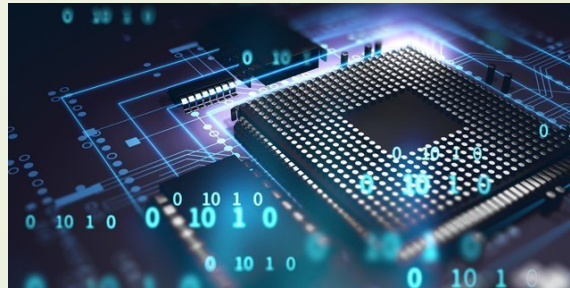


System programming and compiler construction



T.E.(Sem VI)

Shirin Matwankar


Chapter 1

Introduction to System Software

- **Concept of system software**
- **Goals of system software**
- **System program and system programming**
- **Introduction to different system programs**



System software

- **Introduction to system software**
 - **System program**
 - **System programming**
- 

System software

□ Introduction to computer system

- A. View of a student
- B. View of the programmer

A.

USER

User interface

Application
program

OS

Hardware

B.

USER

User
interface

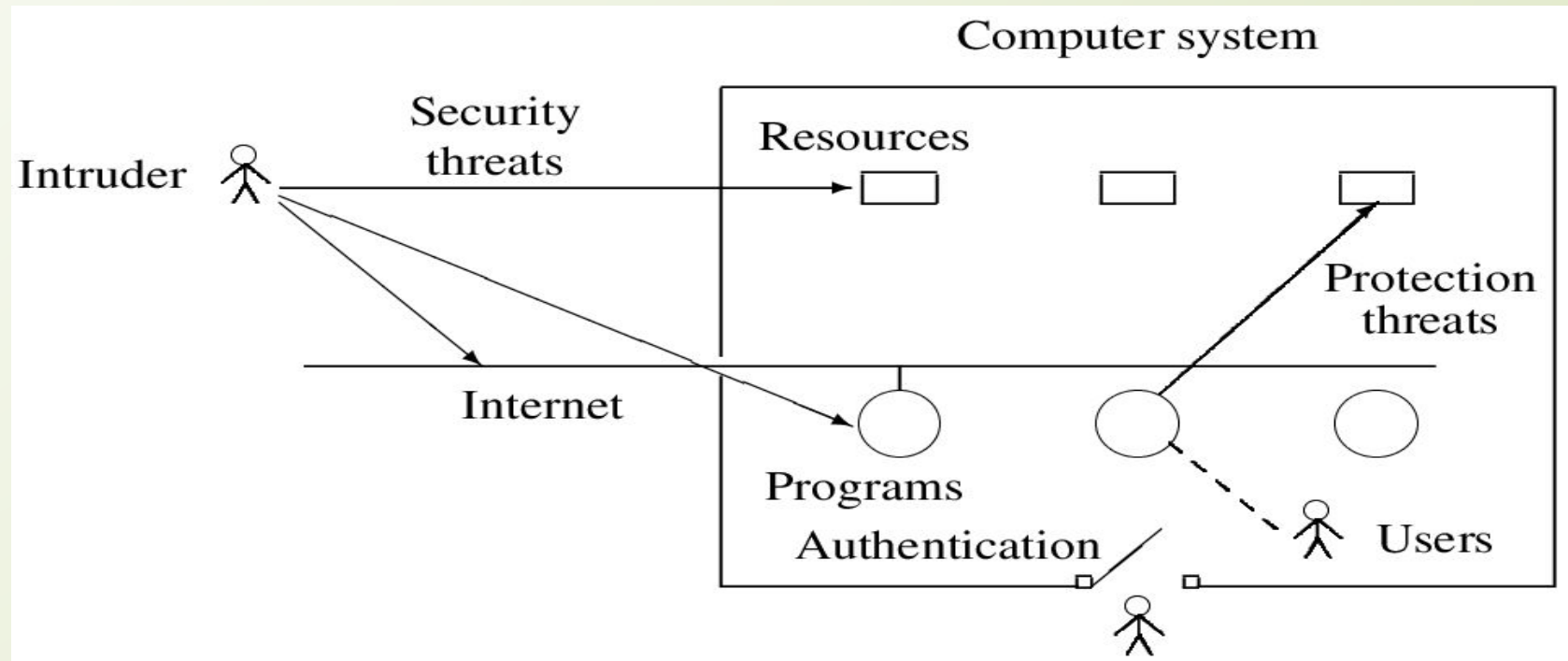
Language
processors

OS

Hardware

System software

- **Goals of system software**
 - **Efficient use**
 - **Non- interference**






□ Non- interference

1.Authentication

2. Internet threats

- Trojan horse**
- Virus**
- worms**

□ User convenience

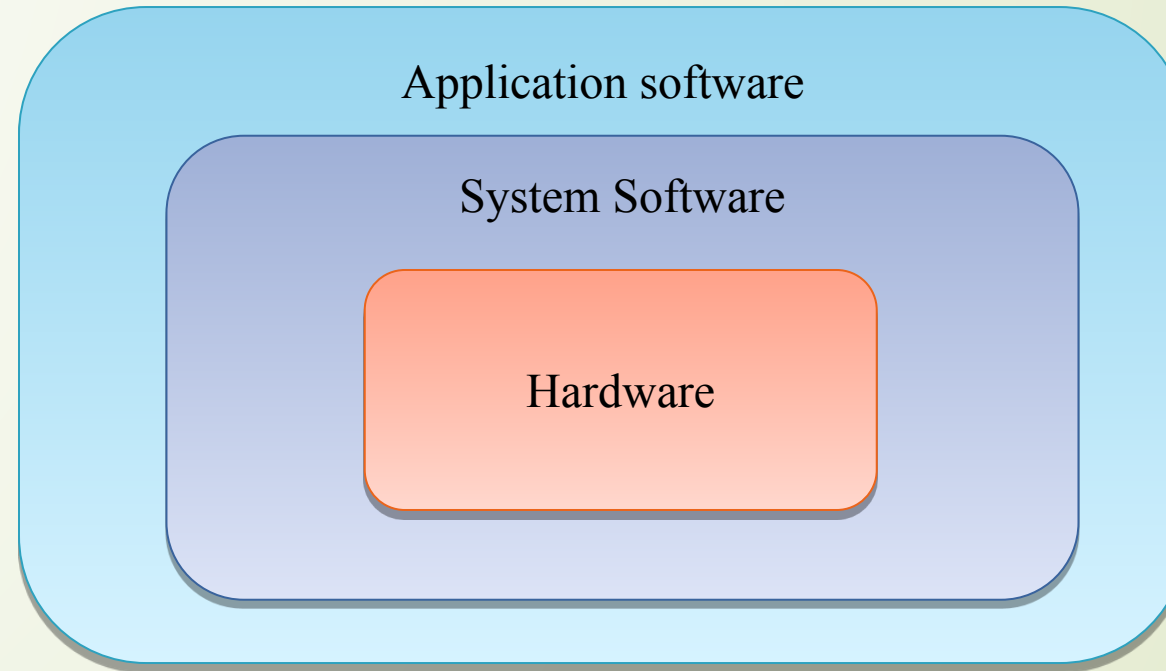
- Fulfillment of necessity**
 - Good service**
 - User friendly interfaces**
 - New programming model**
 - Web oriented features**
 - evolution**
- 

Types of software

❑ Application software

- word processing software (MS word, WordPad, notepad)
- Database software (oracle, MS access)
- Spreadsheet software (MS excel)
- Presentation software (PowerPoint)
- Internet browser (google chrome, Firefox)

❑ System software



- **Operating System (Microsoft Windows, Linux, Mac OS)**
- **Device drivers (printers, scanner, motherboard chipset)**
- **Utility software (antivirus software, file manager, backup software)**



Language programming software

- **Assembler**
- **Compiler**
- **Loader**
- **Linker**
- **Editor**

1. Application domain

It will describe the specification and specify the computation by using terms related to entities and operations.

