🚀+100 Programming Concepts

👩‍💻 1. Programming Basics

📚 1.1 Syntax

The rules and structure that dictate how programs in a specific language must be written. Syntax is like grammar in human language.

📚 1.2 Variables

Named spaces in memory that store values. The value of a variable can change, hence the name.

📚 1.3 Data Types

Different types of data that can be manipulated in a program, like integers, strings, and booleans.

📚 1.4 Constants and Literals

Fixed values that don't change. Constants are named, while literals are unnamed (e.g., 1, 2.2, "hello").

📚 1.5 Expressions and Statements

Expressions are combinations of variables, values, operators, and functions that the programming language interprets and computes to produce another value. Statements represent actions or commands.

👩‍💻 2. Control Structures

📚 2.1 Conditional Statements (if, else, switch)

Control flow statements that perform different computations or actions depending on whether a programmer-specified boolean condition evaluates to true or false.

📚 2.2 Loops (for, while, do-while)

Control flow constructs that allow code to be executed repeatedly based on a condition.

📚 2.3 Break and Continue

'Break' is used to exit the current loop before its natural end, while 'continue' skips the rest of the current loop iteration and proceeds to the next one.

📚 2.4 Error Handling (try, catch, finally)

Mechanisms for catching and handling errors or exceptions during program execution.

📚 2.5 Recursion

The process where a function calls itself as a subroutine. This allows the function to be repeated several times, as it can call itself during its execution.

BY: Waleed Mousa

👩‍💻 3. Data Structures

📚 3.1 Arrays

A **data** structure used to store homogeneous elements (integers, float, etc) at **contiguous** memory locations.

📚 3.2 Linked Lists

A data structure consisting of nodes which hold data and also **link to** the next node **in** the list.

📚 3.3 Stacks and Queues

Stacks are a type of data structure where the last element added is the first one out (LIFO). Queues are a type of data structure where the first element added is the first one out (FIFO).

📚 3.4 Trees

A hierarchical data structure **with a** root value **and** subtrees **of** children, represented **as a** set **of** linked nodes.

📚 3.5 Graphs

A non-linear data structure consisting **of** nodes **and** edges. The nodes are sometimes also referred to **as** vertices **and the** edges are **lines or** arcs that connect **any** two nodes **in the** graph.

👩‍💻 4. Algorithms

📚 4.1 Searching Algorithms (Binary Search, Linear Search)

Methods for finding a particular data in a **structure. Binary** search works on sorted data, while linear search works irrespective of data **being** sorted.

📚 4.2 Sorting Algorithms (Quick Sort, Merge Sort, Bubble Sort)

Methods **for** rearranging **a** list **of items in a** particular order (**ascending or descending**).

📚 4.3 Recursive Algorithms

Algorithms that express **the** solution **of a** problem **in** terms **of a function that calls itself**.

📚 4.4 Dynamic Programming

A **method for solving complex problems by breaking them down into simpler subproblems and utilizing solutions to subproblems to build up the solution to the overall problem**.

📚 4.5 Greedy Algorithms

Algorithms that make **the** locally optimal choice **at each** stage **with the** hope **of** finding **a** global optimum.

BY: Waleed Mousa

👩‍💻 5. Object-Oriented Programming

📚 5.1 Classes and Objects

Classes are blueprints for creating objects (a particular data structure), providing initial values for state (member variables) and implementations of behavior (member functions or methods).

📚 5.2 Encapsulation

The bundling of data with the methods that operate on that data. Encapsulation is used to hide the values or state of a structured data object inside a class, preventing unauthorized parties' direct access to them.

📚 5.3 Inheritance

A mechanism in which one class acquires the property of another class. Used for code reusability and the method of creating a new class using properties of an existing class.

📚 5.4 Polymorphism

The ability of an object to take on many forms. The most common use of polymorphism in OOP occurs when a parent class reference is used to refer to a child class object.

📚 5.5 Abstraction

A mechanism that represents essential features without including the background details.

👩‍💻 6. Software Development Principles

📚 6.1 DRY (Don't Repeat Yourself)

A principle of software development aimed at reducing repetition of software patterns, replacing them with abstractions or using data normalization to avoid redundancy.

📚 6.2 SOLID Principles

A mnemonic acronym for five principles that make software designs more understandable, flexible, and maintainable.

📚 6.3 KISS (Keep It Simple, Stupid)

A design principle stating that systems perform best when they have simple designs rather than complex ones.

📚 6.4 YAGNI (You Aren't Gonna Need It)

A principle of extreme programming (XP) that states a programmer should not add functionality until deemed necessary.

📚 6.5 Separation of Concerns

The process of breaking a computer program into distinct features that overlap in functionality as little as possible.

BY: Waleed Mousa

👩‍💻 7. Functional Programming

📚 7.1 Pure Functions

A function that depends only on its input and produces the same output for the same input with no side-effects.

