CT(IF)-21003 Fundamentals OS

Assignments/Quizzes followed by Oral	40 Marks
End Semester Exam	60 Marks

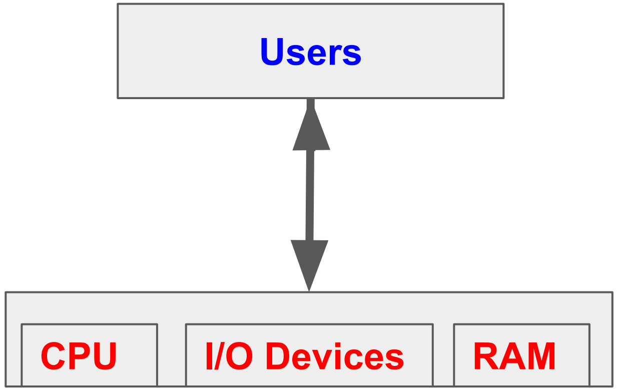
Prof. Harish D.G.
Dept. of Computer and IT
College of Engineering,Pune
www.harishgadade.com

Introduction to Operating Systems

- **Definitions:**
    - Operating Systems is a System Software
    - It Works between the users and computer Hardwares like CPU,I/O Devices, and Memory
    - Operating system is a **Interface** between users and computer Hardwares

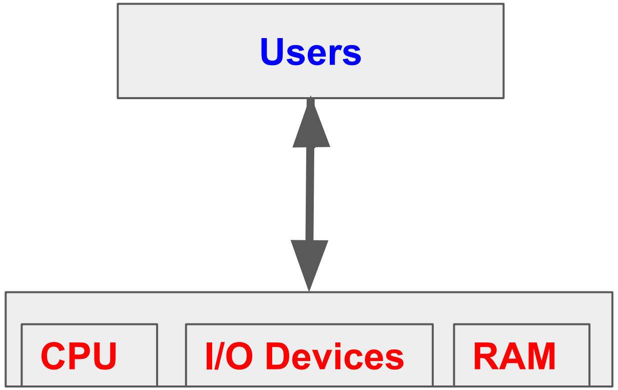
Introduction to Operating Systems

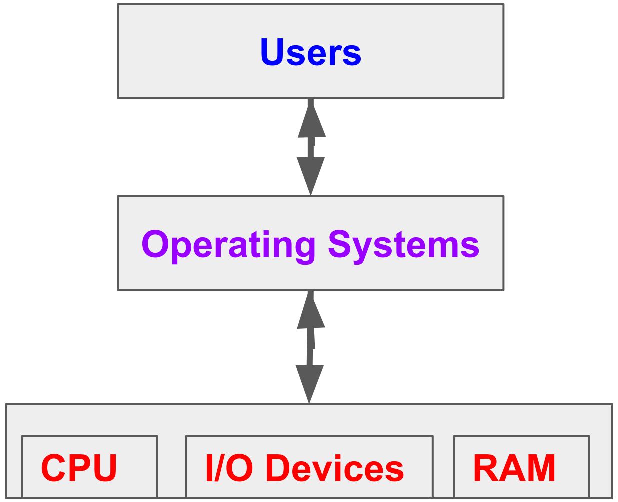
- **Definitions:**
    - Operating Systems is a System Software
    - It Works between the users and computer Hardwares like CPU,I/O Devices, and Memory
    - Operating system is a **Interface** between users and computer Hardwares



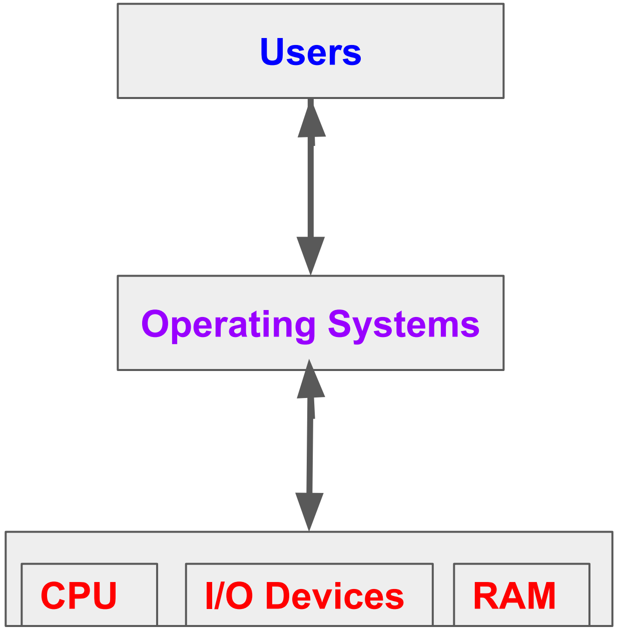
Introduction to Operating Systems

- **Definitions:**
    - Operating Systems is a System Software
    - It Works between the users and computer Hardwares like CPU,I/O Devices, and Memory
    - Operating system is a **Interface** between users and computer Hardwares



