ROBO DYNAMICS

Build Your Own Robot

Session 1: What is Programming?

What you will learn?

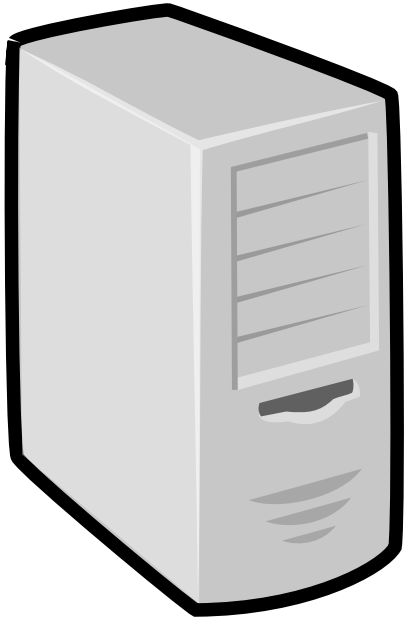
- In this project, we are going to see how traffic lights work?

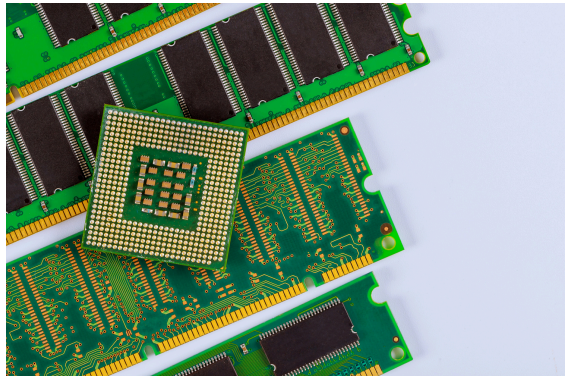
Introduction to Computers and Programming

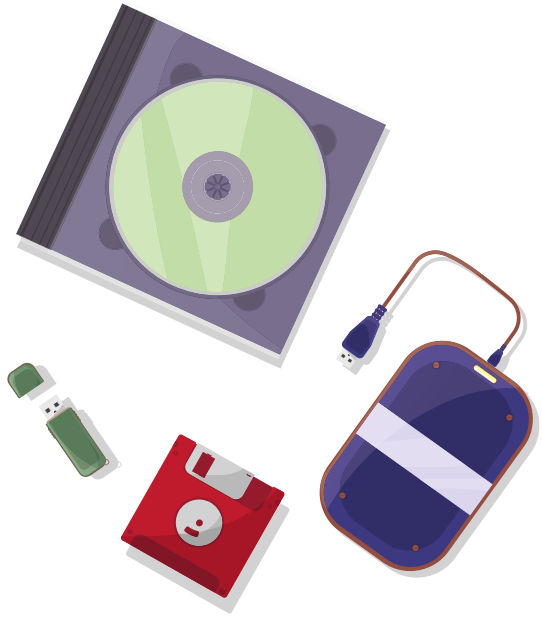
- **Computers**
  - Electronic devices that can perform a variety of tasks by following a set of instructions.

Introduction to Computers and Programming

- **Hardware**
  - The physical components of a computer (e.g., CPU, memory, storage devices)
  -







Introduction to Computers and Programming

- **Software**
  - Programs and operating systems that run on the hardware, enabling it to perform tasks.