2. Execution domain

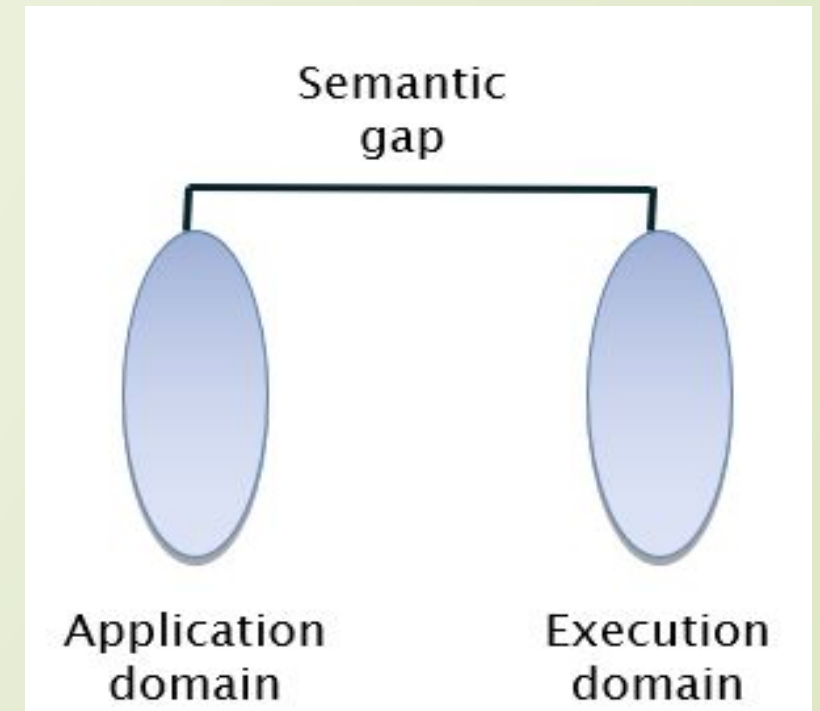
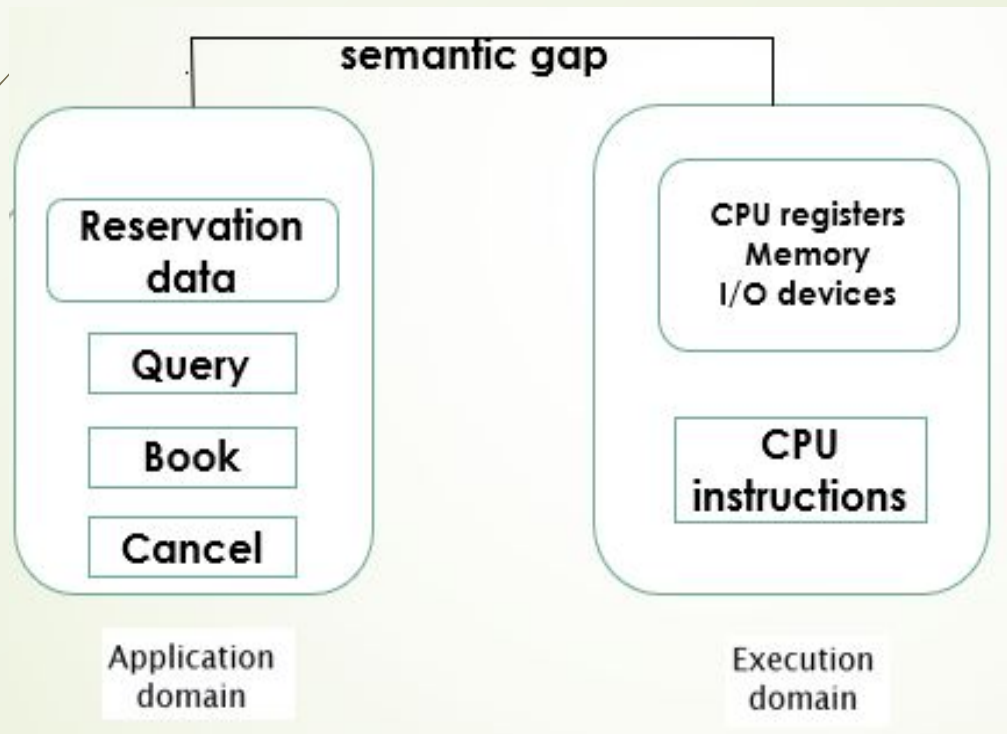
System software will implement