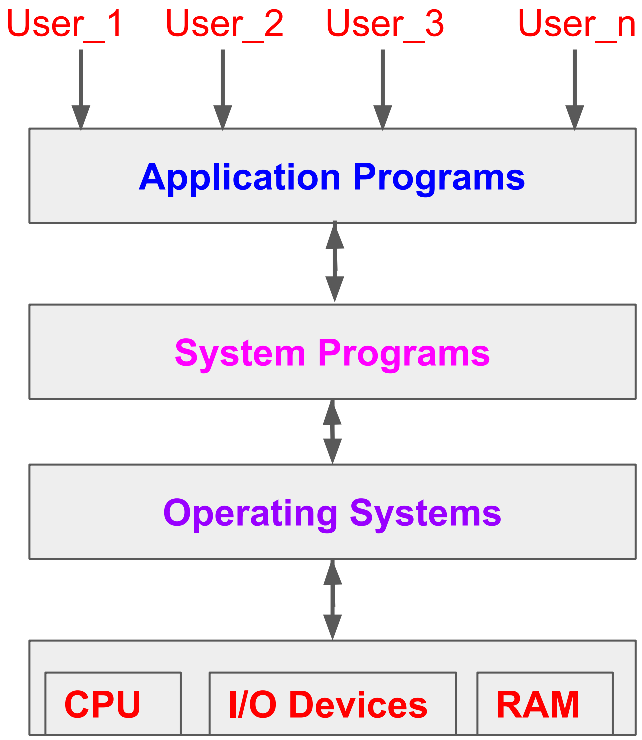


Introduction to Operating Systems

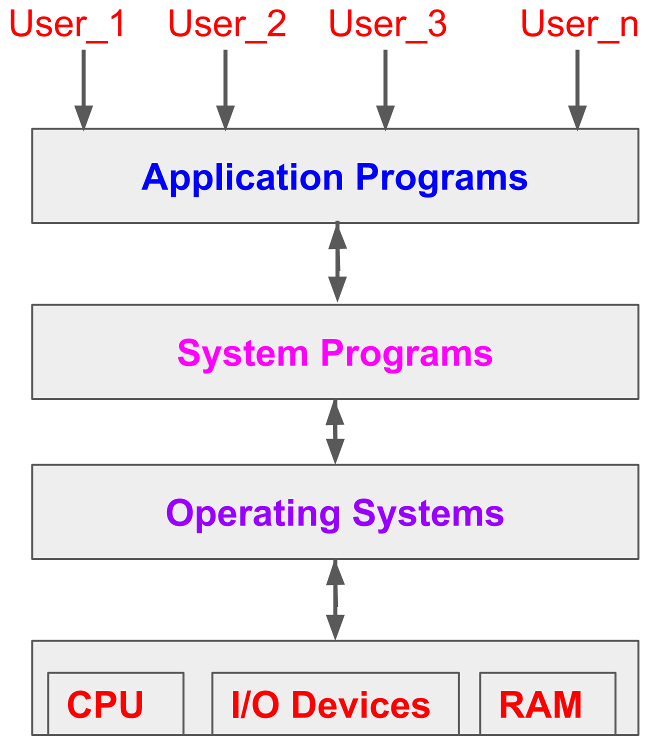


- Functionality of Operating Systems:
  - Resource Management
  - Process Management (CPU Scheduling)
  - Storage Management(HDD)
  - Memory Management(RAM)
  - Security

System Programs



System Programs



- Programs are of two types
  - Application Programs
  - System Programs
- System Programs provide a convenient environment for program developments and execution.
- Application programs are basically designed for specific task.
- System programs are basically operates on computer hardware OR Provide a platform to application programs to run

System Programs

System Programs can be divided into following categories

- File Management
- Status Information
- File Modification
- Programming Language Support
- Program Loading and Execution
- Communication

System Programs

- File Management
    - Create
    - Delete
    - Copy
    - Rename
    - Print
    - Dump

System Programs

- Status Information
  - Ask systems for
    - Date, Time
    - Amount of Available memory or Disk Space
    - Number of Users
    - Detailed Performance
    - Logging and Debugging Information etc

System Programs

- **File Modifications**
  - Several Text Editors may be available to create and modify the content of files stored on disk or other storage devices
  - There may be a special command to search content of the files.

