C PROGRAMMING FOR ENGINEERS

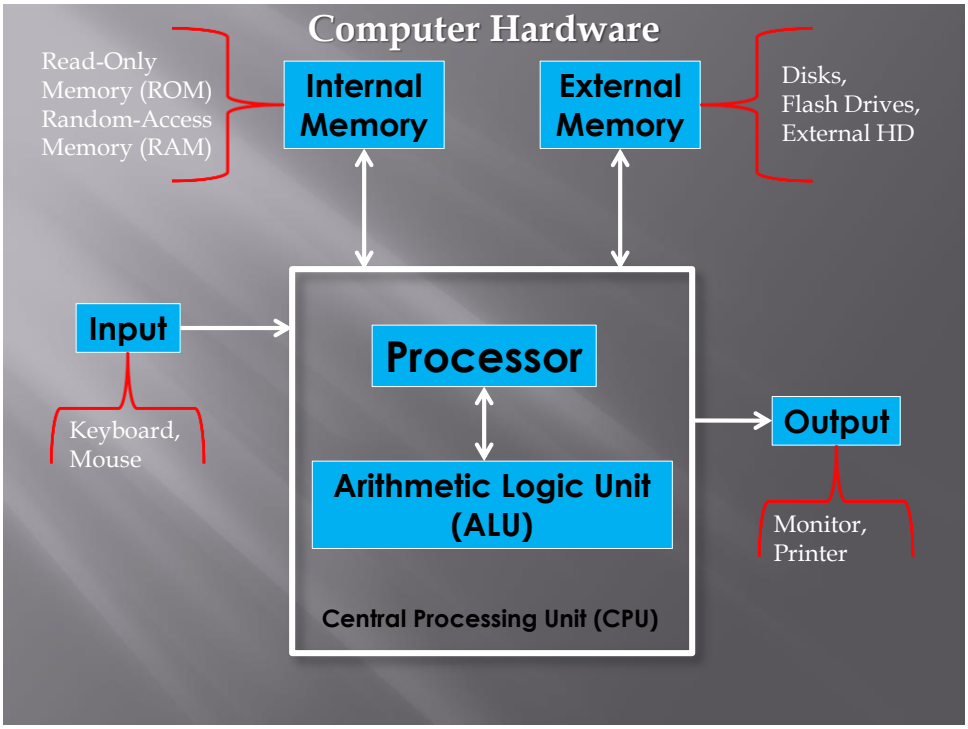
The Basics

History of the C Programming Language

- **C** is a general purpose, machine-independent language very suitable for across platform utilization.
- Was developed at Bell Labs in 1972.
- The **ANSI C** standard was approved in 1989 by the American National Standards Institute(ANSI).

Hardware and Software

- **Computer:** an electronic device that can process data and perform operations that are defined by a set of instructions. The instructions are generally in the form of a written **program.**
- **Hardware:** computer equipment that we interact with either directly or indirectly.
  - keyboard, mouse, terminal, hard disk, printer
- **Software:** programs (commercial or user-built) that define the steps a computer will perform.



Computer Languages

- Low-Level Language (Machine Language)
  - Binary language; written using bits (ones and zeros).
  - Communicate directly with the hardware.
- Assembly Language
  - Written in symbolic statements.
  - Need to know the hardware information.
- High-Level Language
  - User-friendly commands/instructions.
  - Easy to write; large amount of commands.
  - Need to follow syntax (grammar) rules.
  - Types: C, C++, Fortran, Java, Ada, …

Examples of High-Level Languages

FORTRAN 77

```
IF(ALTITUDE .LT. 12500)THEN
   PRINT*, 'Pilot oxygen not required.'
ELSE
   PRINT*, 'Pilot oxygen required.'
ENDIF
```

C

```
if(altitude  < 12500)
   printf("Pilot oxygen not required.");
else
   printf("Pilot oxygen required.");
```