📚 7.2 High Order Functions

Functions that operate on other functions, either by taking them as arguments or by returning them.

📚 7.3 Immutability

Unchanging over time or unable to be changed. In functional programming, data is immutable.

📚 7.4 Recursion in Functional Programming

Functional programming relies heavily on recursion as the primary control structure, rather than loops and conditionals.

📚 7.5 Lambda Calculus

The smallest universal programming language in the world. All computations in lambda calculus are done with expressions, not statements.

👩‍💻 8. Programming Paradigms

📚 8.1 Procedural Programming

A programming paradigm, derived from structured programming, that uses a linear or top-down approach.

📚 8.2 Object-Oriented Programming

A programming paradigm based on the concept of "objects", which can contain data and code: data in the form of fields, and code, in the form of procedures.

📚 8.3 Functional Programming

A programming paradigm where programs are constructed by applying and composing functions.

📚 8.4 Event-Driven Programming

A programming paradigm in which the flow of the program is determined by events like user actions, sensor outputs, or messages from other programs.

📚 8.5 Logic Programming

A programming paradigm in which program statements express facts and rules about problems within a system of formal logic.

BY: Waleed Mousa

👩‍💻 9. Memory Management

📚 9.1 Stack and Heap

Stack is a special region of computer's memory that stores temporary variables created by each function. Heap is a region of computer's memory that is not managed automatically for you.

📚 9.2 Garbage Collection

A form of automatic memory management where the garbage collector attempts to reclaim memory occupied by objects that are no longer in use by the program.

📚 9.3 Memory Leaks

Happens when a program does not release the memory it has acquired for temporary use back to the system.

📚 9.4 Pointers

A variable which stores the address of another variable. Pointers are used for various purposes in C and C++.

📚 9.5 Buffer Overflow

An anomaly where a program, while writing data to a buffer, overruns the buffer's boundary and overwrites adjacent memory.

👩‍💻 10. Software Testing

📚 10.1 Unit Testing

A level of software testing where individual units or components of a software are tested.

📚 10.2 Integration Testing

The phase in software testing in which individual software modules are combined and tested as a group.

📚 10.3 Functional Testing

A type of testing which verifies that each function of the software application operates in conformance with the requirement specification.

📚 10.4 Load Testing

The process of putting demand on a system or device and measuring its response.

📚 10.5 Debugging

The process of identifying and removing errors from computer hardware or software.

BY: Waleed Mousa

👩‍💻 11. Databases

📚 11.1 SQL

Structured Query Language, a standard language for managing and manipulating databases.

📚 11.2 NoSQL

A non-relational database designed to scale out, with wide-column stores, graph databases, key-value pairs and document databases.

📚 11.3 ACID Properties

Atomicity, Consistency, Isolation, Durability - a set of properties that guarantee database transactions are processed reliably.

📚 11.4 Indexing

A data structure technique to efficiently retrieve records from the database files based on some attributes on which the indexing has been done.

📚 11.5 Database Normalization

The process of organizing the fields and tables of a relational database to minimize redundancy and dependency.

👩‍💻 12. Web Development

📚 12.1 HTML/CSS/JavaScript

The building blocks of the web. HTML for structure, CSS for style, and JavaScript for interactivity.

📚 12.2 HTTP Protocol

Hypertext Transfer Protocol, the foundation of any data exchange on the Web.

📚 12.3 REST API

Representational State Transfer, a style of software architecture for distributed hypermedia systems such as the World Wide Web.

📚 12.4 Web Security (XSS, CSRF, SQL Injection)

Various techniques to protect web applications, websites and servers from threats.

📚 12.5 Responsive Design

An approach to web design that makes web pages render well on a variety of devices and window or screen sizes.

👩‍💻 13. Version Control

📚 13.1 Git

A distributed version control system for tracking changes in source code during software development.

📚 13.2 GitHub

A hosting service for Git repositories providing a web-based graphical interface.

📚 13.3 Branching and Merging

Key concepts in version control that facilitate concurrent work and track different versions of a project.

📚 13.4 Pull Requests

A method of submitting contributions to an open development project.

📚 13.5 GitFlow

A branching model for Git, created by Vincent Driessen, that allows a team to manage a project more efficiently.

👩‍💻 14. Software Development Methodologies

📚 14.1 Agile

A set of values and principles that encourage flexible responses to changes.

📚 14.2 Scrum

An agile framework for managing knowledge work, with an emphasis on software development.

📚 14.3 Waterfall

A sequential (non-iterative) design process, used in software development processes, in which progress is seen as flowing steadily downwards (like a waterfall).

📚 14.4 Kanban

A scheduling system for lean and just-in-time manufacturing, developed by Toyota.

📚 14.5 Extreme Programming (XP)

A software development methodology which is intended to improve software quality and responsiveness to changing customer requirements.

