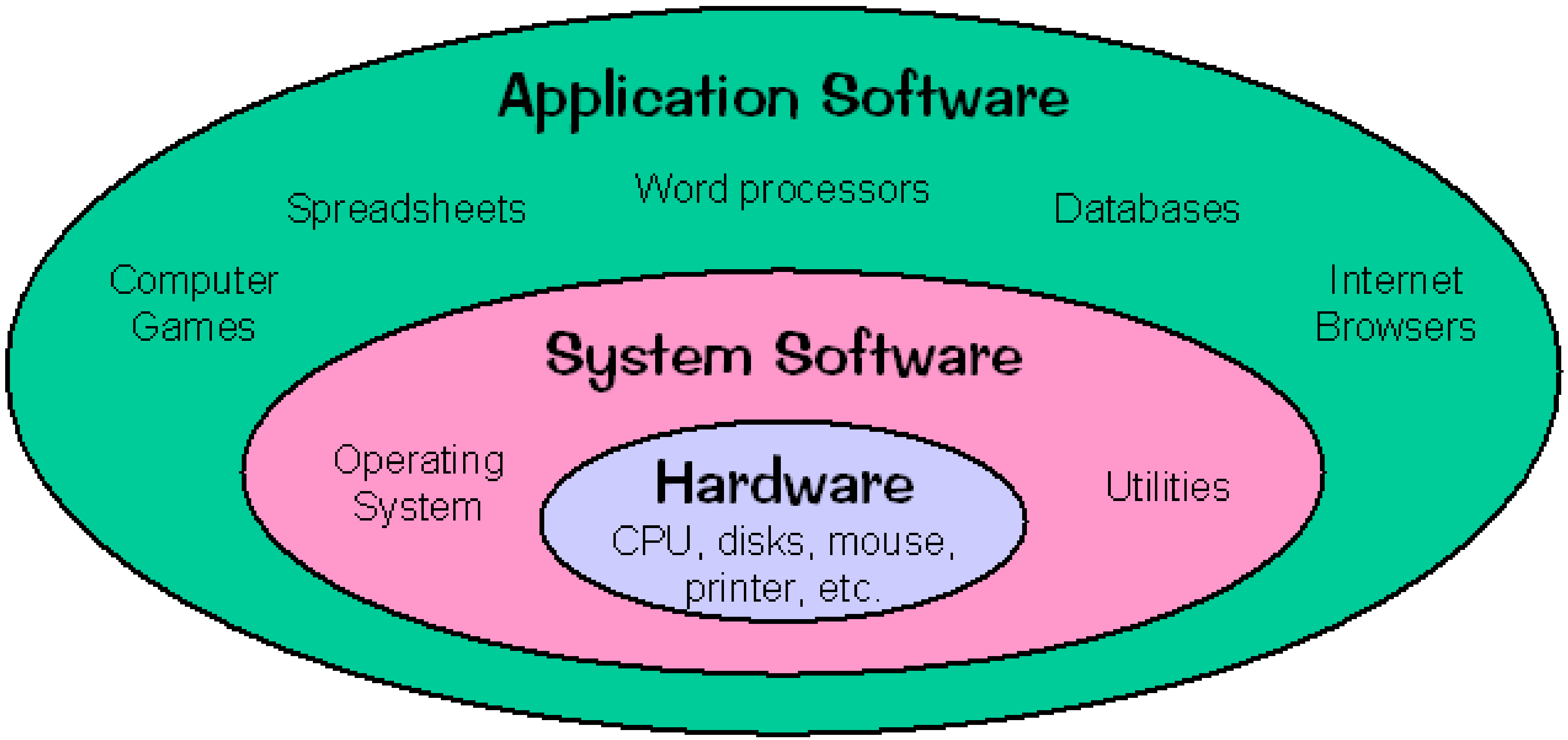
Introduction to Computing

L3-L5

Review

- ✓Problem solving
- ✓Logic and its importance in problem solving
- ✓Computational problems and its classifications
- ✓Computer organization and operating system
- ✓Different computer languages
- ✓Typical C program development environment

