time.

It's important to note that debugging is an essential skill for any software developer, as it is the primary means by which bugs are identified, fixed, and prevented from reoccurring.

Sources: https://www.programiz.com/dsa/data-structure-types
INTRODUCTION TO COMPUTER SCIENCE - Dessalegn Mequanint Yehuala