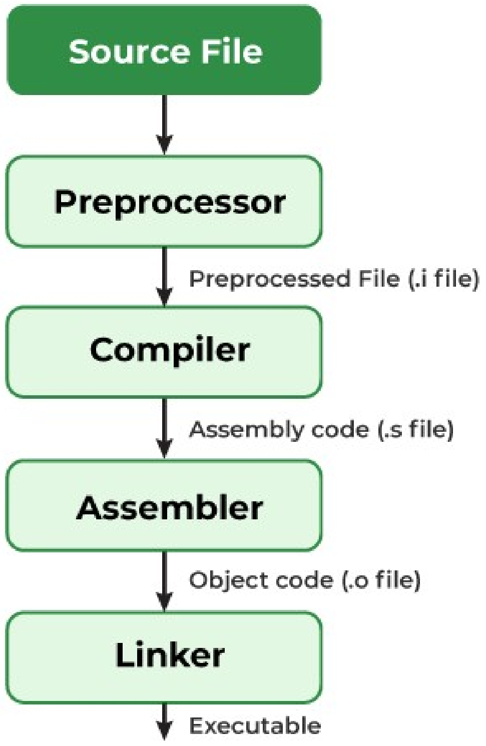
Module 1.3

- Program Execution Process
- Systems Development Life Cycle

SOMAIYA VIDYAVIHAR UNIVERSITY
K J Somaiya School of Engineering
(formerly K J Somaiya College of Engineering)

Program Execution Process

- Introduction
- Overview of stages
- Loading and Execution



Introduction

What is Program Execution?

1. The process of converting human-readable source code into an executable program and running it.
2. Applies to **both C and C++** (though C++ adds OOP features).
3. Involves **multiple stages** before a program runs.

Overview of Stages

1. **Editing** – Writing source code (.c / .cpp files)
2. **Preprocessing** – Handling #include, #define, macros
3. **Compilation** – Translating source code to assembly
4. **Assembly** – Converting assembly to machine code (object files)
5. **Linking** – Combining object files and libraries
6. **Loading & Execution** – Running the program in memory

Overview of Stages

Editing

- Tools: Text editors / IDEs (e.g., Code::Blocks, Visual Studio, Vim)
- Files: .c (C) / .cpp (C++)
- May include header files (.h / .hpp)

Pre-processing

- Preprocessor directives start with #
- Tasks:
- Include header files (#include)
- Macro substitution (#define)
- Conditional compilation (#if, #ifdef, etc.)

Overview of Stages

Compilation

- Converts preprocessed code into **assembly language**.
- Checks for syntax errors and semantic correctness.
- Output: Assembly file (.s)

Assembly

- Assembler converts .s → **Object file** (.o / .obj).
- Machine code but **not yet executable**.

Overview of Stages

Linking

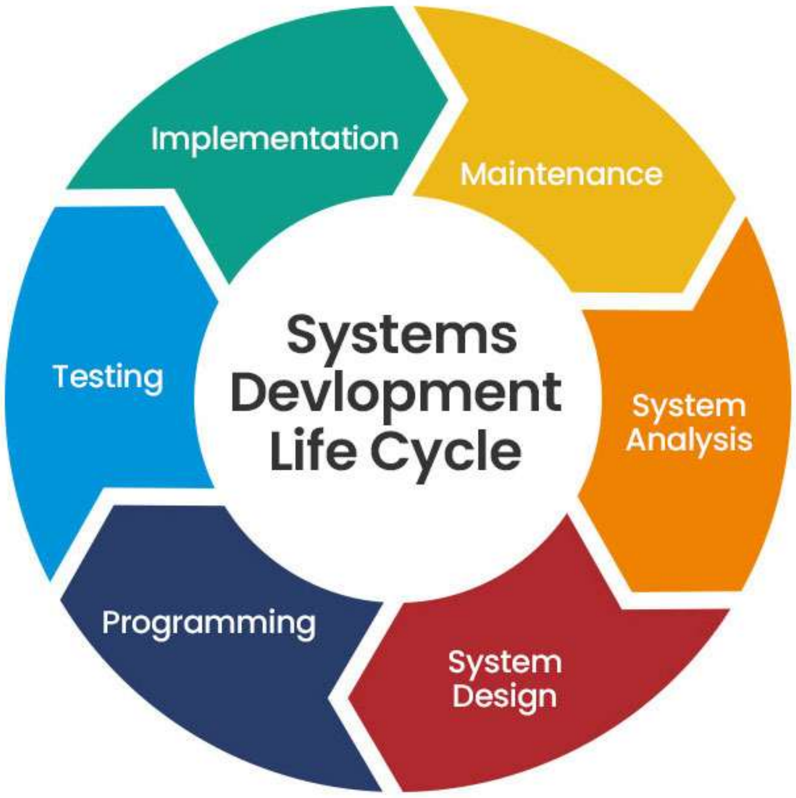
- Combines object files and libraries into a single executable.
- Resolves external references (e.g., library functions like printf()).
- Output: Executable file (.exe on Windows, no extension on Linux).

Loading & Execution

- Loader loads the executable into RAM.
- Allocates memory for:
- Code segment
- Data segment
- Stack
- Heap
- CPU starts execution from the main() function.

SOMAIYA VIDYAVIHAR UNIVERSITY
K J Somaiya School of Engineering
(formerly K J Somaiya College of Engineering)

Systems Development Life Cycle



Systems Development Life Cycle

Systems Development Life Cycle Overview

Definition: A structured approach to developing information systems through a series of well-defined phases

Purpose:

- Provide a systematic framework for system development
- Ensure quality, efficiency, and project success
- Minimize risks and control costs
- Deliver systems that meet user requirements

Key Characteristics:

- Phased approach with defined deliverables
- Systematic and disciplined methodology
- Focus on planning, documentation, and quality assurance
- Stakeholder involvement throughout the process

The Seven Key Phases

1. Planning & Analysis
Define project scope and objectives
Conduct feasibility studies
Identify stakeholders and requirements

2. System Analysis
Gather detailed requirements
Analyze current system limitations
Document functional specifications

3. System Design
Create system architecture
Design user interfaces and databases
Develop technical specifications

4. Implementation (Development)
Code development and programming
Unit testing and integration
System configuration

The Seven Key Phases

5. Testing
System testing and quality assurance
User acceptance testing (UAT)
Performance and security testing

6. Deployment
System installation and go-live
User training and documentation
Data migration

7. Maintenance
Ongoing support and bug fixes
System updates and enhancements
Performance monitoring