CSE-105:
PROGRAMMING FUNDAMENTALS
LECTURE 2: BASICS OF C

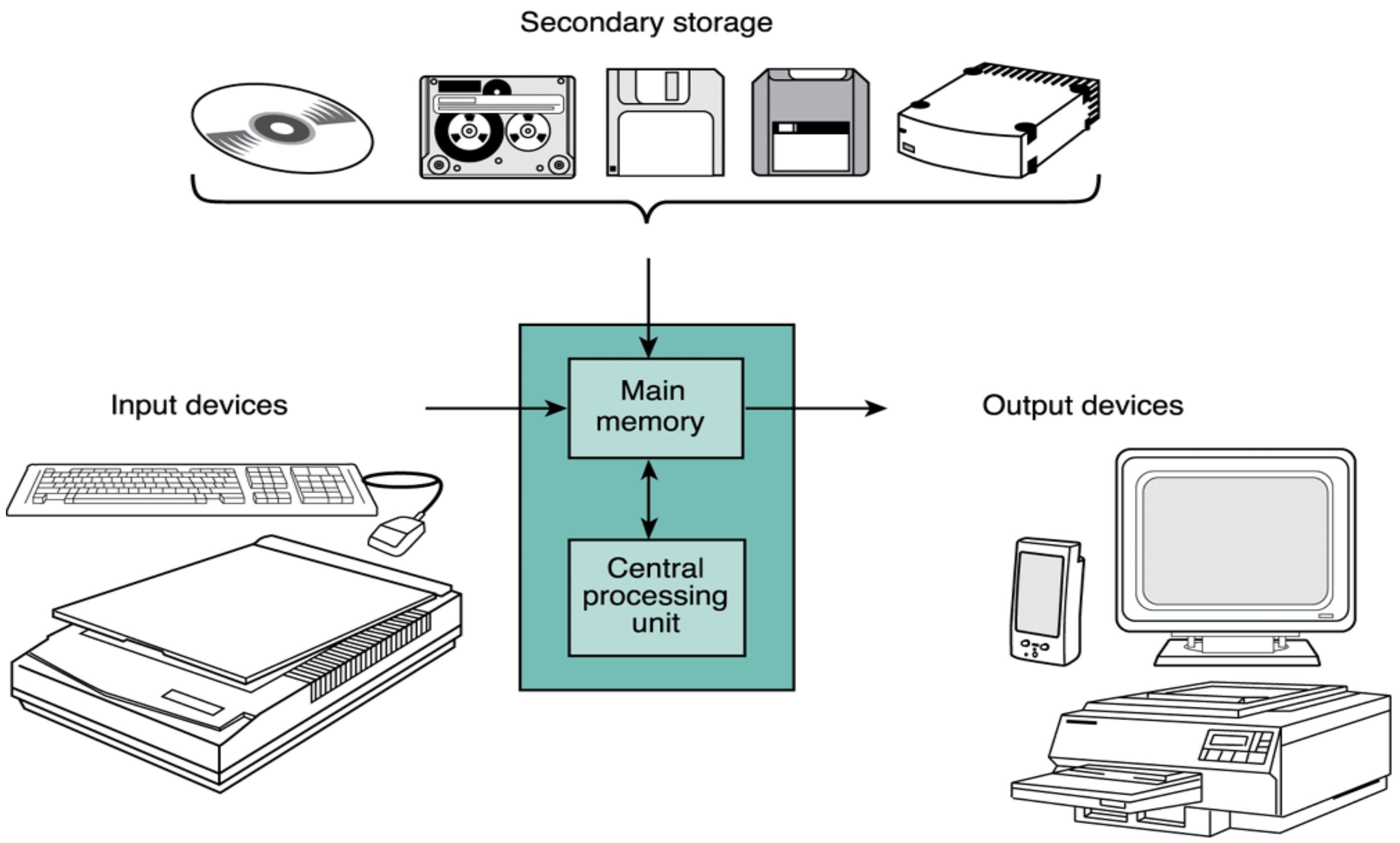
COURSE INSTRUCTOR: MD. SHAMSUJJOHA

What is Computer?