System Programs

- Programming Language Support
    - Compiler
    - Assembler
    - Debugger
    - Interpreter

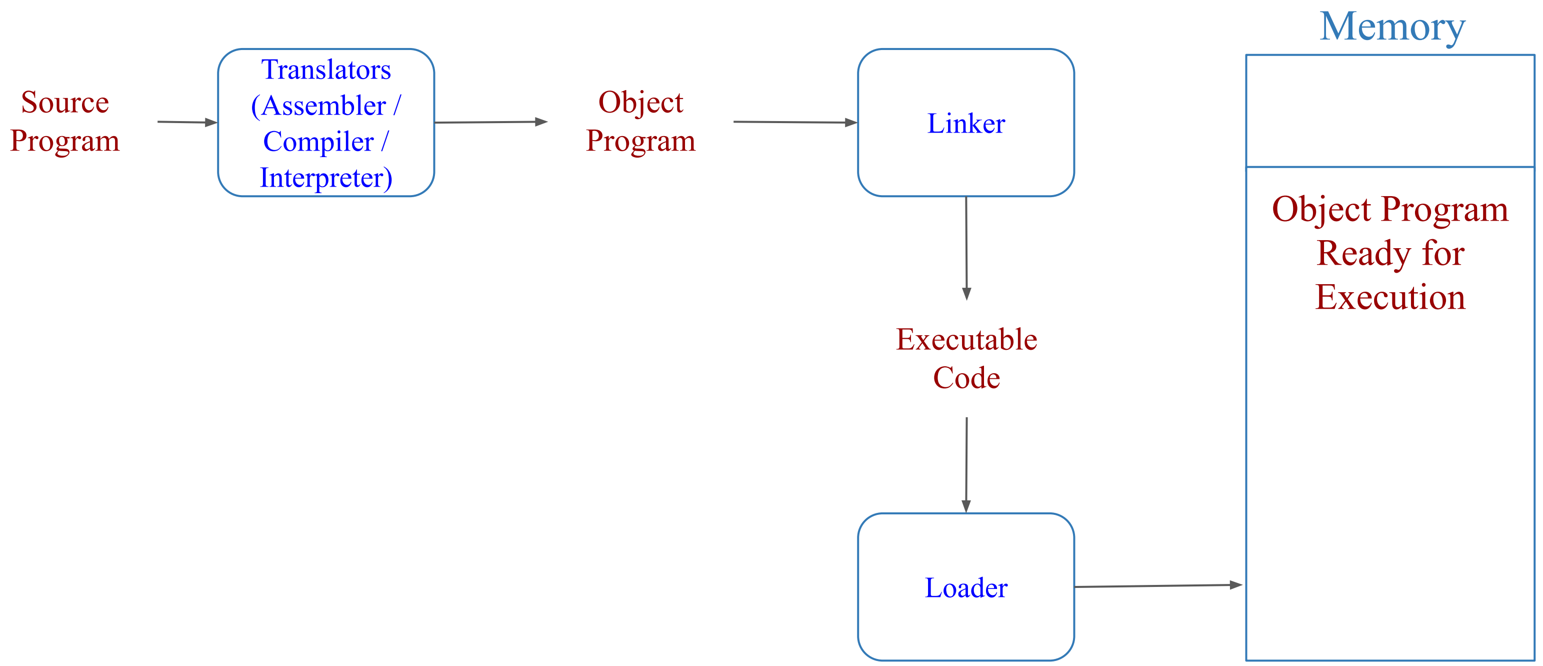
System Programs

- Program Loading and Execution
    - Once the program is assembled or compiled, it must be loaded into main memory to be executed
    - The system may provide
        - Absolute Loader
        - Relocate Loader
        - Linking Editors, and
        - Overlay Loader

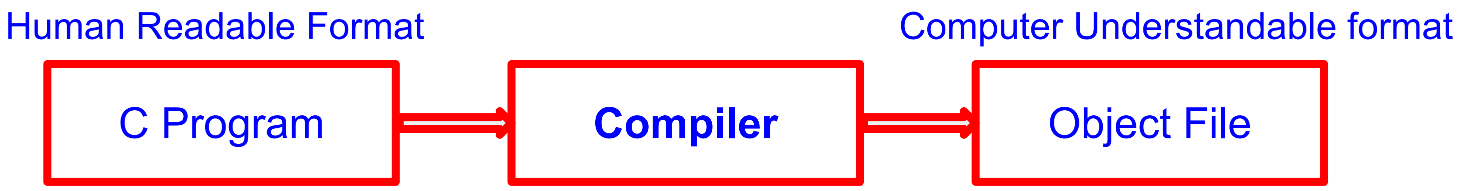
System Programs

- Communication
  - These programs provide mechanism for:
    - Creating virtual connections among processes, users, and computer systems
    - Allowing users to send messages to one another's screens.
    - To browse web pages
    - To send electronic-mail messages
    - To log in remotely or to transfer files from one machine to another machine.