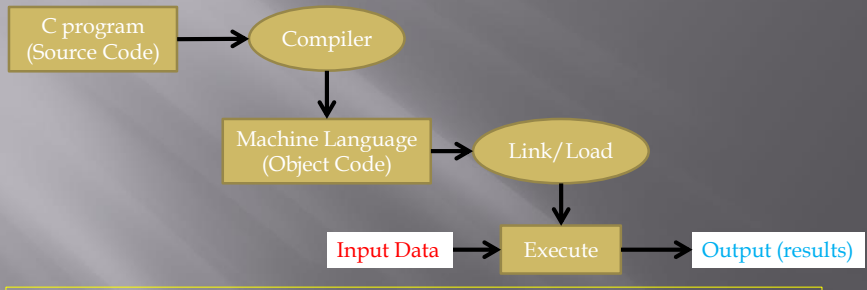
Pascal

```
if altitude < 12500 then
   WriteLn('Pilot oxyen not required.')
else
   WriteLn('Pilot oxygen required.');
```

Ada

```
if altitude < 12500 then
   Ada.Text_IO.Put_Line ("Pilot oxygen not required.");
else
   Ada.Text_IO.Put_Line ("Pilot oxygen required.");
end if;
```

Steps When Executing a Computer Program



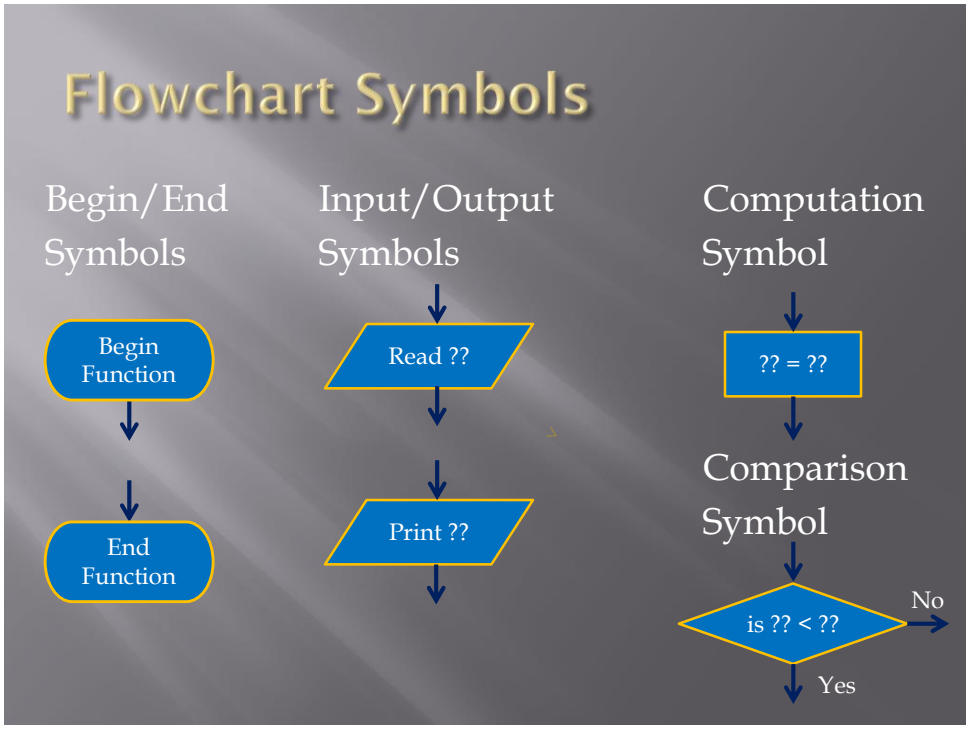
- **Compiler**
  - A piece of software.
  - Translates (converts) a program written in a high-level language (**source code**) into machine language (**object code**).
- **Linker**
  - Another piece of software.
  - Converts the machine language (**object code**) into the **executable** program and links any external files required by the program.