- Computer
  - Device capable of performing computations and making logical decisions
  - Computers process data under the control of sets of instructions called computer programs
- Hardware
  - Various devices comprising a computer
  - Keyboard, screen, mouse, disks, memory, CD-ROM, and processing units
- Software
  - Programs that run on a computer

Computer Organization

- Six logical units of computer



Components of a PC

Computer Organization

- Six logical units of computer
  1. Input unit
     - Accepts information from the user and transforms it to ***digital codes*** that the computer can process → Receiving section:
     - Obtains information from input devices such as Keyboard, mouse, microphone, Scanner …
  2. Output unit
     - An ***interface*** by which the computer conveys the output to the user → "Shipping" section
     - Takes information processed by computer, Places information on output devices Screen, printer, networks, …
       - Information used to control other devices

Computer Organization

- Six logical units of computer
  3. Memory unit
     - A semiconductor device which stores the information necessary for a program to run.
     - 2 types
       - ROM (Read Only Memory)
         - Contains information that is necessary for the computer to boot up
         - The information stays there permanently even when the computer is turned off.
       - RAM (Random Access Memory)
         - Contains instruction or data needed for a program to run
         - Got erased when the computer is turned off.

Computer Organization

- Six logical units of computer

  Central processing unit (CPU)

  - Does most of the work in executing a program
  - The CPU inside a PC is usually the microprocessor consists of 3 main parts:

  A. Control Unit
     - Fetch instructions from main memory and put them in the instruction register (Also called Forth logic unit of a Computer)

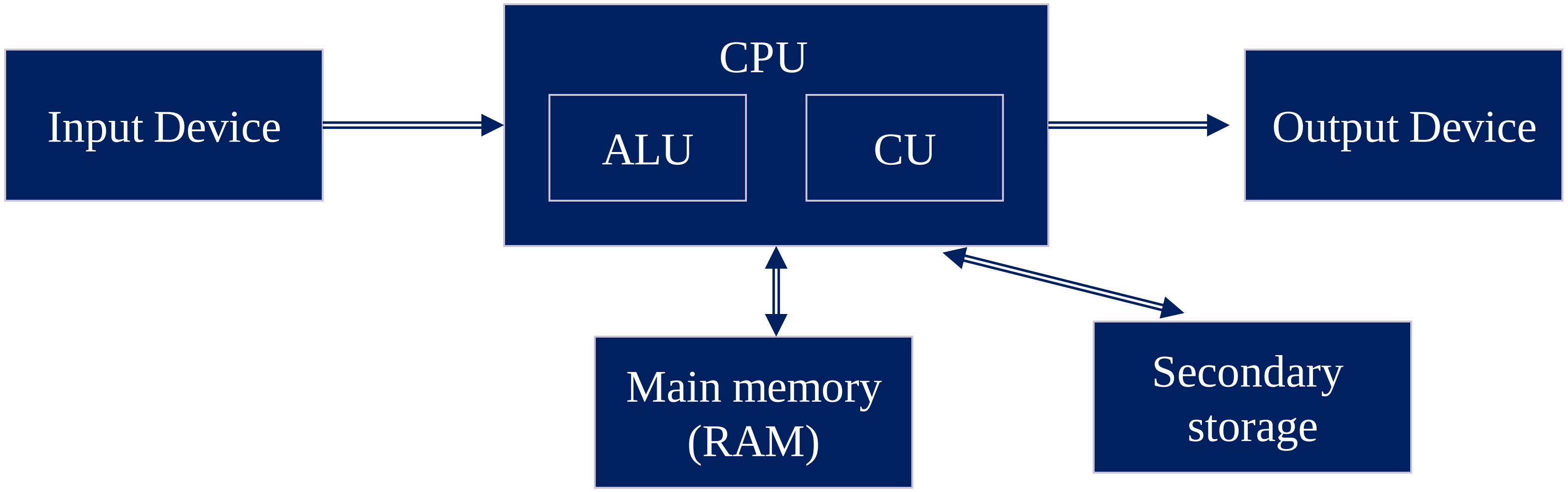
  B. ALU (Arithmetic Logic Unit)
     - Execute arithmetic operations (Also called Fifth logic unit of a Computer)