Application software, System software & Hardware



Objectives for today

To learn and appreciate the following concepts

✓Introduction to algorithms and flowcharts

✓Algorithms and flowcharts for simple problems

Session outcome

- At the end of session the student will be able to write
  - Algorithms and flowcharts for simple problems
  - Simple C programs

Algorithm

✓A step by step procedure to solve a particular problem

✓Named after Arabic Mathematician Abu Jafar Mohammed Ibn Musa **Al Khowarizmi**

Algorithmic Notations

Name of the algorithm [mandatory]

[gives a meaningful name to the algorithm based on the problem]

Start [Beginning of the algorithm]

Step Number [mandatory]

[indicate each individual simple task]

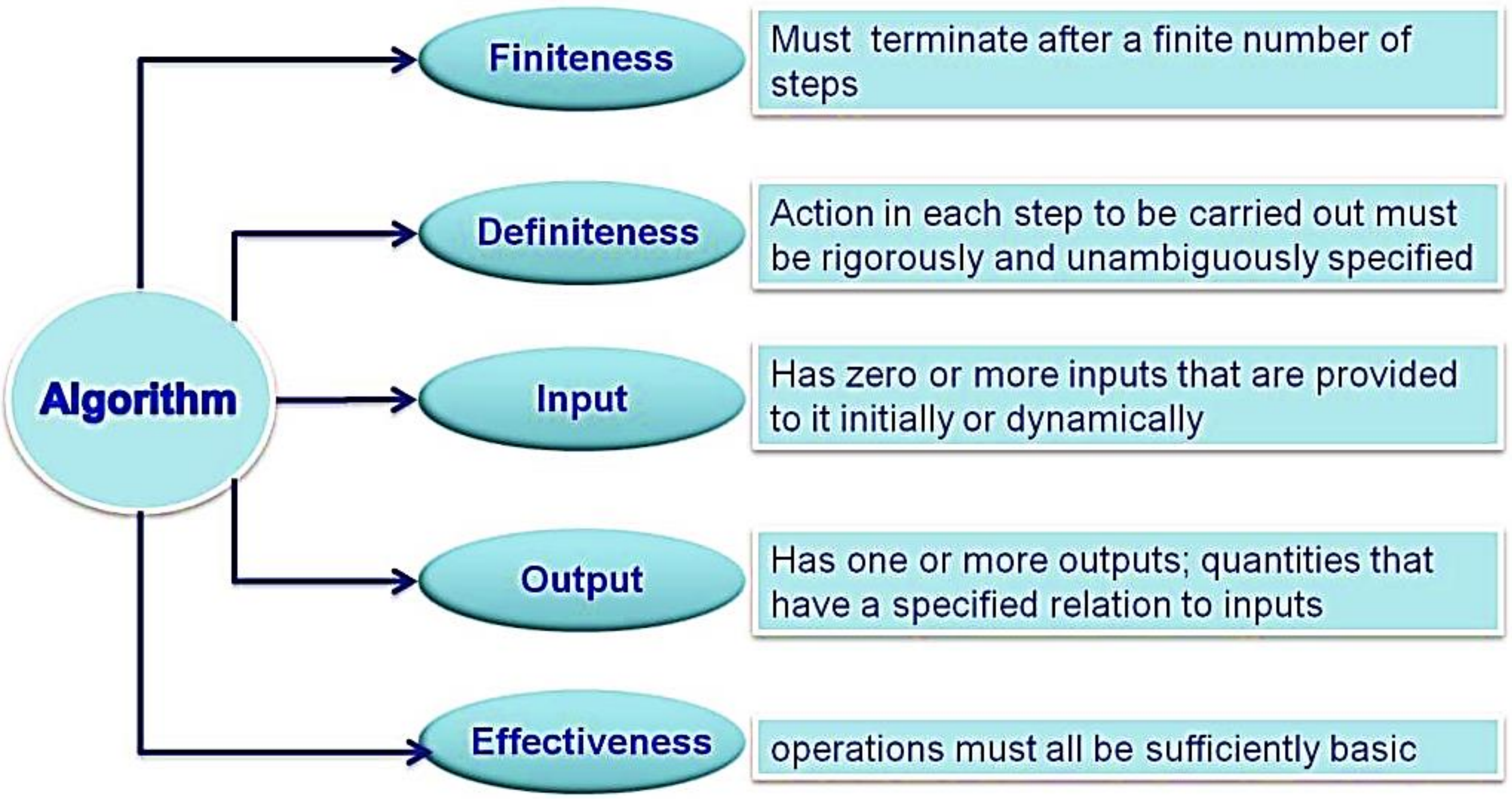
Explanatory comment [optional]