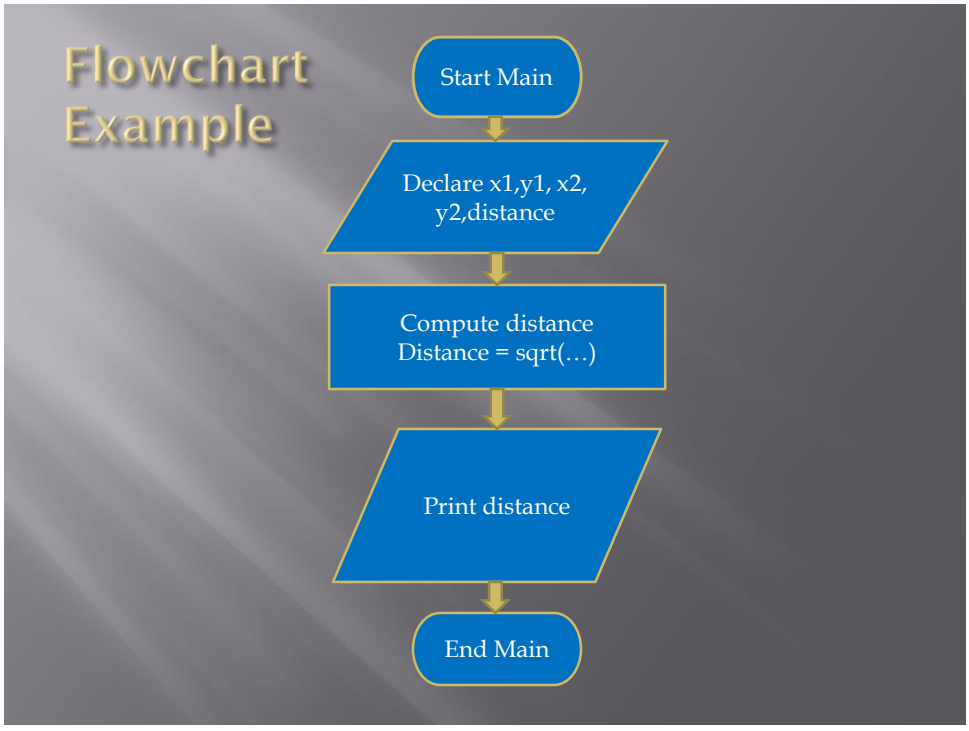
What You Should Know

- **Source Program** (program developer writes in high-level language)
  - Can be printed.
- **Object Program** (created by the compiler)
  - Cannot be printed in a readable format.
- **Executable Program** (created by the linker)
  - Cannot be printed in a readable format.
- **Syntax Errors** (created by the compiler)
  - They are displayed in a readable format and can be printed.
- **Linker Errors** (created by the linker)
  - They are in a readable format and can be printed.
- **Execution Errors or Run-time Errors** (created by the operating system)
  - Displayed in readable format.
  - Sometimes difficult to determine cause and what is affected.
- **Logic Errors** (created by the program developer)
  - Not displayed and often very difficult to determine the cause.

Problem-Solving Methodology

1. State the PROBLEM to be solved.
2. Develop the INPUT/OUTPUT format.
3. Work a HAND CALCULATION to be use as a benchmark for future testing.
4. Write the ALGORITHM (program).
5. Perform vigorous TESTING for all possible conditions.


Flowchart Symbols
Begin/End Symbols
Begin Function
End Function
Input/Output Symbols
Read ??
Print ??
Computation Symbol
?? = ??
Comparison Symbol
is ?? < ??
No
Yes


Flowchart Example
Start Main
Declare x1,y1, x2, y2,distance
Compute distance Distance = sqrt(…)
Print distance
End Main

Example Program

```
/*************************************************************************
   SAMPLE PROGRAM
   This program computes the distance between two points.
   The results are printed to the computer screen.
*************************************************************************/

/* Preprocessor directives */
#include <stdio.h>
#include <math.h>

/* Main function */
int main(void)
{
   /* Declare variables */
   double x1=1, y1=5, x2=4, y2=7, side1, side2, distance;

   /* Compute sides of triangle and distance */
   side1 = x2-x1;
   side2 = y2-y1;
   distance = sqrt(side1*side1 +  side2*side2);

   /* Print distance */
   printf("\nDistance = %5.2f",distance);

   /* Exit program */
   return 0;
}
/*************************************************************************/
```