Computer Organization

- Six logical units of computer

6. Secondary storage unit
    - Long-term, high-capacity “warehouse” section
    - Storage
        - Inactive programs or data
    - Secondary storage devices
        - Disks
    - Longer to access than primary memory
    - Less expensive per unit than primary memory

The von Neumann architecture



How it works

- How does a computer execute a program ? (example programs: a computer game, a word processor, etc)
  - The instructions that comprise the program are copied from the permanent secondary memory into the main memory
  - After the instructions are loaded, the CPU starts executing the program.
  - For each instruction, the instruction is retrieved from memory, decoded to figure out what it represents, and the appropriate action carried out. (the *fetch execute cycle*)
  - Then the next instruction is fetched, decoded and executed.

Machine Languages, Assembly Languages, and High-level Languages

- Three types of computer languages
  1. Machine language
     - Only language computer directly understands
     - "Natural language" of computer
     - Defined by hardware design
       - Machine-dependent
     - Generally consist of strings of numbers
       - Ultimately 0s and 1s
     - Instruct computers to perform elementary operations
       - One at a time
     - Cumbersome for humans
     - Example:

```
+1300042774
+1400593419
+1200274027
```

Machine Languages, Assembly Languages, and High-level Languages

- Three types of computer languages
  2. Assembly language
     - English-like abbreviations representing elementary computer operations
     - Clearer to humans
     - Incomprehensible to computers
       - Translator programs (assemblers)
         - Convert to machine language
     - Example:

```
LOAD   BASEPAY
ADD    OVERPAY
STORE  GROSSPAY
```

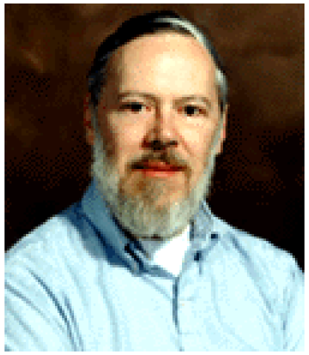
Machine Languages, Assembly Languages, and High-level Languages

- Three types of computer languages
  3. High-level languages
     - Similar to everyday English, use common mathematical notations
     - Single statements accomplish substantial tasks
       - Assembly language requires many instructions to accomplish simple tasks
     - Translator programs (compilers)
       - Convert to machine language
     - Interpreter programs
       - Directly execute high-level language programs
     - Example:

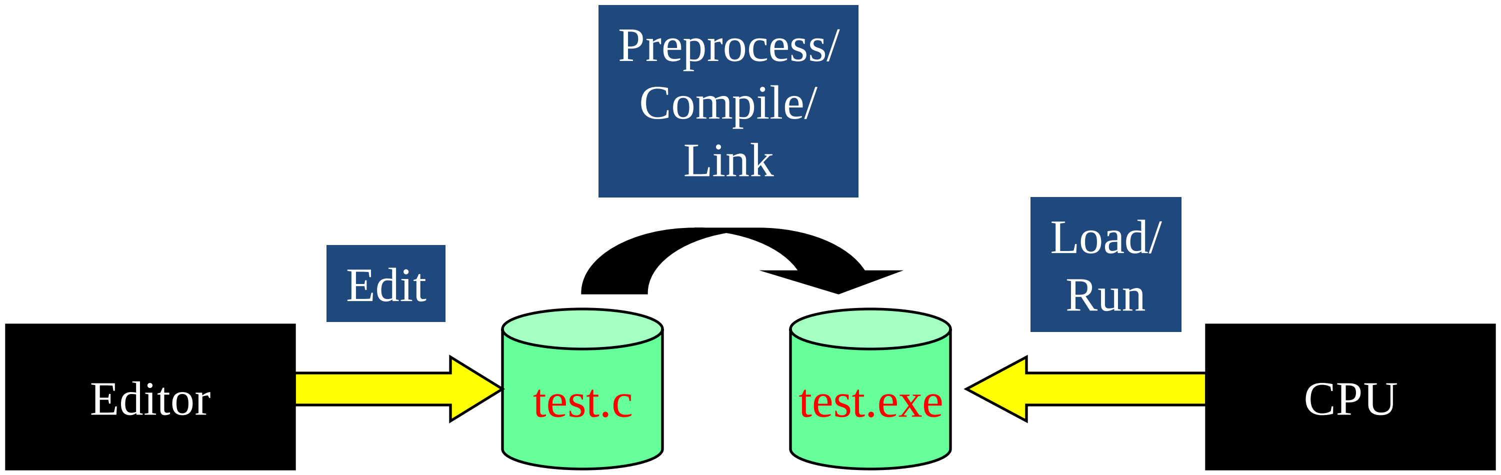
```
grossPay = basePay + overTimePay
```