Program Execution Process


Source Program
Translators (Assembler / Compiler / Interpreter)
Object Program
Linker
Executable Code
Loader
Memory
Object Program Ready for Execution

Compiler/Assembler

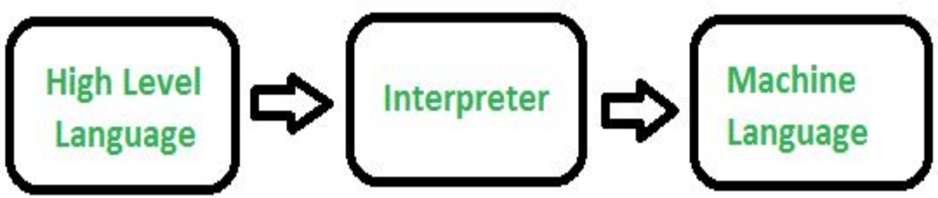


- A *compiler* is a program that translates a source program written in some high-level programming language (such as Java) into machine code.
- **The generated machine code can be later executed many times against different data each time.**
- A compiler is a tool which has the ability to read the source code and translate it to object level code.
- **The output of the compiled code is referred to as the object code or sometimes called the object module. (It is to be noted that this object file/object module is not related to OOP).**

Why do we need compiler to execute the program?

- Because computer can't understand the source code directly. It will understand only object level code.
- Source codes are human readable format but the system cannot understand it.
- So, the compiler is intermediate between human readable format and machine-readable format.

Interpreter



- Advantage : It is executed line by line which helps users to find errors easily.
- Disadvantage : It takes more time to execute successfully than compiler.

- Converts high level language to machine level language
- Read Line-by-line
- If an error is found on any line, the execution stops till it is corrected.
- e.g. Phyton, Ruby, Perl, PHP and Matlab.

Interpreter

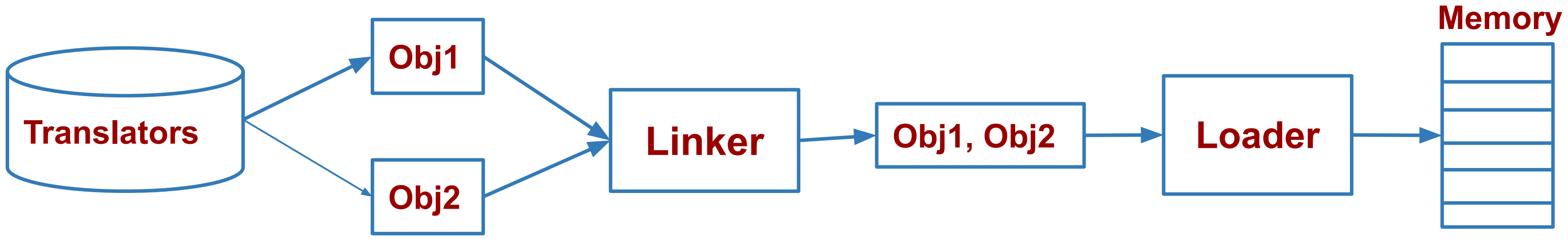
- All high level languages need to be converted to machine code so that the computer can understand the program after taking the required inputs.
- The software by which the conversion of the high level instructions is performed line-by-line to machine level language, other than compiler and assembler, is known as INTERPRETER.
- If an error is found on any line, the execution stops till it is corrected. This process of correcting errors is easier as it gives line-by-line error but the program takes more time to execute successfully.

Interpreter

Top Interpreters according to the computer languages –

- Phyton- CPhyton, PyPy, Stackless Phyton, IronPhyton
- Ruby- YARV, Ruby MRI (CRuby)
- JAVA- HotSpot, OpenJ9, JRockIt
- Kotlin- JariKo