Introduction to Computers and Programming

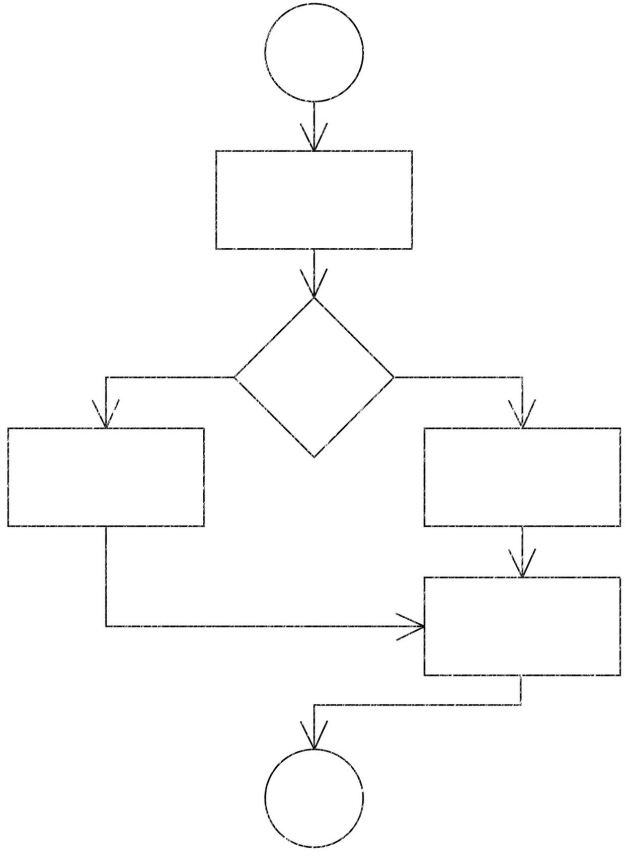
- **Programming**
  - The process of creating software by writing code. A programmer writes instructions in a programming language to solve problems or perform tasks

Basic Concepts of Programming

- **Algorithm**
  - A step-by-step procedure or formula for solving a problem. It’s a plan for solving a problem using a sequence of instructions
- **Example**
  - Recipe for baking a cake. Each step needs to be followed in a specific order to achieve the desired result
-

Basic Concepts of Programming

- **Flowchart**
  - A diagram that represents an algorithm. It uses symbols to show the flow of steps involved in solving a problem
- Symbols
  - **Oval**: Start/End
  - **Rectangle**: Process/Instruction
  - **Diamond**: Decision/Conditional
  - Arrows: Flow of control



Programming Languages

- **What is a Programming Language?**
  - A programming language is a formal language comprising a set of instructions that produce various kinds of output.
  - Used by programmers to communicate with computers and create software.
  -

Programming Languages

-
- **Types of Programming Languages**
  - **Low-Level Languages:** Close to machine code. Examples: Assembly language.
  - **High-Level Languages:** Easier for humans to understand and write. Examples: Python, Java, C**.**
  -

python™

Introduction to C Programming

- **History:**
  - Developed in the early 1970s by Dennis Ritchie at Bell Labs.
- **Importance:**
  - Foundation for many other programming languages. Used in systems programming, game development, and embedded systems.
  - 
  - 

The Role of C Programming

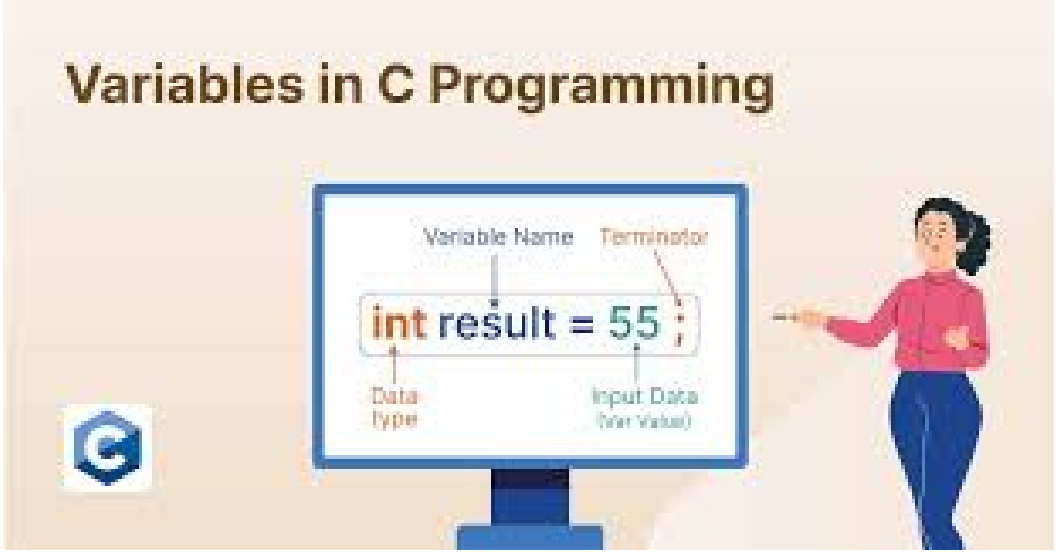
- **Applications of C**
  - System software (e.g., operating systems, compilers).
  - Embedded systems (e.g., microcontrollers, automotive systems).
  - Game development.
  - Application software.
-