C History

- Developed between 1969 and 1973 along with Unix
- Due mostly to Dennis Ritchie
- Designed for systems programming
  - Operating systems
  - Utility programs
  - Compilers
  - Filters



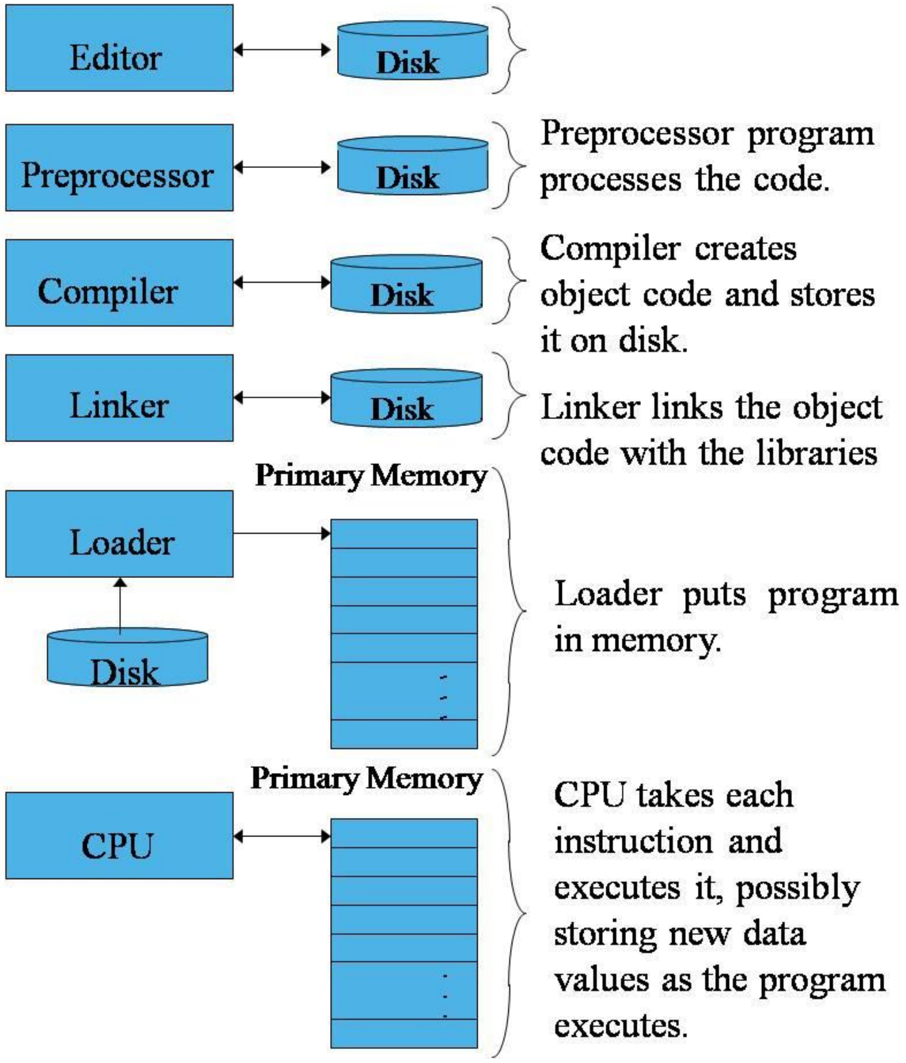
C Programming Environment



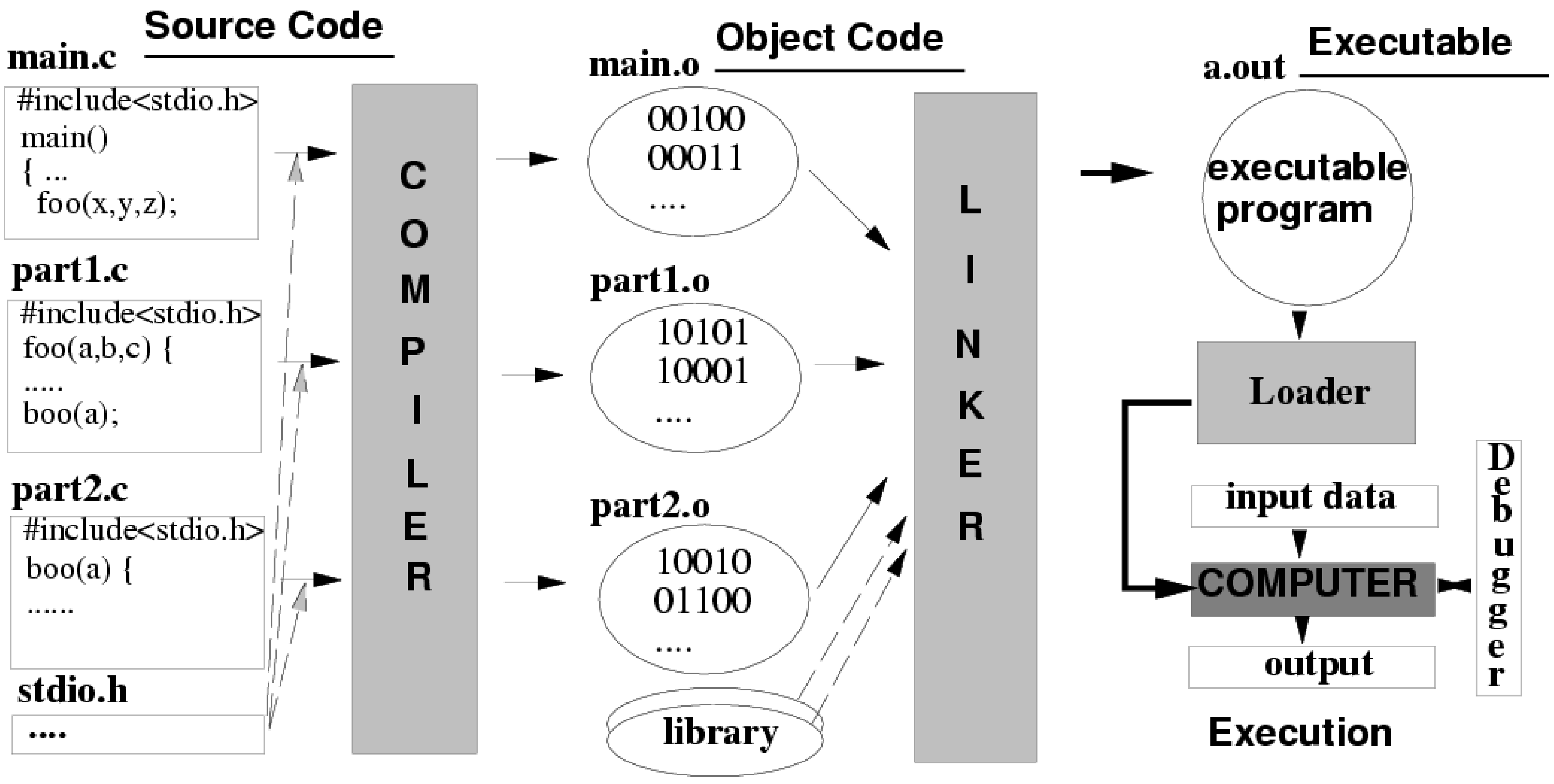
C Programming Environment

- Phases of C Programs:

1. *Edit*
2. *Preprocess*
3. *Compile*
4. *Link*
5. *Load*
6. *Execute*


Editor
Disk
Preprocessor
Disk
Preprocessor program processes the code.
Compiler
Disk
Compiler creates object code and stores it on disk.
Linker
Disk
Linker links the object code with the libraries
Primary Memory
Loader
Disk
Loader puts program in memory.
Primary Memory
CPU
CPU takes each instruction and executes it, possibly storing new data values as the program executes.

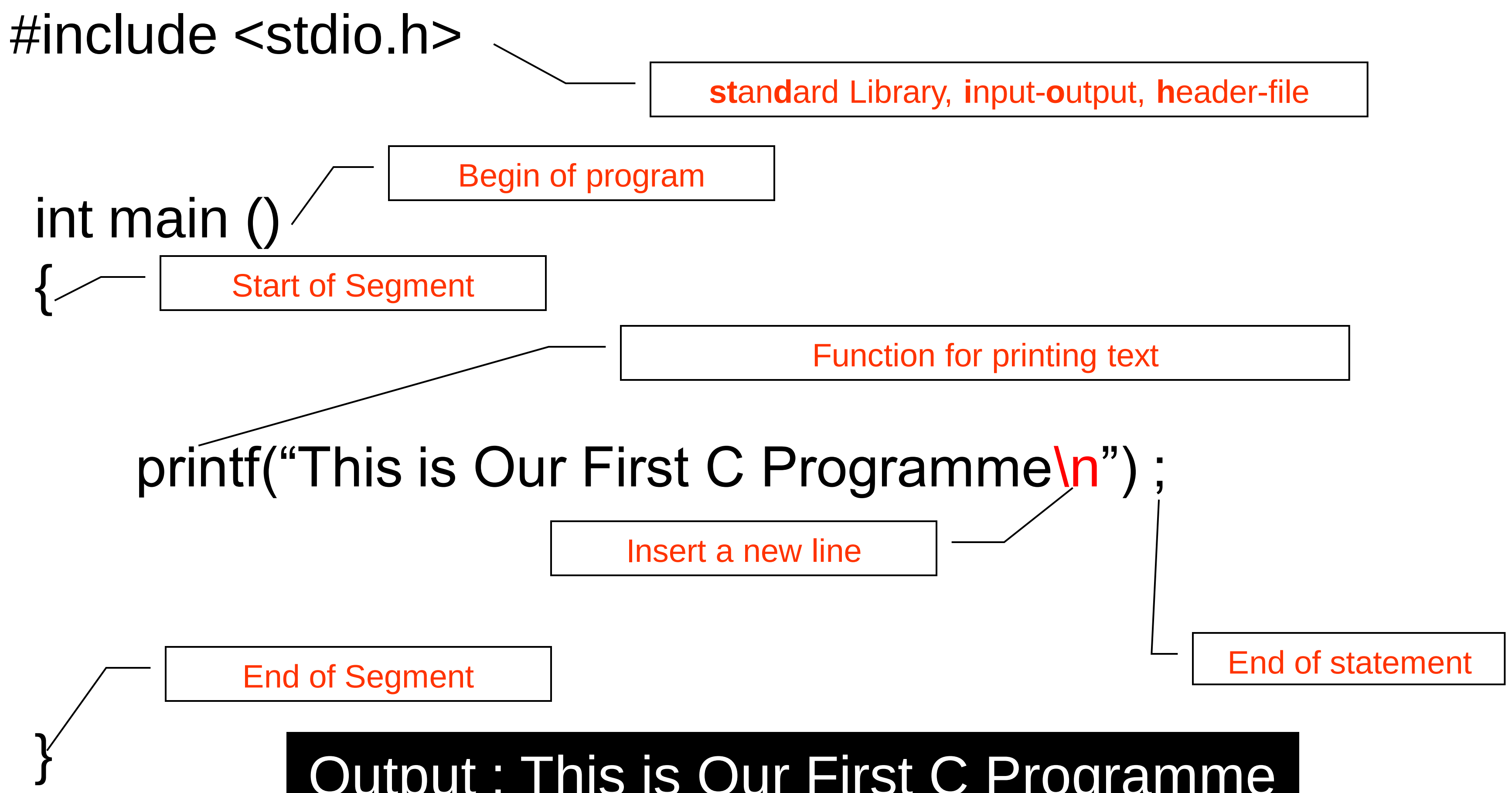
C Compilers, Linkers, Loaders



Some Key Terms

- Source Program
  - printable/Readable Program file
- Object Program
  - nonprintable machine readable file
- Executable Program
  - nonprintable executable code
- Syntax errors
  - reported by the compiler
- Linker errors
  - reported by the linker
- Execution/Run-time errors
  - reported by the operating system

A Simple Program in C - exp



```
#include <stdio.h>

int main ()
{

    printf("This is Our First C Programme\n") ;

}
```

stan**d**ard Library, **i**nput-**o**utput, **h**eader-file
Begin of program
Start of Segment
Function for printing text
Insert a new line
End of statement
End of Segment

Output : This is Our First C Programme

A Simple Program in C - exp

```
#include <stdio.h>

 int main ()
 {

       printf(“This is Our First C\n Programme”) ;


 }
```

```
Output : This is Our First C
          Programme
```

A Simple Program in C - exp

```
#include <stdio.h>

 int main ()
 {

      printf(“This \n is Our First C\n Programme”) ;


 }
```

```
Output : This
              is Our First C
              Programme
```

A Simple Program in C - exp

```
#include <stdio.h>

 int main ()
 {

        printf("This \n is Our First C\n Programme") ;
        printf("PROGRAMME") ;


 }