Linker

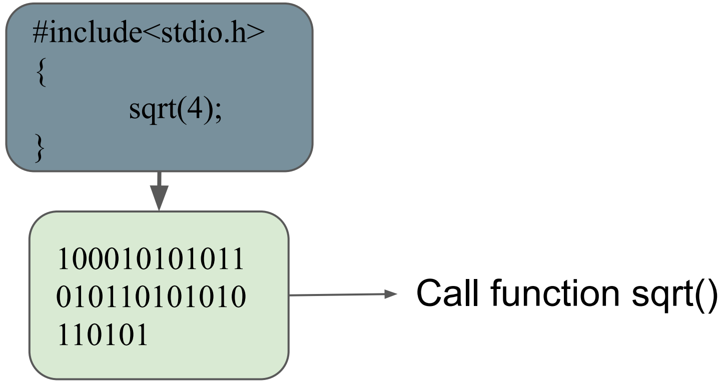


- Static Linking
- Dynamic Linking

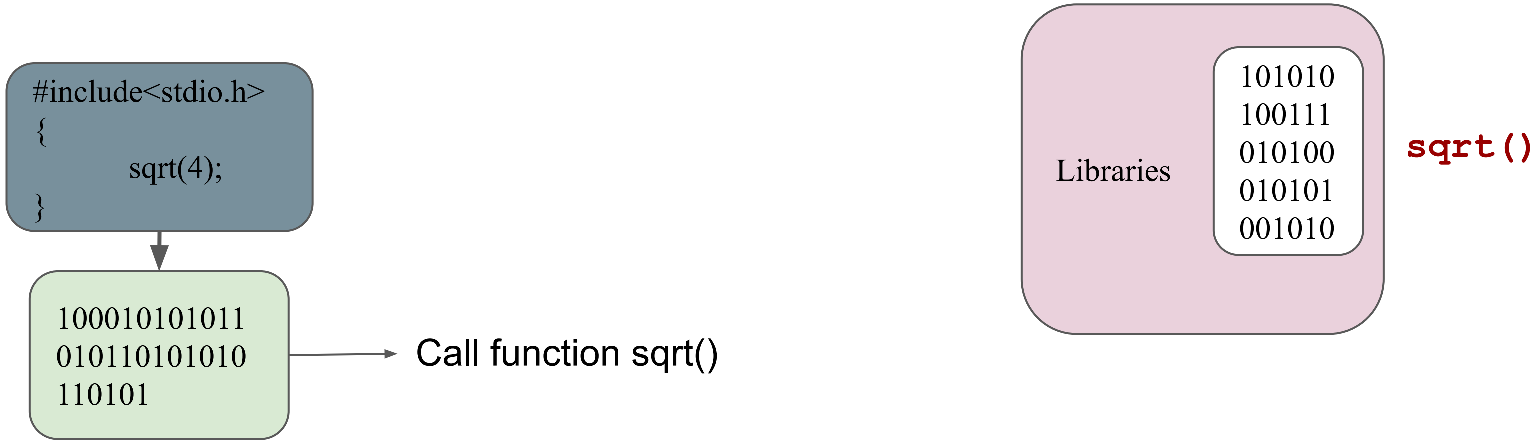
Static Linker

```
#include<stdio.h>
{
        sqrt(4);
}
```