The Role of C Programming

- **Why Learn C?**
  - Provides a strong foundation in programming concepts.
  - Helps understand how computers work at a low level.
  - Influences many modern languages (e.g., C++, Java, Python).
  -

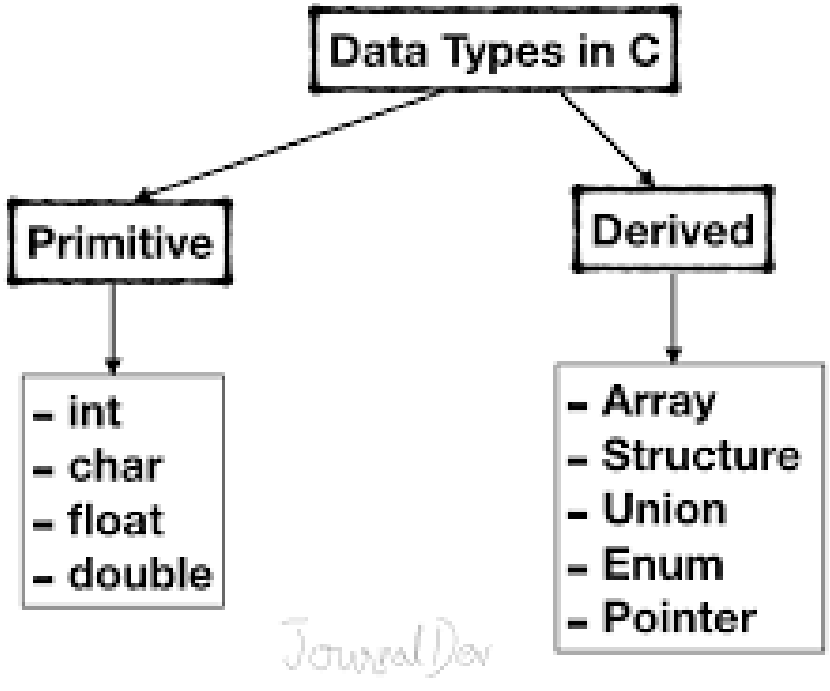
Basic Programming Concepts

- **Variables**:
  - Storage locations in a program with a specific type and value. Example: int age = 10;
-