the specification in execution domain.

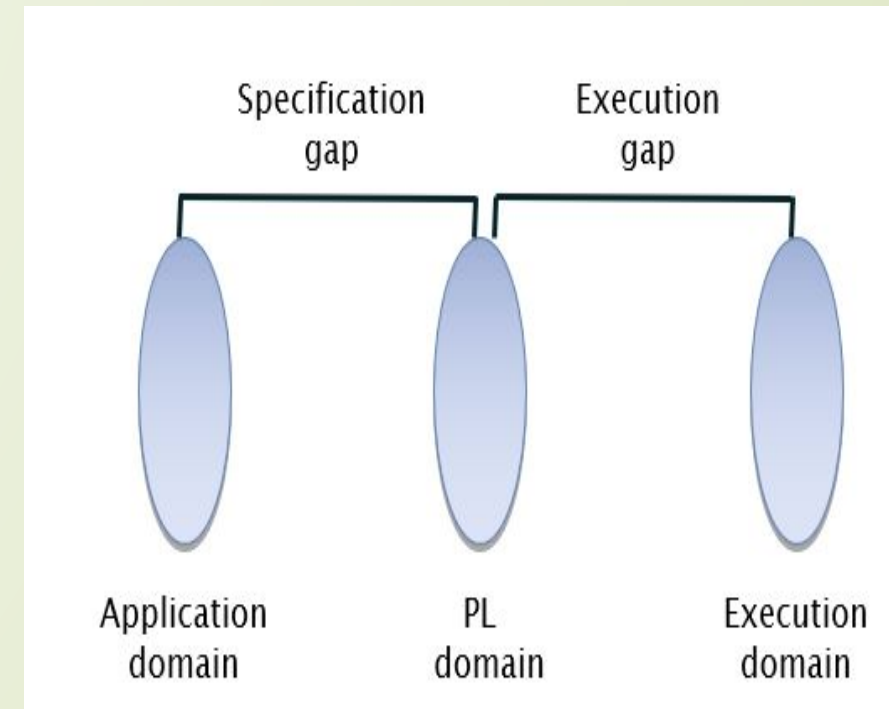
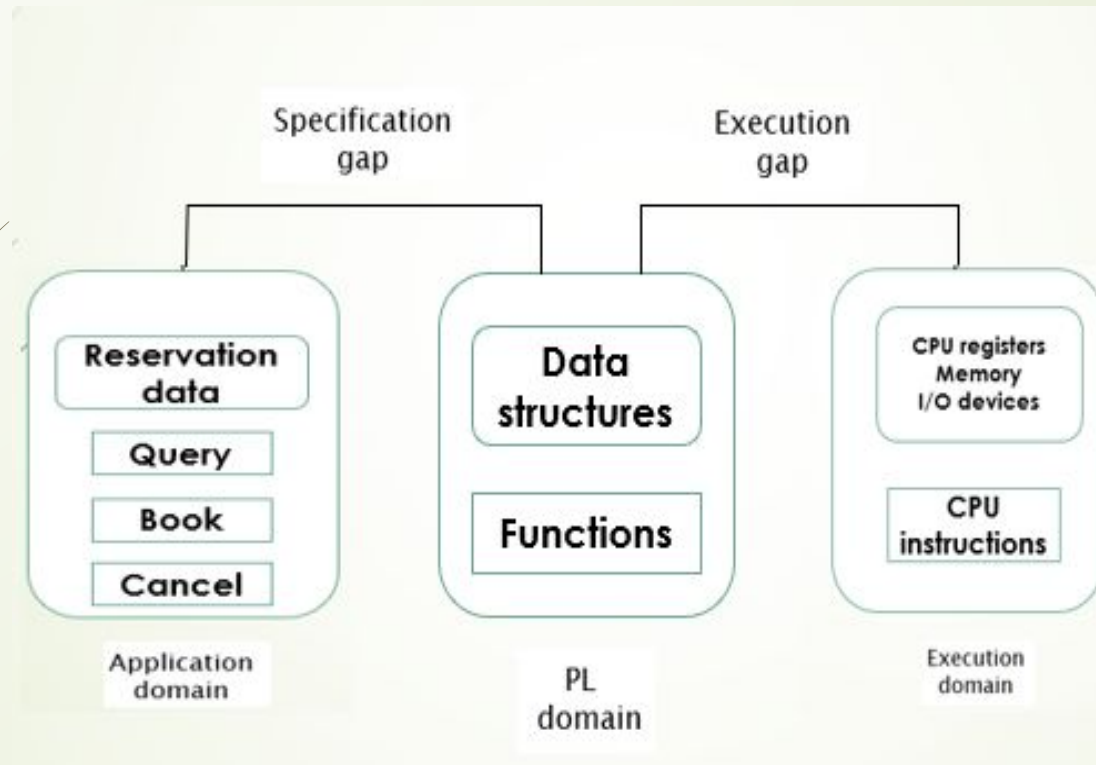
3. Semantic