Static Linker



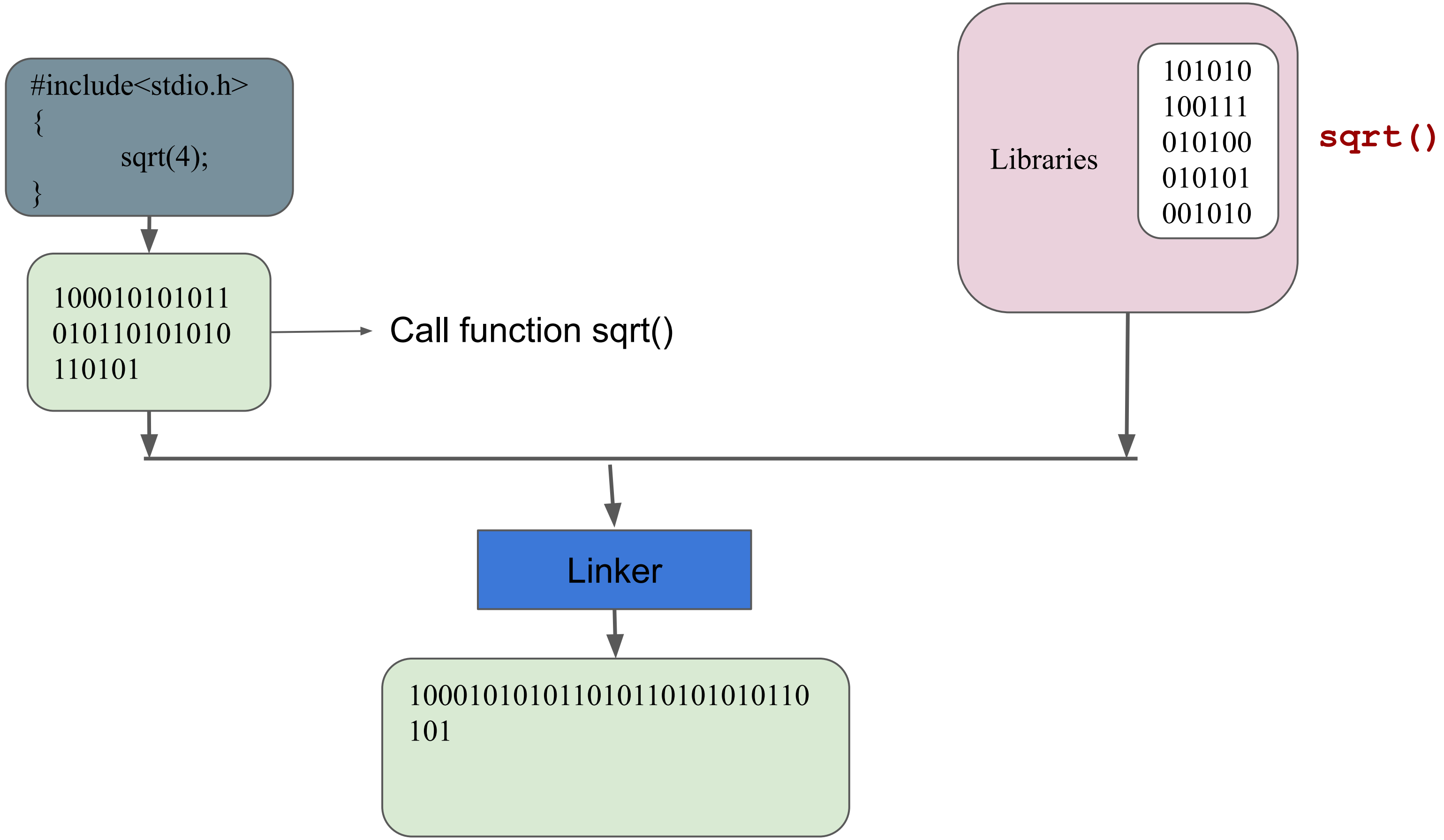
Static Linker



```
#include<stdio.h>
{
        sqrt(4);
}
```

```
100010101011
010110101010
110101
```

Static Linker