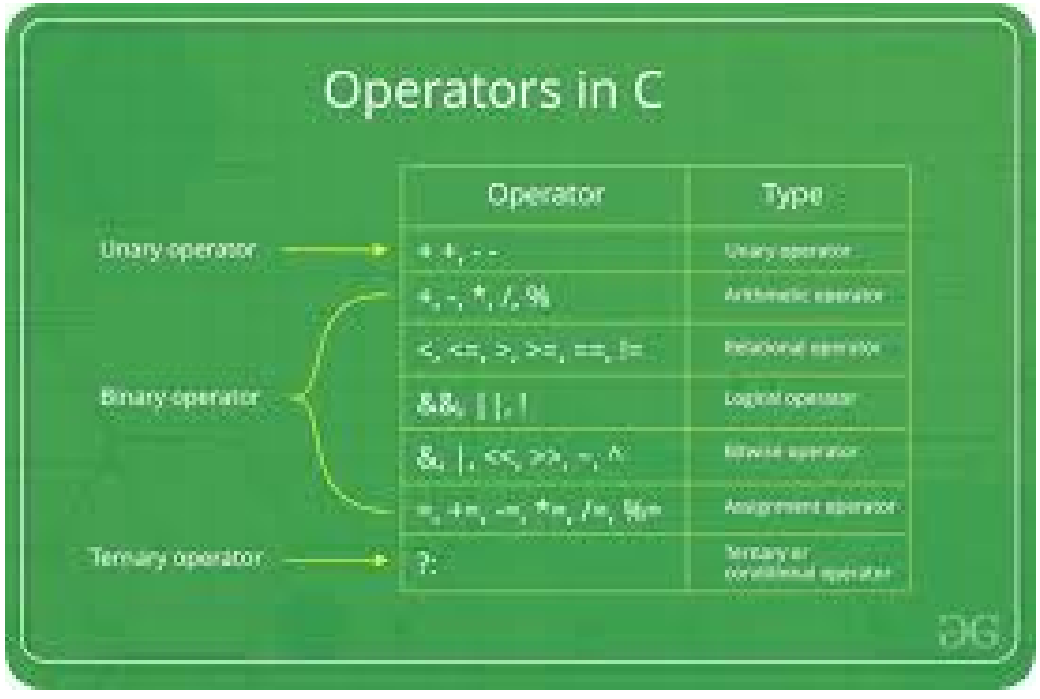
Basic Programming Concepts

- **Data Types:**
  - Types of data that can be stored in variables. Examples: int (integer), char (character), float (floating-point number).
-



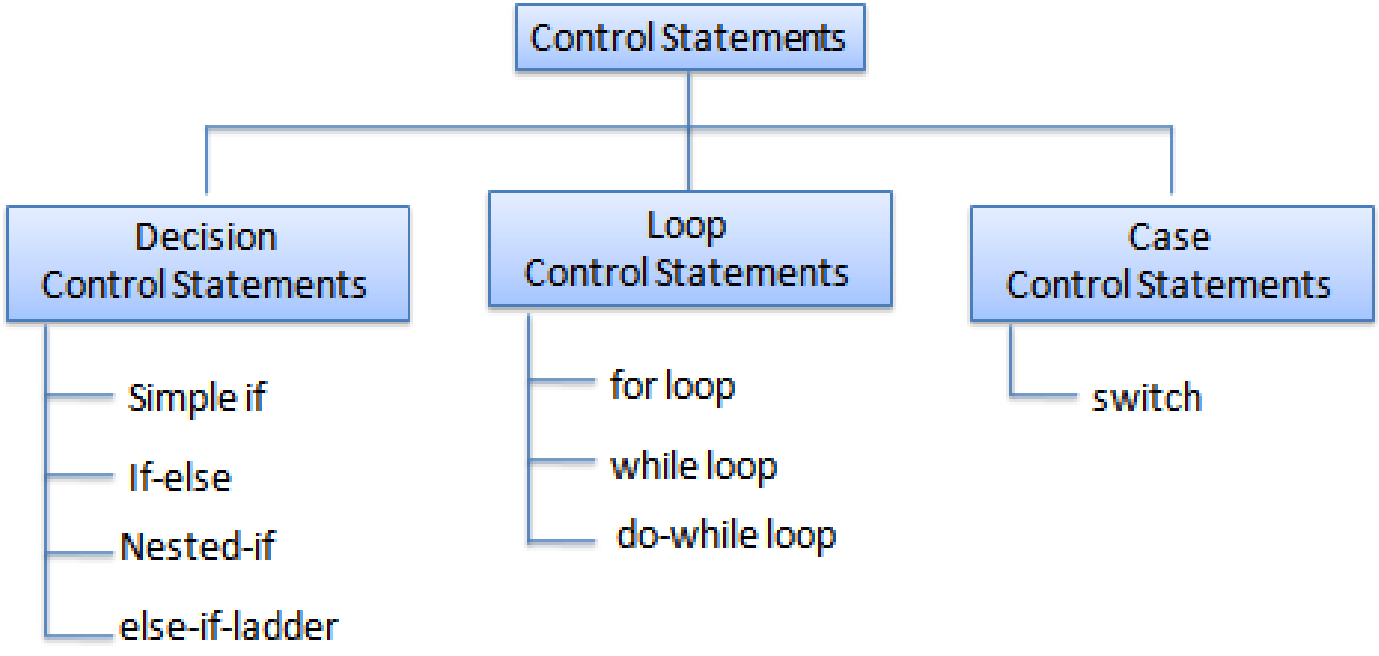
Basic Programming Concepts

- **Operators:**
  - Symbols that perform operations on variables and values. Examples: + (addition), - (subtraction), * (multiplication), / (division).
-