Semantic represents rules of meaning of domain and indicates order of operations.

4. Semantic gap

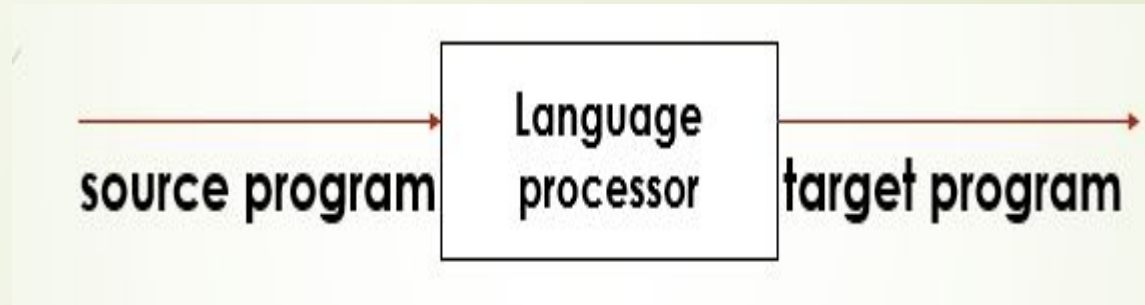
Bridges specification of domains.





Language processor


- A language processor is a software which bridges a specification or execution gap.

