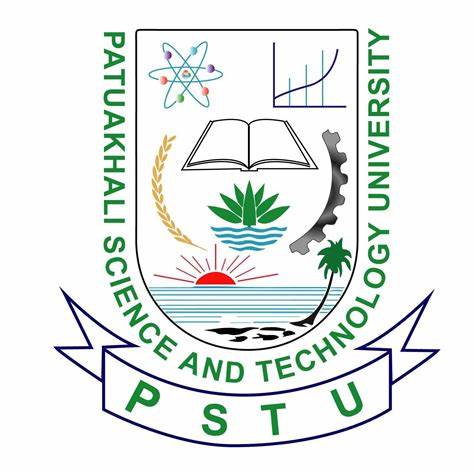
**PATUALHALI SCIENCE AND TECHNOLOGY UNIVERSITY**



# 

Course Code: CCE-112

### **SUBMITTED TO:** **Prof.** **Dr. Md Samsuzzaman Sobuz**

### **Department of Computer and Communication Engineering**

**Faculty of Computer Science and Engineering**

**SUBMITTED BY:**

Name: MD. MEHEDI HASAN

ID : 2102016

Registration No: 10143

Faculty of Computer Science and Engineering

**Date of submission: 03/03/2023**

**Discuss About Program, Coding, Programmer, Programming language**

**Program:** A program is a set of instructions that tell a computer what to do. It is created using a programming language and can perform a variety of tasks, from simple calculations to complex data processing and analysis. Programs can be used to automate tasks, solve problems, and create new products and services.

**Coding:** Coding is the process of writing code or instructions in a programming language that a computer can understand and execute. Coding involves translating human-readable instructions into machine-readable instructions that a computer can execute.

**Programmer:**A programmer is a person who writes code or instructions in a programming language to create software applications, games, websites, and other computer programs. Programmers use programming languages and tools to design, code, test, and debug software.

**Programming language:**A programming language is a set of instructions or rules that a programmer uses to write code or instructions in a format that a computer can understand and execute. There are many programming languages, such as Python, Java, C++, JavaScript, Ruby, and many more.

Each language has its own syntax, rules, and features that make it suitable for different types of applications and programming tasks.



In programming, the combination of five basic parts are used to create algorithms and programs that can solve problems and perform tasks. These five basic parts are:

**Input:** This is the data or information that is received by the program from an external source, such as a user or another program. Examples of input include user input via a keyboard or mouse, data from a file or database, or input received from a network connection.

**Output:** This is the result or response generated by the program based on the input provided. Examples of output include displaying text on a screen, generating a report, or sending data to a remote server.

**Math:** This involves performing mathematical calculations or operations on the input data to generate the desired output. Examples of math operations include addition, subtraction, multiplication, and division.

**Condition execution:** This involves making decisions based on the input data and executing different sections of code based on those decisions. This is often done using if-else statements or switch statements. For example, a program might check if a certain condition is true and execute one set of code if it is, and another set of code if it is not.

**Repetition:** This involves repeating a section of code multiple times, either a set number of times or until a certain condition is met. This is often done using loops, such as while loops or for loops. For example, a program might repeat a section of code until a user enters valid input.

**The Basic Coding Concepts**

**Variables:** Variables are used to store values in a program, which can be accessed and modified throughout the program. They are declared with a name and a data type, such as integer, float, string, or Boolean.

**Conditions:** Conditions allow programs to make decisions based on whether certain conditions are met. These are typically implemented using if-else statements or switch statements.

**Data types and data structures:** Data types define the type of data that can be stored in a variable. Common data types include integers, floats, strings, and Booleans. Data structures are collections of data, such as arrays, lists, and maps.

**Event handling:** Event handling allows programs to respond to user input or system events. This is typically implemented using event listeners or callback functions.

**Loops:** Loops allow programs to repeat a block of code multiple times. Common types of loops include for loops, while loops, and do-while loops.

Describe Run Algorithm, Script, Function and operator in Programming

**Describe Run Algorithm, Script, Function and operator in Programing**

In programming, there are several important concepts that are essential to understanding how code works. Here are the definitions of four key programming terms: Run Algorithm, Script, Function, and Operator.