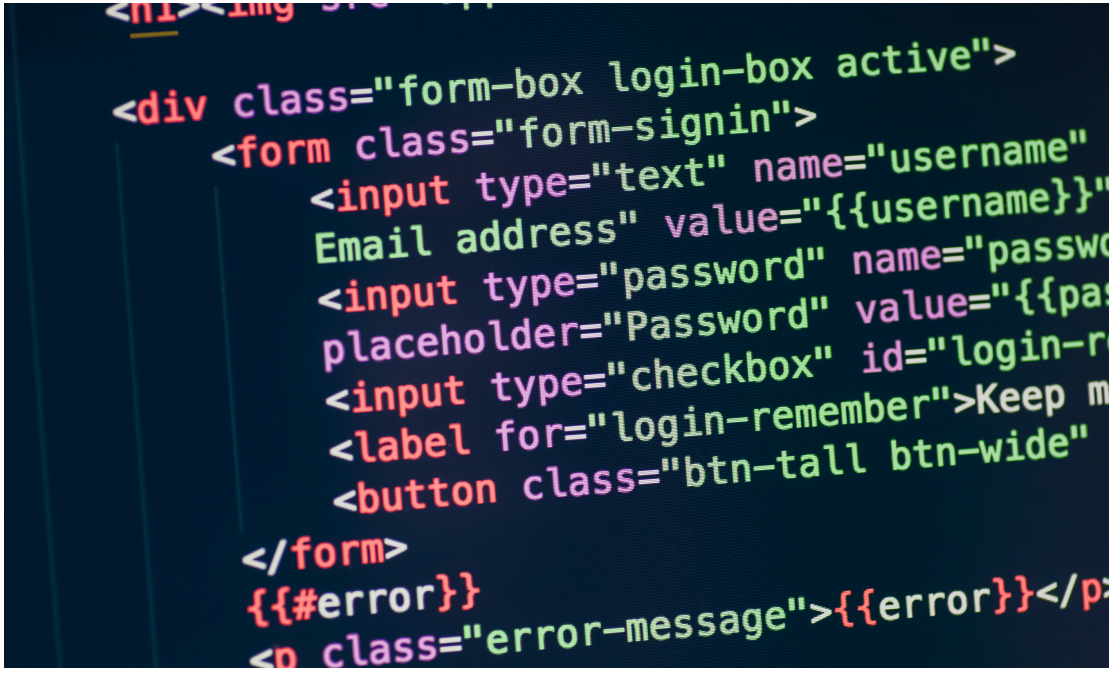
Basic Programming Concepts

- **Control Structures:**
  - Direct the flow of the program. Examples: if statements, loops (for, while).
-