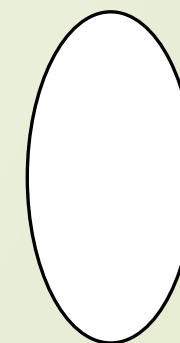
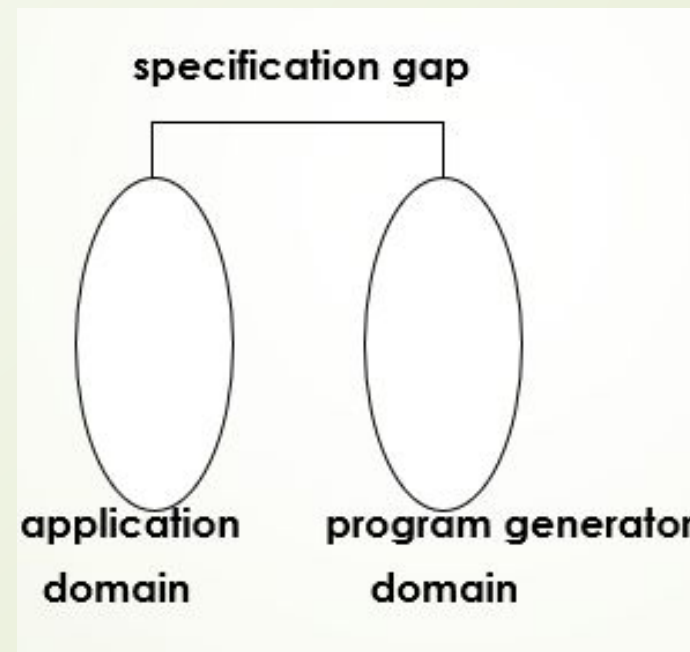
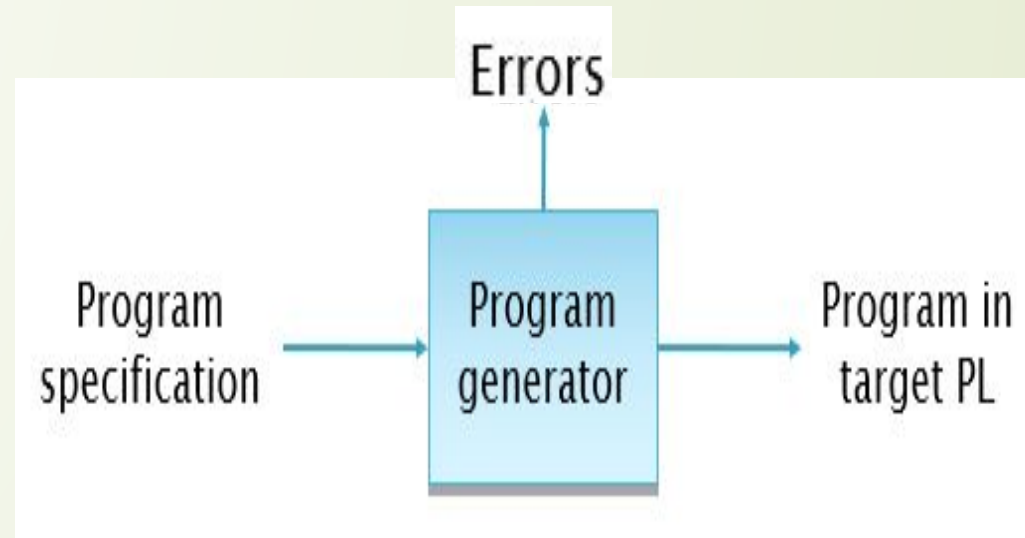


- Fundamental activities divided into those that bridge the specification gap and execution gap.
 - Program generation activities
 - Program execution activities

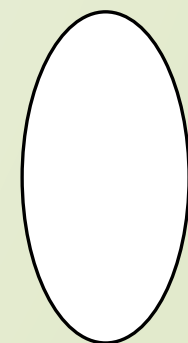


Program generation activities

- A program generation activity aims at automatic generation of a program.
 - A source language is a specification language of an application domain and the target language is procedure-oriented PL.
 - Program generator introduces a new domain between the application and PL domain , call this the program generator domain.
 - Specification gap now between Application domain and program generation domain, reduction in the specification gap increases the reliability of the generated program.
- 