BY: Waleed Mousa

👩‍💻 15. Concurrency and Multithreading

📚 15.1 Processes and Threads

Processes are instances of programs in execution. Threads are the smallest sequence of programmed instructions that can be managed independently by a scheduler.

📚 15.2 Synchronization

Mechanisms to ensure that two concurrent processes do not interfere with each other.

📚 15.3 Deadlock

A situation where a process is unable to proceed because the resources it needs are being held by another waiting process.

📚 15.4 Starvation

A situation where a process is unable to proceed because the resources it needs are continually given to other processes.

📚 15.5 Race Conditions

An undesirable situation that occurs when a device or system attempts to perform two or more operations at the same time, but because of the nature of the device or system, the operations must be done in the proper sequence.

👩‍💻 16. Network Programming

📚 16.1 TCP/IP

Transmission Control Protocol/Internet Protocol is a set of communication protocols used in the internet and similar networks.

📚 16.2 Sockets

One endpoint of a two-way communication link between two programs running on the network.

📚 16.3 Protocols (HTTP, FTP, SMTP)

A set of rules governing the exchange or transmission of data between devices.

📚 16.4 Firewalls

A network security system that monitors and controls incoming and outgoing network traffic based on predetermined security rules.

📚 16.5 DNS (Domain Name System)

The phonebook of the Internet. Humans access information online through domain names, like nytimes.com or espn.com. Web browsers interact through Internet Protocol (IP) addresses.

BY: Waleed Mousa

👩‍💻 17. Design Patterns

📚 17.1 Singleton

A design pattern that restricts the instantiation of a class to a single instance.

📚 17.2 Factory

A design pattern that provides an interface for creating objects in a superclass, but allows subclasses to alter the type of objects that will be created.

📚 17.3 Observer

A design pattern in which an object, named the subject, maintains a list of its dependents, called observers, and notifies them automatically of any state changes.

📚 17.4 Decorator

A design pattern that allows behavior to be added to an individual object, dynamically, without affecting the behavior of other objects from the same class.

📚 17.5 Strategy

A design pattern that enables selecting an algorithm at runtime.

👩‍💻 18. Security Concepts

📚 18.1 Encryption

The method by which information is converted into secret code that hides the information's true meaning.

📚 18.2 Hashing

A function that converts an input (or 'message') into a fixed-size string of bytes.

📚 18.3 Authentication vs. Authorization

Authentication is the process of verifying who you are. Authorization is the process of verifying what you have access to.

📚 18.4 Public Key Infrastructure

A set of roles, policies, and procedures needed to create, manage, distribute, use, store & revoke digital certificates and manage public-key encryption.

📚 18.5 Secure Sockets Layer/Transport Layer Security (SSL/TLS)

Cryptographic protocols designed to provide communications security over a computer network.

👩‍💻 19. Operating System Concepts

📚 19.1 Process Management

The activity of managing the operation of a process within an operating system.

📚 19.2 File Systems

Methods and data structures that an operating system uses to keep track of files on a disk or partition.

📚 19.3 Memory Management

The function responsible for managing the primary memory in a computer system.

📚 19.4 Device Management

The process of managing the usage of the hardware resources allocated to a virtual machine.

📚 19.5 Interprocess Communication

The mechanisms an operating system provides to allow the processes to manage shared data.

👩‍💻 20. Software Engineering Principles

📚 20.1 SOLID Principles

A set of principles for designing flexible and maintainable object-oriented systems.

📚 20.2 DRY Principle

"Don't Repeat Yourself" - a principle aimed at reducing repetition of software patterns.

📚 20.3 YAGNI Principle

"You Aren't Gonna Need It" - a principle of extreme programming (XP) that states a programmer should not add functionality until deemed necessary.

📚 20.4 KISS Principle

"Keep It Simple, Stupid" - a design principle noting that systems perform best when they have simple designs rather than complex ones.

📚 20.5 Law of Demeter

A design guideline that suggests a module should not know about the innards of the objects it manipulates.

👩‍💻 21. Cloud Computing

📚 21.1 Infrastructure as a Service (IaaS)

Online services that provide high-level APIs used to dereference various low-level details of underlying network infrastructure.

📚 21.2 Platform as a Service (PaaS)

A category of cloud computing services that provides a platform allowing customers to develop, run, and manage applications without the complexity of building and maintaining the infrastructure.

📚 21.3 Software as a Service (SaaS)

A software licensing and delivery model in which software is licensed on a subscription basis and is centrally hosted.

📚 21.4 Serverless Architecture

A software design pattern where applications are hosted by a third-party service, eliminating the need for server software and hardware management by the developer.

📚 21.5 Containerization (Docker, Kubernetes)

An OS-level virtualization method used to deploy and run distributed applications without launching an entire virtual machine (VM) for each app.