

## Module 1: Fundamental for Beginners

### Introduction to Programming

- Programming involves giving computers instructions to automate tasks, solve problems, and create software. Learning programming opens career opportunities and improves logical thinking.
- Programming languages are like special instructions for computers. They can be high-level (easier to learn, like JavaScript or Python) or low-level (more complex but offering greater control, like Assembly). High-level languages are commonly used in web development and data analytics, while low-level languages work closely with hardware.
- Algorithmic thinking helps break down problems into simple steps. For example, finding the largest number in a list involves looping through the list and comparing numbers.
- Developers use tools like IDEs (e.g., Visual Studio Code) to write and test code and version control systems to track code changes.

### Working with Variables and Operators

- Variables store values, and identifiers are the names given to these variables. Declaring a variable creates it, while assigning a value puts data into it.
- Accessing a variable before declaring it leads to a reference error, as the program doesn't recognise the variable.
- Variables have data types, like numbers, strings, and Booleans, which define the type of data stored and the operations that can be performed on them.
- The developer console in browsers (accessed via F12) lets you write and run JavaScript code, helping in practice and debugging.
- The "console.log()" function in JavaScript prints values, useful for debugging and viewing outputs.
- Variables in JavaScript can be reassigned to different values, and their data types can change as the program runs.
- Arithmetic operators like "+" are used to perform calculations in JavaScript.
- In languages like JavaScript, data types are assigned automatically during runtime and can change dynamically.

### Working with Control Flows

- Control flows let you execute or repeat instructions based on specific conditions. For instance, data modelling is only performed if data is available; otherwise, the program searches for data. Control flows make programs dynamic by responding to conditions, improving flexibility in how tasks are handled.
- Boolean expressions evaluate to either true or false, guiding the decision-making in your code. They are key to defining conditions in control flows.
- Comparison operators compare values, while logical operators combine or invert Boolean expressions, allowing for more complex decision-making.

- Conditional flows execute code based on conditions, and iterative flows repeat code until conditions change, like using a while loop to continuously search for data.
- A while loop repeats instructions while a condition remains true. If the condition becomes false, the loop stops.
- A flowchart visually represents control flows, showing steps like checking for data and deciding whether to model data or search for it.

### Working with Functions

- Functions are blocks of code designed for specific tasks. They help make programs easier to read and maintain by allowing code reuse.
- To run a function, you call it by its name followed by parentheses. Functions can take inputs (parameters) to perform tasks.
- In JavaScript, functions are defined using the function keyword, followed by a name, parentheses, and a block of code.
- Functions can accept parameters to personalise their behaviour. These are provided inside the parentheses when calling the function.
- Functions can return a value using the return statement, which allows us to use the result in the rest of the program.
- Functions make code more organised, reusable, and easier to understand, playing a crucial role in efficient programming.

### Overview of Web Development

- HTML is the standard markup language used to create the structure of web pages. It uses tags to define elements like headings, paragraphs, links, and images, forming the backbone of a webpage's content.
- CSS is used to control the presentation and layout of HTML elements. It allows developers to apply styles such as colours, fonts, and spacing, ensuring that web pages are visually appealing and user-friendly.
- JavaScript is a programming language that enables interactivity and dynamic behaviour on web pages. It allows developers to manipulate HTML and CSS, respond to user actions, and create features like form validation, animations, and real-time updates.
- Static web pages have fixed content that does not change unless manually updated. They are typically simple and load quickly, making them suitable for sites that do not require user interaction or frequent updates.
- Dynamic web pages display content that can change based on user interactions or other factors. They often use server-side or client-side technologies to fetch and display data, providing a more personalised and interactive user experience.
- This approach involves the server processing requests, fetching data, and sending fully rendered web pages to the client. It handles logic and data fetching on the server, making it suitable for complex applications.

- In this method, the browser fetches data and updates the webpage dynamically using JavaScript. This allows for real-time updates without reloading the page, creating a smoother user experience.
- Frontend development focusses on the client-side of web applications, including the design, layout, and interactivity of the user interface. It ensures that users can easily navigate and interact with the website.
- Backend development deals with server-side logic, database management, and application functionality. It is responsible for processing requests, storing data, and ensuring the application runs smoothly behind the scenes.
- These tools, such as Git, GitHub, and code editors, assist developers in writing, testing, and managing code efficiently. They streamline collaboration, version control, and debugging processes, enhancing overall productivity.

### **Introduction to Full Stack Development**

- Full-stack development involves working on both front-end and back-end technologies, enabling developers to create comprehensive user experiences.
- A full-stack developer designs applications, manages databases, and works across the entire technology stack, making them versatile and valuable in the development process.
- Front-end development focusses on the client side using HTML, CSS, and JavaScript, while back-end development handles server logic, database interactions, and data processing.
- Key technologies include HTML for structure, CSS for styling, and JavaScript for interactivity, with popular frameworks like Bootstrap and React enhancing development.
- Back-end development utilises server-side languages like Python and JavaScript, often using frameworks and connecting to databases such as MySQL and MongoDB.
- A tech stack is the combination of technologies used in full-stack development. Popular stacks like MERN, MEAN, and LAMP integrate well, facilitating support and development.
- The MERN stack, which includes MongoDB, Express.js, React, and Node.js, is a widely-used tech stack for efficient and scalable application development.
- Full-Stack Development offers versatility, improved efficiency, and cost-effectiveness, as developers can handle both front-end and back-end tasks.
- The role requires broad knowledge of multiple technologies and managing the complexity of both front-end and back-end tasks, which can be demanding.
- A solid foundation in HTML, CSS, and JavaScript is essential, followed by learning frameworks and database technologies, with hands-on projects to gain experience.

**THE END**

*My notes...*