[gives an explanation for each step, if needed]

Termination [mandatory]

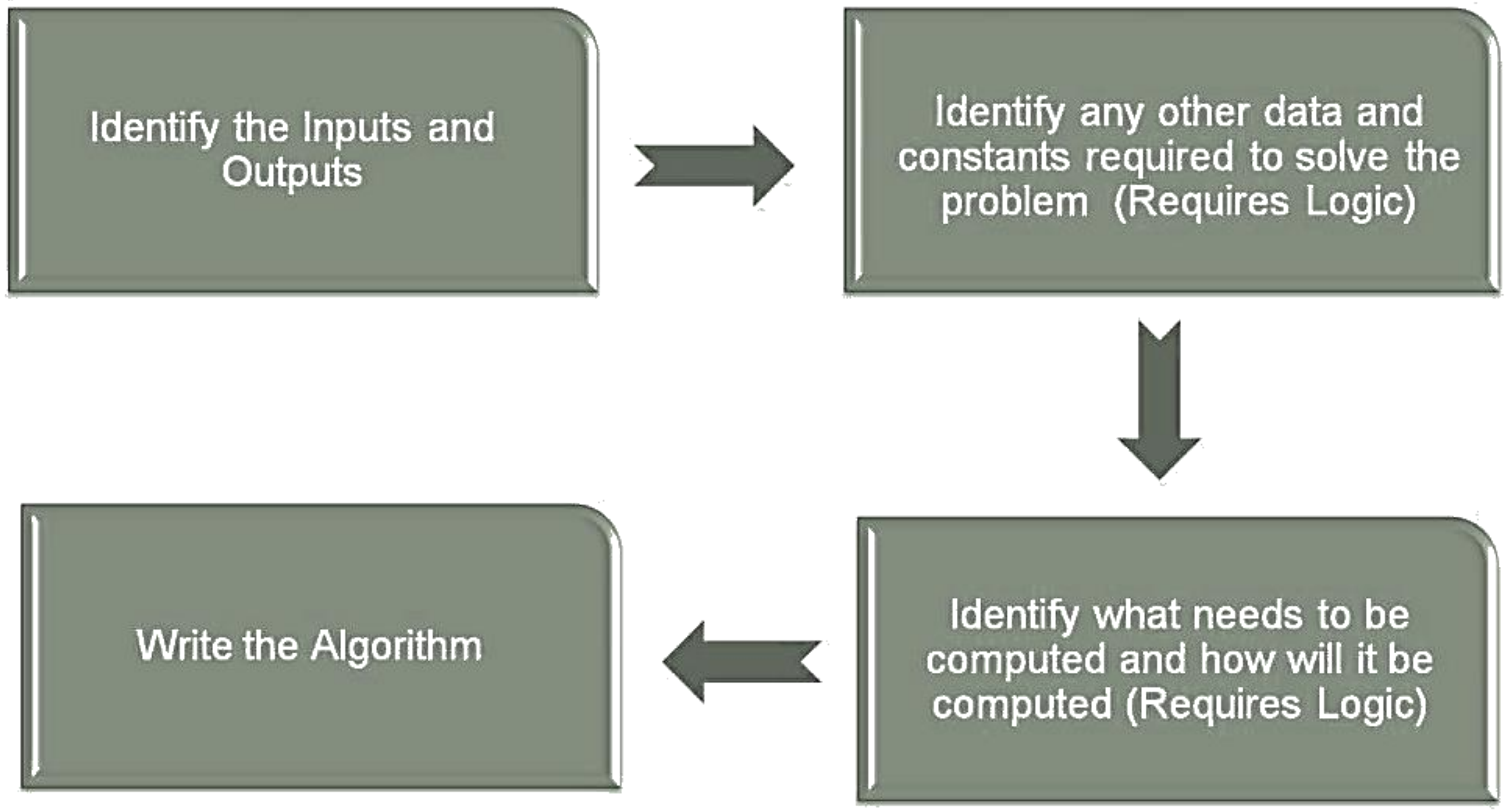
[tells the end of algorithm]