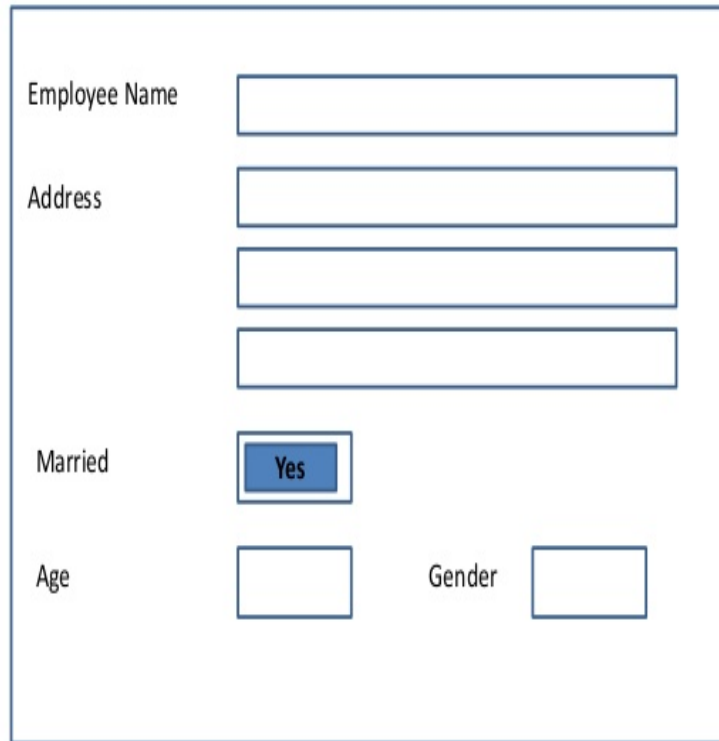


Target PL domain



Execution domain

Program generators for specific application domains



Employee Name

Address

Married

Age Gender

Figure: Screen displayed by a screen handling program

*Employee name : char : start(line=2,position=25)
end(line=2,position=80)*