```

```
Output : This
                 is Our First C
                 ProgrammePROGRAMME
```

A Simple Program in C - exp

```
#include <stdio.h>

 int main ()
 {

         printf(“This \n is Our First C\n Programme ”) ;
         printf(“PROGRAMME”) ;

 }
```

```
Output : This
                is Our First C
                Programme PROGRAMME
```

A Simple Program in C - exp

```
#include <stdio.h>

 int main ()
 {

        printf(“This \n is Our First C\n Programme ”) ;
        printf(“ PROGRAMME”) ;

 }
```

```
Output : This
          is Our First C
          Programme  PROGRAMME
```

A Simple Program in C - exp

```
#include <stdio.h>

 int main ()
 {

       printf("This \n is Our First C\n Programme ") ;
       printf("\nPROGRAMME") ;


 }
```

```
Output : This
               is Our First C
               Programme
              PROGRAMME
```

A Simple Program in C - exp

```
#include <stdio.h>

 int main ()
 {

      printf(“This \n is Our First C\n Programme\n ”) ;
      printf(“PROGRAMME”) ;


 }
```

```
Output : This
             is Our First C
             Programme
             PROGRAMME
```

A Simple Program in C - exp

```
#include <stdio.h>

 int main ()
 {

     printf("T\nh\ni\ns is Our First C Programme ") ;


 }
```

```
Output : T
         h
         i
         s is Our First C Programme
```

Summary

- We have looked at some underlying hardware
- We have seen some different types of languages;
  - the relevance of **semantics** and **syntax**.
- We have observed the detail necessary in an **imperative language** to **instruct** a computer properly.
- Finally, we examined the syntax to **print** a line of text to the screen of our computer.

Questions or Suggestions

THANK YOU!

Inquiry
dishacse@yahoo.com