Properties of an algorithm



Finiteness	Must terminate after a finite number of steps
Definiteness	Action in each step to be carried out must be rigorously and unambiguously specified
Input	Has zero or more inputs that are provided to it initially or dynamically
Output	Has one or more outputs; quantities that have a specified relation to inputs
Effectiveness	operations must all be sufficiently basic

Steps to develop an algorithm


Identify the Inputs and Outputs
Identify any other data and constants required to solve the problem (Requires Logic)
Identify what needs to be computed and how will it be computed (Requires Logic)
Write the Algorithm

Compute the area of circle

```
Name of the algorithm : Compute the area of a circle
Step1:        Start
Step 2:       Input  radius
Step 3:       [Compute the area]
              Area ← 3.1416 * radius  *  radius
Step 4:       [Print the Area]
              Print  'Area of a circle =', Area
Step 5:       [End of algorithm]
              Stop
```

Interchange values of two variables

Name of the algorithm: Interchange values of 2 variables

```
Step 1:     Start
Step 2:     Input A,B
Step 3:     temp ← A
Step 4:     A←B
Step 5:     B←temp
Step 6:     Print 'A=' , A
            Print 'B=' , B
Step 7:     [End of Algorithm]
            Stop
```

Largest of 3 numbers

```
Name of the algorithm: Find largest of 3 numbers
Step 1:  Start
Step 2:  [Read the values of A, B and C]
         Read A, B, C
Step 3:  [Compare A and B]
         if  A>B Go to step 5
Step 4:  [Otherwise compare B with C]
         if  B>C then
                 Print 'B' is largest'
         else
                 Print 'C' is largest'
                 Go to Step 6
Step 5:  [Compare A and C for largest]
         if  A>C then
                 Print 'A' is largest'
         else
                 Print 'C' is largest'
Step 6: [End of the algorithm]
         Stop
```

Largest of 3 Numbers contd

```
Step 5:     [Compare A and C for largest]
                 if  A>C then
                 Print ‘A’ is largest’
            else
                 Print ‘C’ is largest’
Step 6: [End of the algorithm]
        Stop
```

Factorial of a number

Name of the algorithm: Compute the factorial of a number

```
Step1:      Start
Step 2:     Input N
Step 3:     fact ← 1
Step 4:     for count=1 to N in step of 1 do
                begin
                   fact←fact*count
                end
Step 5:     Print 'fact of N=', fact
Step 6:     [End of algorithm]
            Stop
```

Tutorial on Algorithms

Write the algorithm to

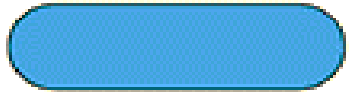
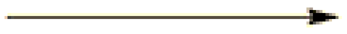
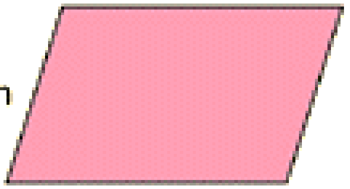
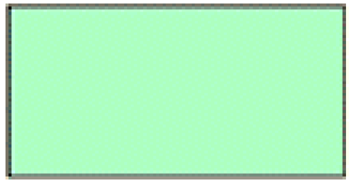
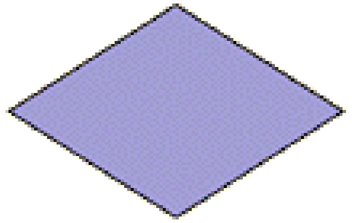
- Find the area of triangle when three sides are given.
- Add two integers
- Find the sum of digits of a number
- Find the Sum and Mean of first N natural numbers