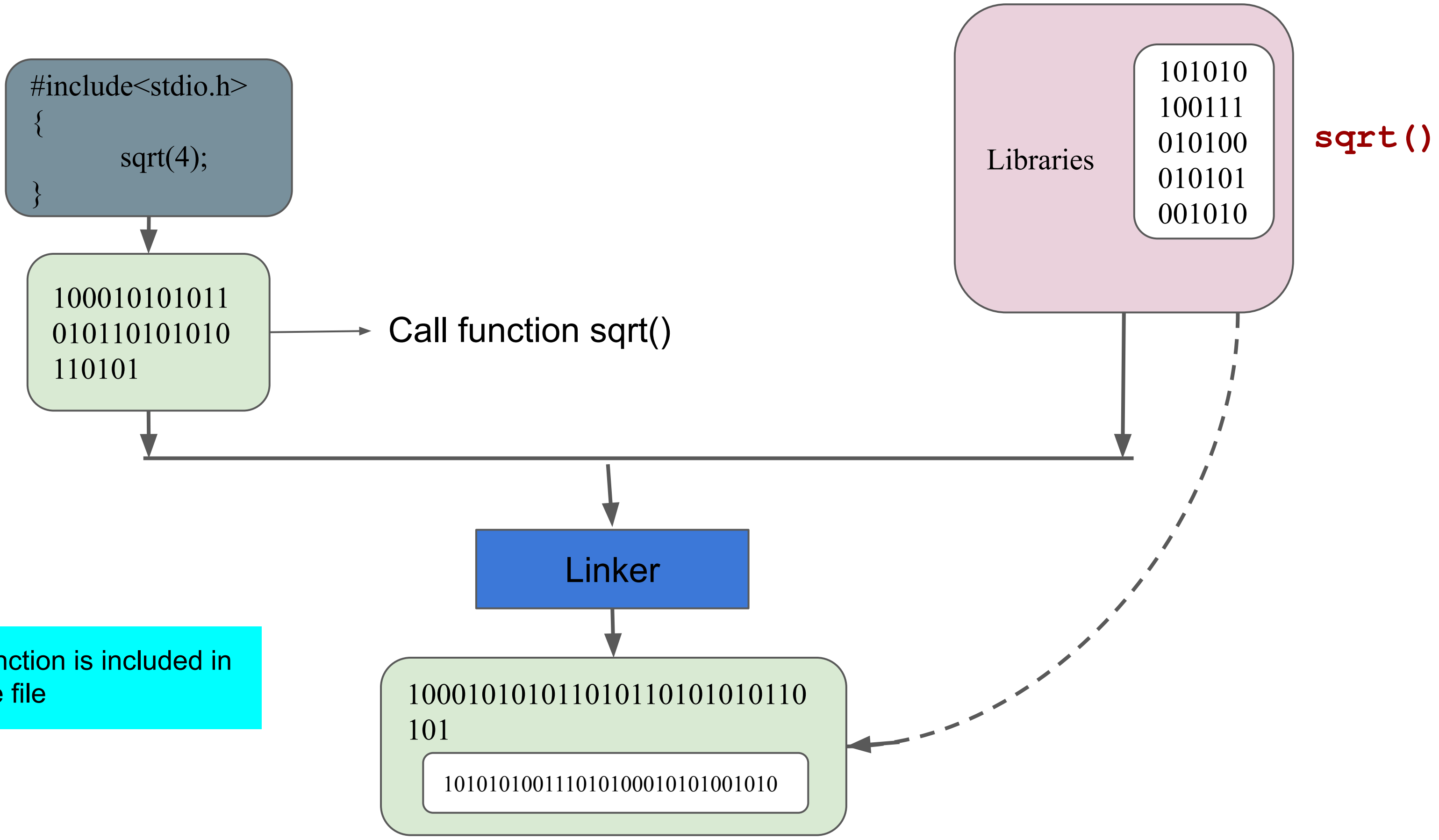
```
#include<stdio.h>
{
        sqrt(4);
}
```

```
100010101011
010110101010
110101
```

```
101010
100111
010100
010101
001010
```

```
100010101011010110101010110
101
```