**Run Algorithm:** An algorithm is a set of instructions or steps that are followed to solve a problem or complete a task. Running an algorithm means executing those instructions in the order they are written to produce an outcome. In programming, algorithms are used to solve complex problems, such as sorting data, searching for information, or processing user input.

**Script:** A script is a series of instructions or commands written in a programming language. Scripts are used to automate tasks or perform actions on data. For example, a script might be used to download data from a website, process that data, and then upload the results to a different location.

**Function:** A function is a block of code that performs a specific task. Functions are used to break down complex tasks into smaller, more manageable pieces. They can take input parameters and return output values, making them flexible and reusable. Functions are a fundamental building block of most programming languages.

**Operator:** An operator is a symbol or keyword that performs a specific operation on one or more values. For example, the addition operator (+) adds two values together, while the assignment operator (=) assigns a value to a variable. Operators are used extensively in programming to perform mathematical calculations, manipulate data, and control program flow.

**Why We Use Scratch Programming**

Scratch programming is a visual programming language developed by MIT that is designed to teach coding concepts to children and beginners in a fun and interactive way. Here are some reasons why Scratch programming is popular:

**It's easy to learn:** Scratch is an intuitive visual programming language that uses block-based coding, which means you don't have to type out code. Instead, you can drag and drop blocks of code to create programs.

**It's fun:** Scratch is designed to be engaging and fun, with colorful graphics, animations, and sound effects. Children and beginners can create their own interactive stories, games, and animations.

**It teaches computational thinking:** Scratch helps children and beginners develop problem-solving skills, logical reasoning, and computational thinking. They learn how to break down complex problems into smaller, more manageable parts and how to use algorithms to solve them.

**It's free:** Scratch is open-source software that is available for free, which makes it accessible to everyone.

**It's a stepping stone to other programming languages:** Scratch is a great introduction to programming concepts and can be used as a stepping stone to other programming languages such as Python, JavaScript, and C++.

Overall, Scratch programming is an excellent way for children and beginners to learn the basics of programming and develop valuable computational thinking skills.

**What Can Scratch Programming Do?**

Scratch programming is a visual programming language that allows users to create interactive stories, animations, games, and simulations. Here are some things that Scratch can do:

**Create Animation:** Scratch has a library of sprites (characters) and backgrounds that can be used to create animations. Users can also import their own images and create their own animations using the built-in animation tools.

**Build games:** With Scratch, users can create a variety of games, including platformers, puzzles, and arcade-style games. They can use the built-in game mechanics and physics engines to add movement and interactivity to their games.

**Tell stories:** Scratch allows users to create interactive stories with characters, dialogue, and plot. They can use the built-in text-to-speech feature to add voice-over to their stories.

**Simulate systems:** Scratch can be used to create simulations of real-world systems, such as weather patterns, traffic flow, and animal behavior. Users can use programming concepts like loops, variables, and conditionals to simulate complex systems.

**How Do You Install Scratch In Your Computer**

To install Scratch on your computer as a student, follow these steps:

1. Go to the Scratch website: <https://scratch.mit.edu/>.
2. Click on the "Create" button in the top right corner of the screen.
3. Click on the "Download" button in the top right corner of the screen.
4. Scroll down to the "Scratch Desktop" section and choose the appropriate version of Scratch for your operating system (Windows, Mac, or Linux).
5. Click on the download link for the version you need.
6. Once the download is complete, open the installer file. On Windows, it should be a .exe file; on Mac, it should be a .dmg file.
7. Follow the on-screen instructions to install Scratch on your computer. This may involve selecting a location for the installation, agreeing to the terms of service, and entering your computer's administrator password (if prompted).
8. After the installation is complete, you should be able to launch Scratch from your computer's applications or programs folder. On Windows, it may also create a shortcut on your desktop.
9. When you launch Scratch for the first time, you may be prompted to create a Scratch account or log in with an existing account. This is optional, but it allows you to save and share your Scratch projects online.