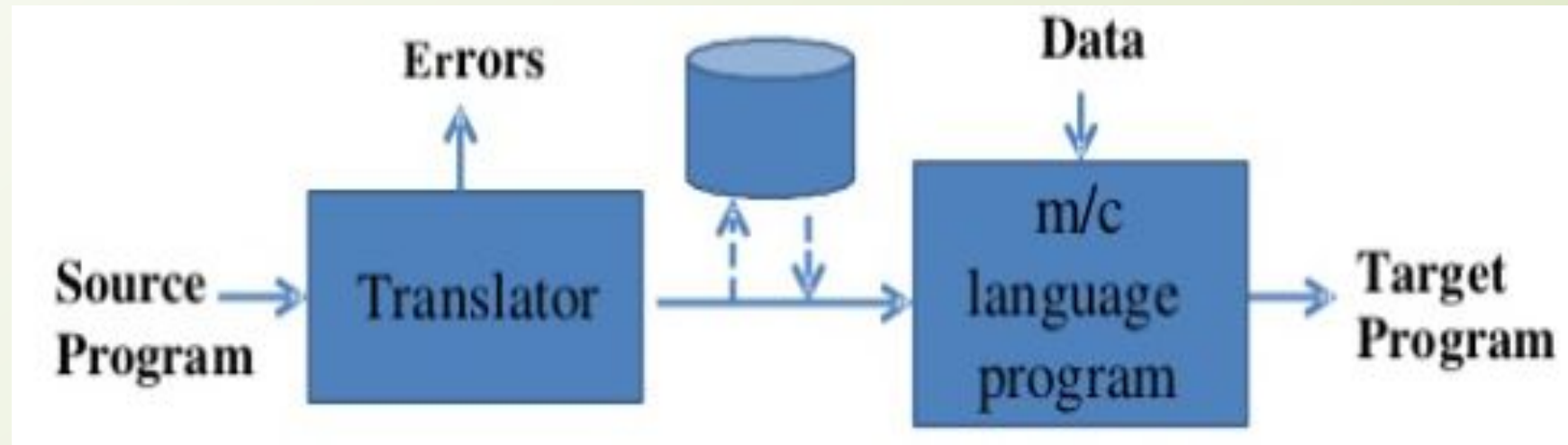
*Married : char : start(line =10, position=25)
end(line=10,position=27)*

Value('yes', 'no') default('Yes')

❑ Program execution

I. Program translation

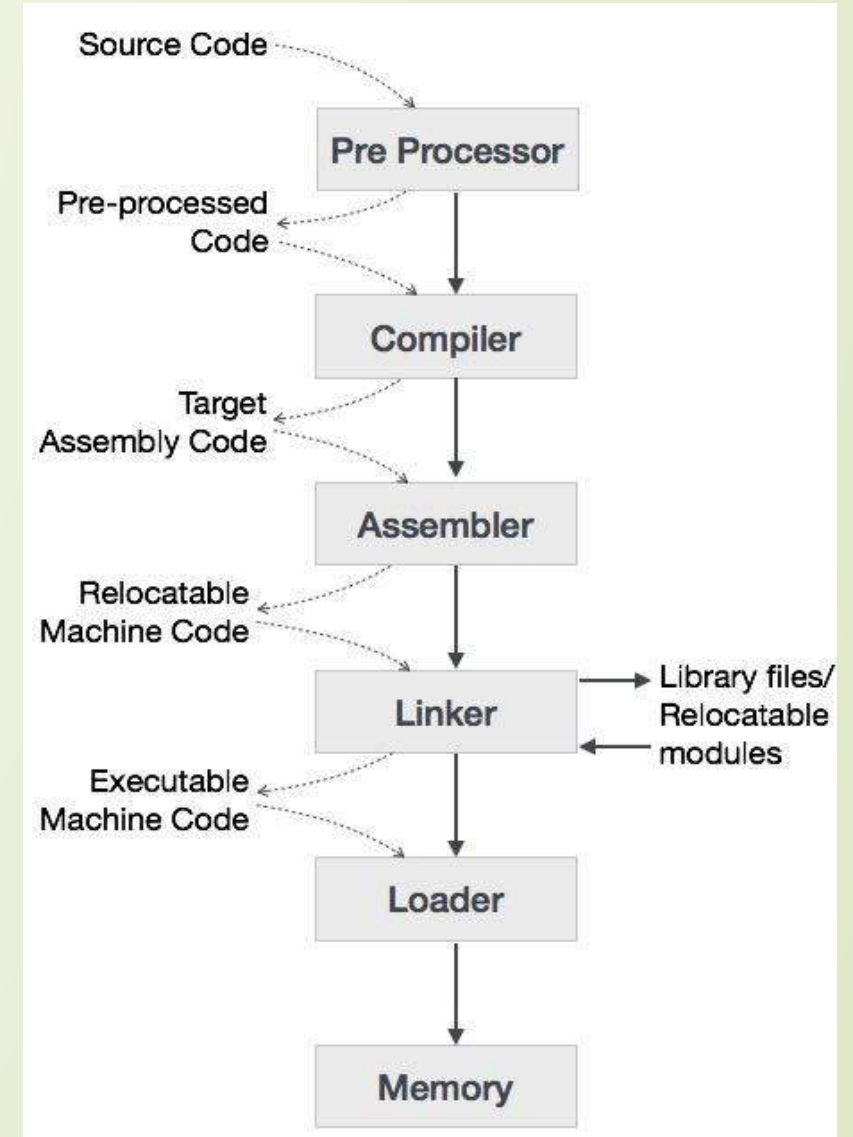
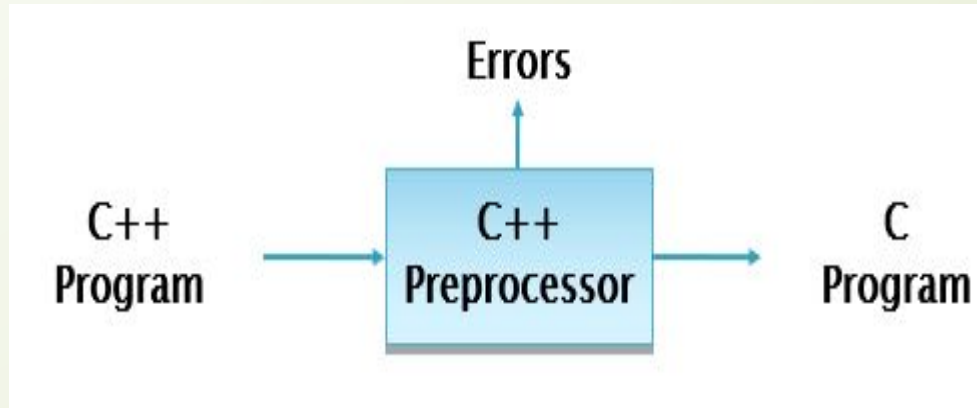
■ Program translator