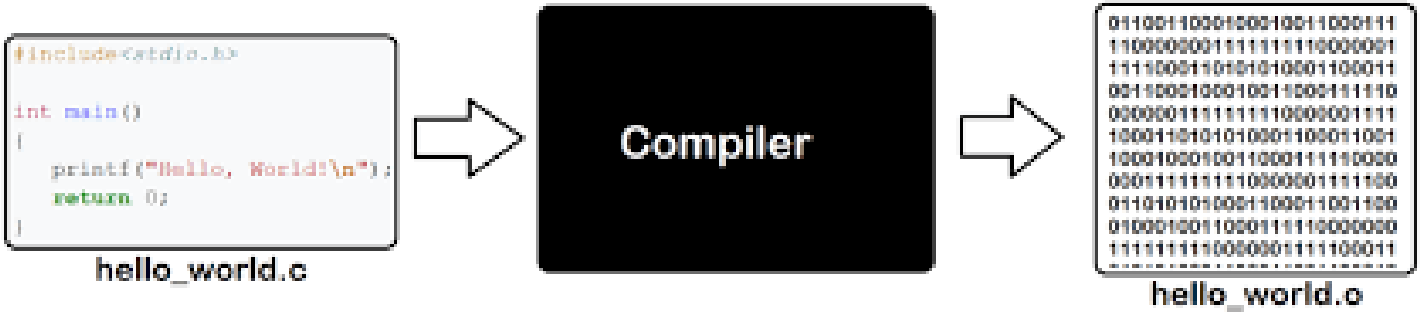
Writing and Running Programs

- **Source Code:**
  - The human-readable code written by the programmer.
-

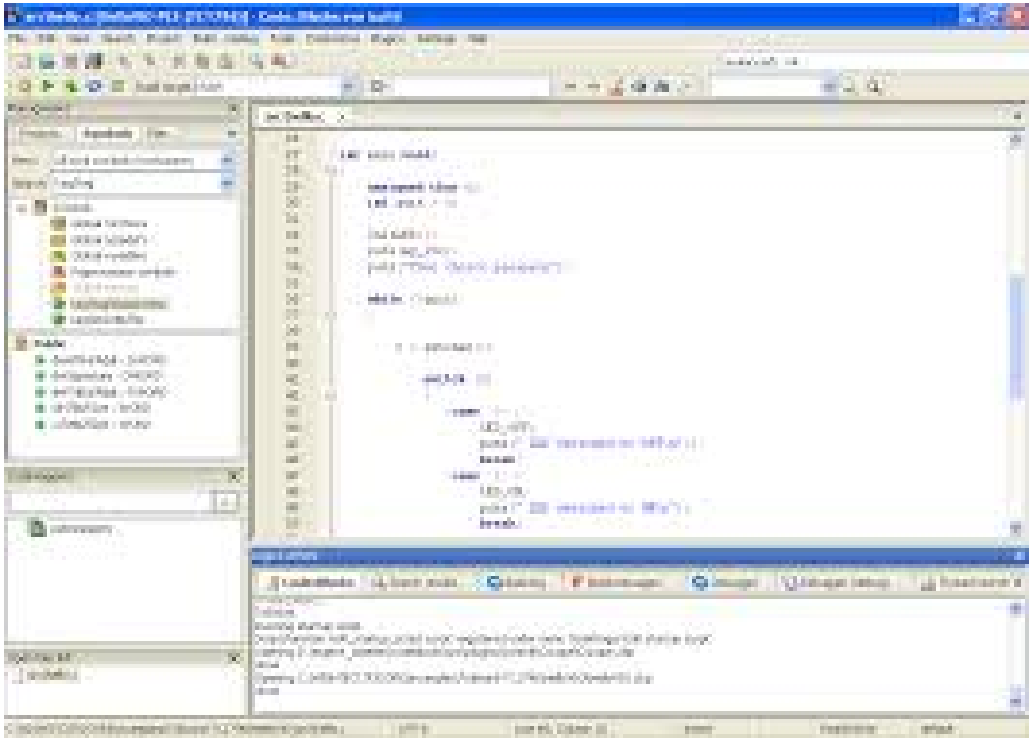


Writing and Running Programs

- **Compiler:**
  - **A tool that converts source code into machine code that the computer can execute.**
-



Writing and Running Programs

- **IDE (Integrated Development Environment):**
  - Software that provides comprehensive facilities to programmers for software development. Examples: Code::Blocks, Dev-C++.
- 

Basic Program Structure

- **Headers**:
  - Include necessary libraries. Example: #include <stdio.h>
- **Main Function:**
  - Entry point of a C program. Example: int main() { ... }
- **Statements:**
  - Instructions within the main function. Example: printf("Hello, World!");

Basic Program Structure

- **Headers**:
  - Include necessary libraries. Example: #include <stdio.h>
- **Main Function:**
  - Entry point of a C program. Example: int main() { ... }
- **Statements:**
  - Instructions within the main function. Example: printf("Hello, World!");

Example of a Simple C Program

```c
#include <stdio.h>

int main() {
    printf("Hello, World!\n");
    return 0;
}
```

Explanation

- **#include <stdio.h>:**
  - Includes the standard input-output library.
- **int main() { ... }:**
  - Main function where the program execution begins.
- **printf("Hello, World!\n");:**
  - Prints "Hello, World!" to the screen.
- **return 0;:**
  - Indicates that the program ended successfully.
-

Learning Programming

- **Practice**:
  - Regular coding practice helps improve programming skills.
- **Problem-Solving:**
  - Programming enhances problem-solving abilities by breaking down complex problems into smaller, manageable tasks.
- **Creativity:**
  - Programming allows for creative solutions and building projects from scratch.
- **Persistence:**
  - Debugging and troubleshooting are essential parts of the learning process.