Static Linker



Code of sqrt function is included in final executable file

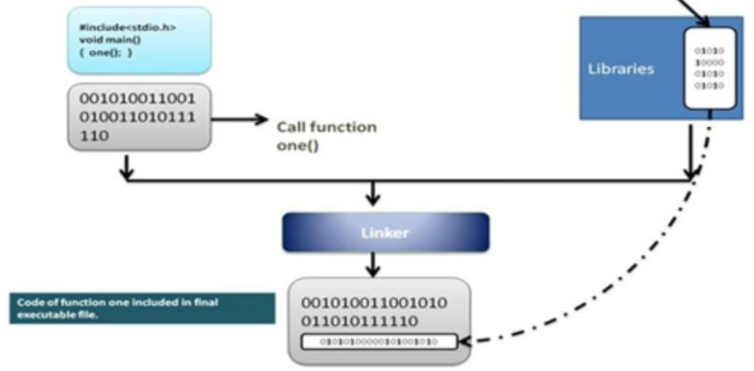
Static Linker

1

exe file generated

4

Again compilation needed so that exe file include updated sqrt() function



```
#include<stdio.h>
void main()
{ one(); }
```

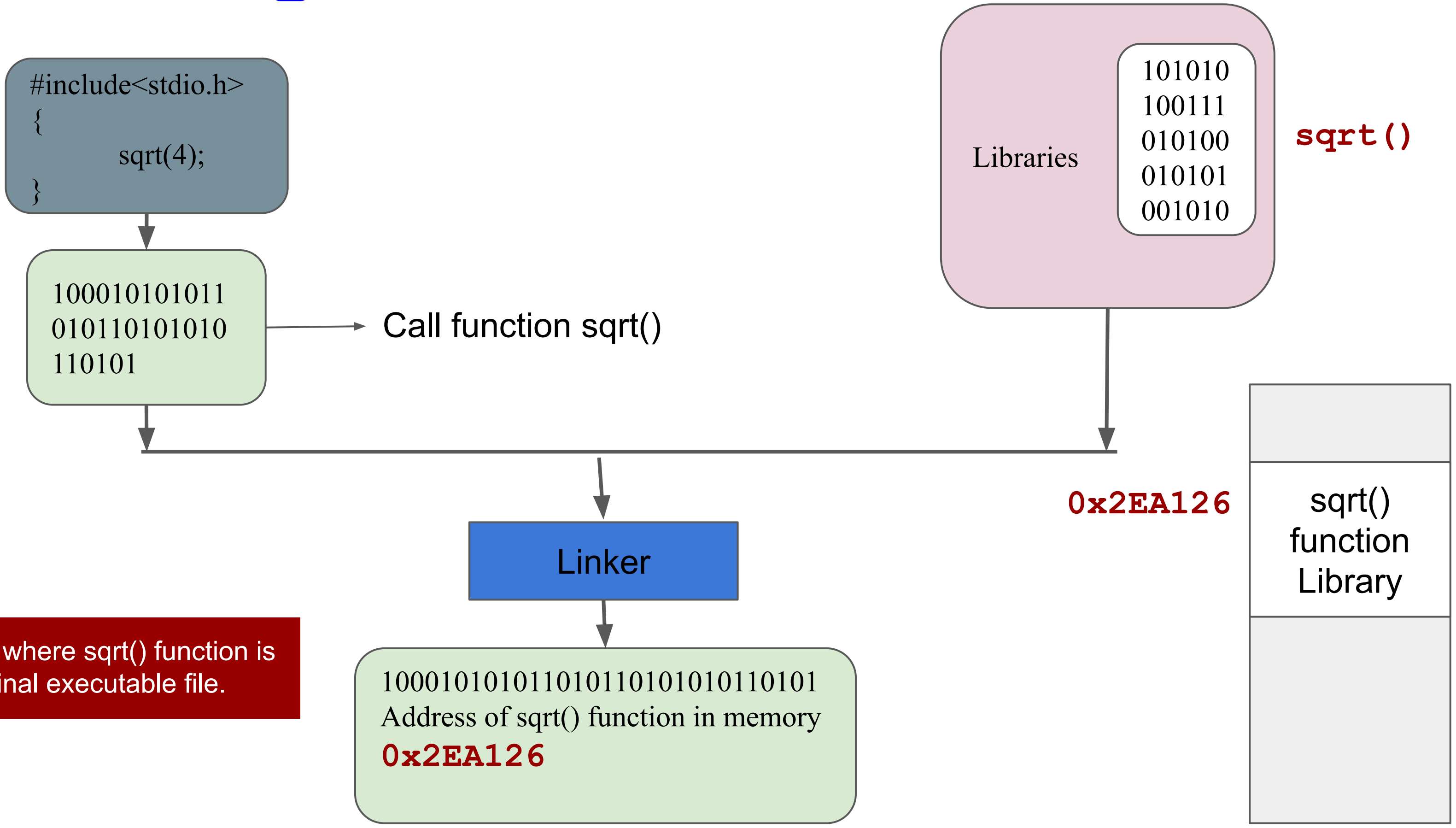
2

User take exe file.

3

Some changes made in one function.

Dynamic Linker

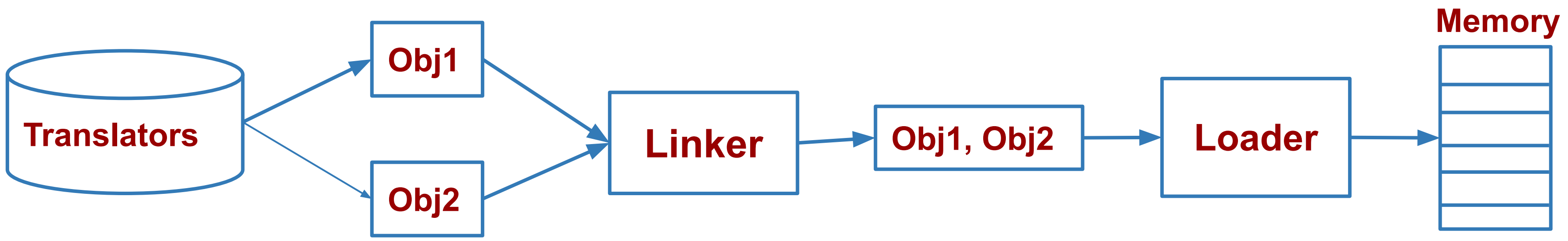


Memory Addresses, where sqrt() function is loaded, included in final executable file.

Dynamic Linker

- Static Linker
  - Windows - .lib (Library)
  - Linux - .a (Archive)
- Dynamic Linker
  - Windows - .dll (Dynamic Link Library)
  - Linux - .so (Shared Objects)

Loader



Loader

In computer systems a loader is the part of an operating system that is responsible for loading programs and libraries. It is responsible for initiating the execution process

- Process / Functions of Loader:
  - Allocation: The space for program is allocated in the main memory by calculating the size of the program
  - Loading : Brings the object program in to the memory for execution
  - Relocation : When program is loaded from secondary memory to primary memory, its address gets change,this address will be handled by the loader.
  - Linking : Which combines two or more separate object programs and supplies the necessary information.

Linker Vs Loader

Linker	Loader
Main function of linker is to generate executable file	To load executable file into main memory
The linker takes input of object code generated by compiler/ assembler	And the loader takes input of executable files generated by linker
Linking can be defined as process of combining various pieces of codes and source code to obtain executable code.	Loading can be defined as process of loading executable codes to main memory for further execution.
Another use of linker is to combine all object modules.	It helps in allocating the address to executable codes/files.