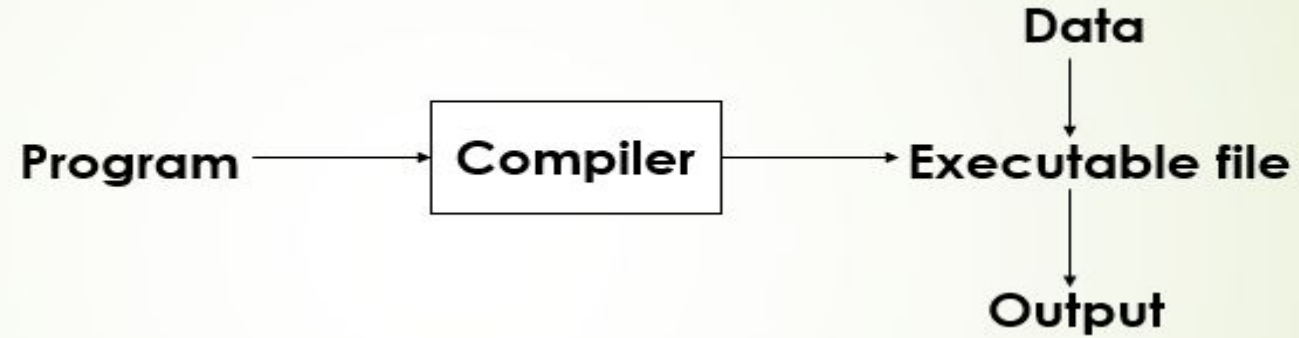
An arrangement of language processors

□ Pre-Processor

□ Example



□ Compiler

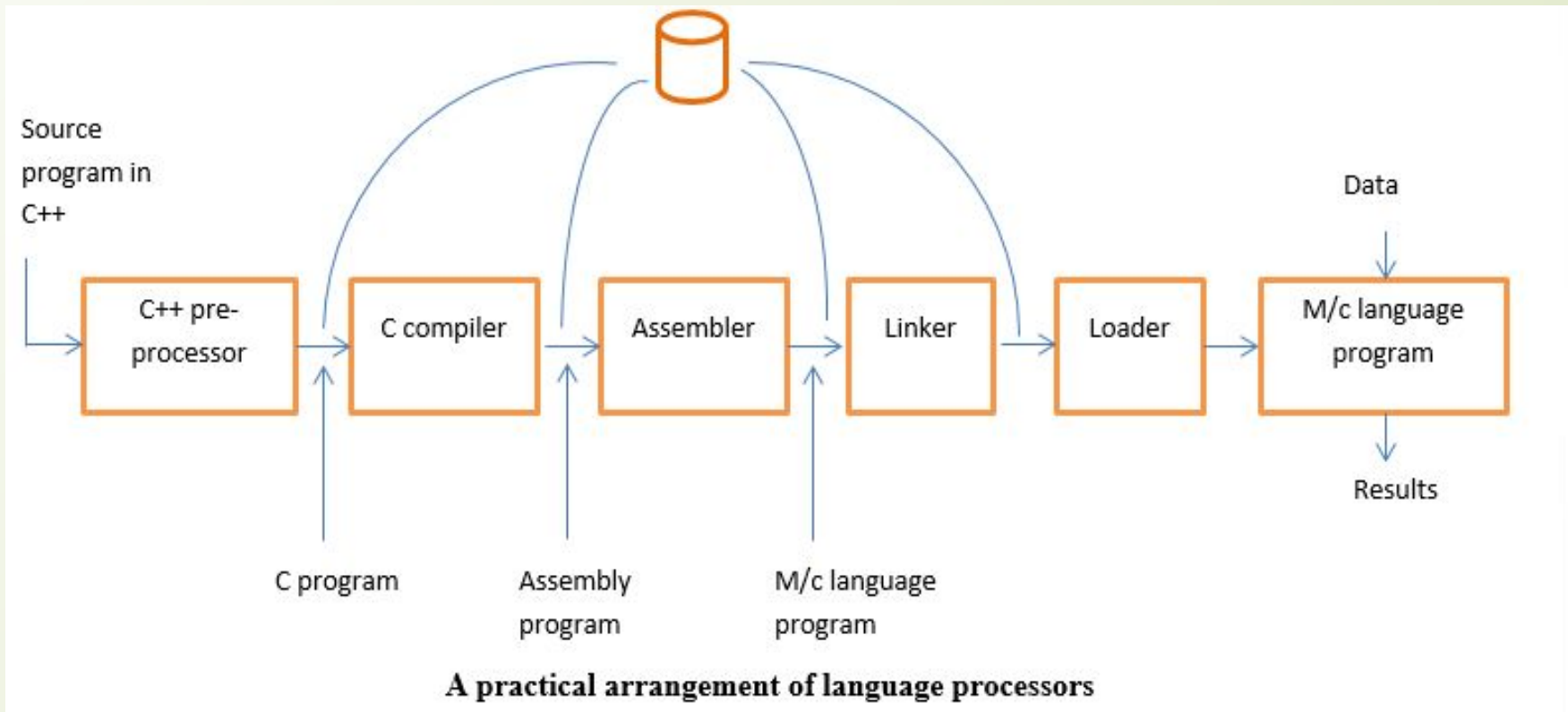


□ Assembler

□ Linker

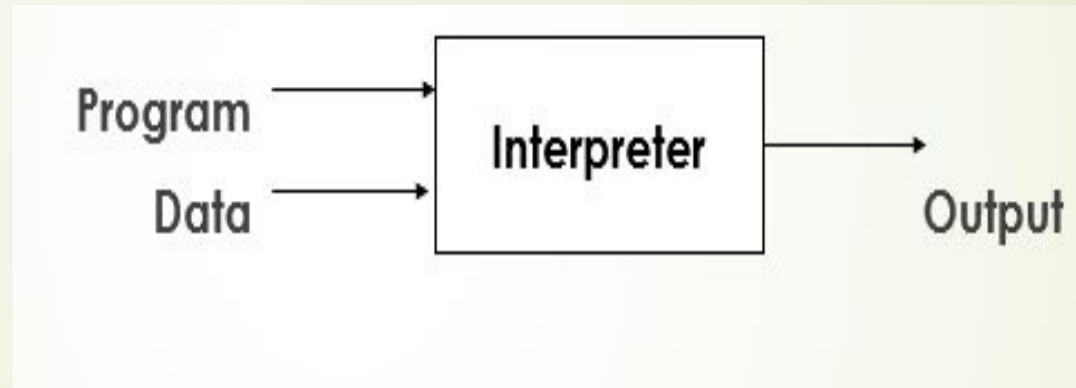
□ Loader

Example

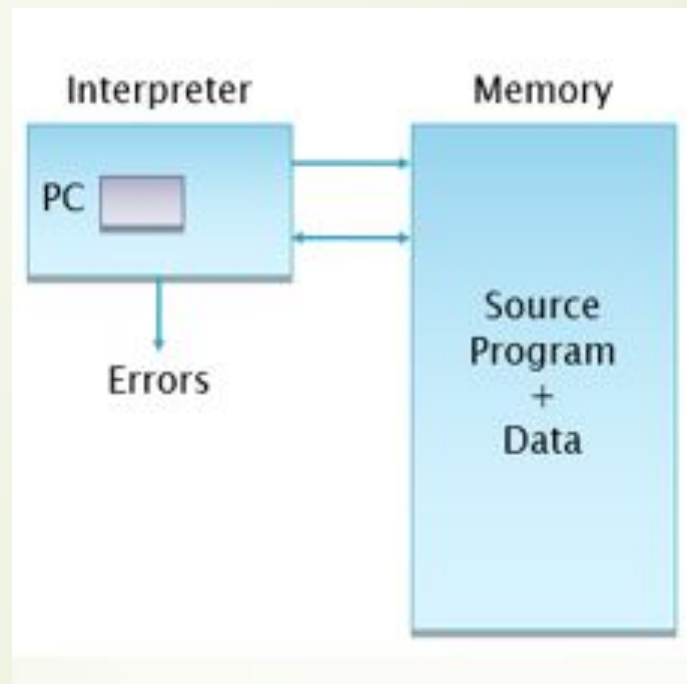


II. Program interpretation

Interpreter



A)



B)