Flowcharts

✓In Computer Science, **Flow chart** is used to represent algorithm which basically provides a solution to any computational problem.

- **Flowchart:** A graphical/pictorial representation of computation

Basic Flowchart Symbols

Name	Symbol	Use in flowchart
Oval		Denotes the beginning or end of a program.
Flow line		Denotes the direction of logic flow in a program.
Parallelogram		Denotes either an input operation (e.g., INPUT) or an output operation (e.g, PRINT).
Rectangle		Denotes a process to be carried out (e.g., an addition).
Diamond		Denotes a decision (or branch) to be made. The program should continue along one of two routes (e.g., IF/THEN/ELSE).

Area of the circle

Name of the algorithm:
Compute the area of a circle

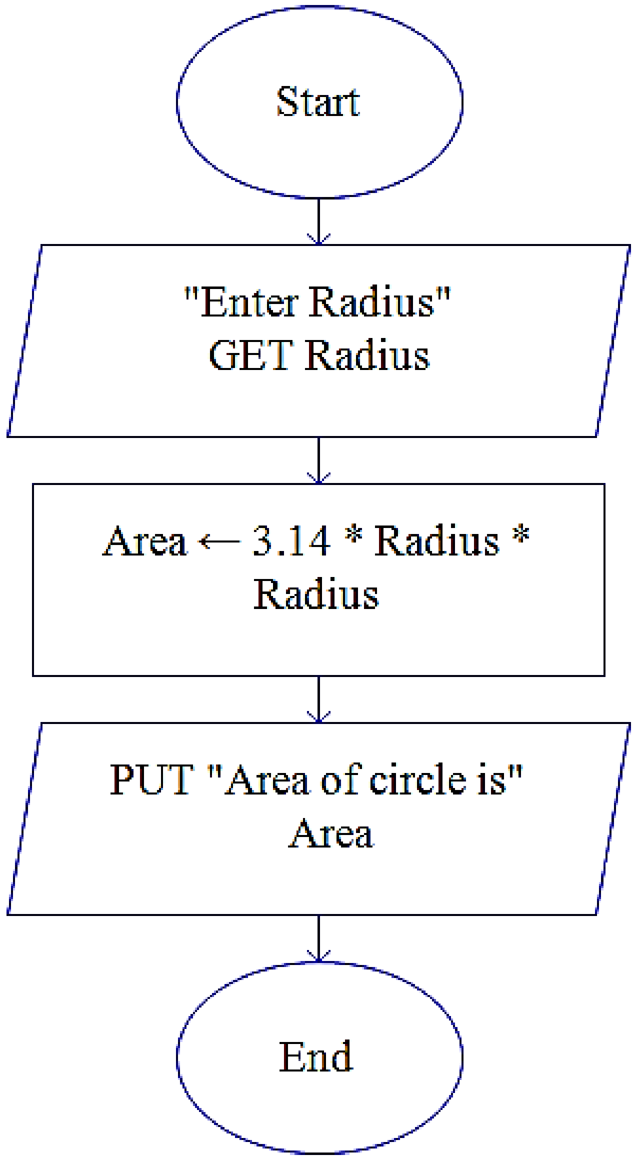
```
Step1:  Input  radius

Step 2: [Compute the area]
        Area ← 3.1416 * radius*radius

Step 3: [Print the Area]
        Print  'Area of a circle =', Area

Step 4: [End of algorithm]
        Stop
```

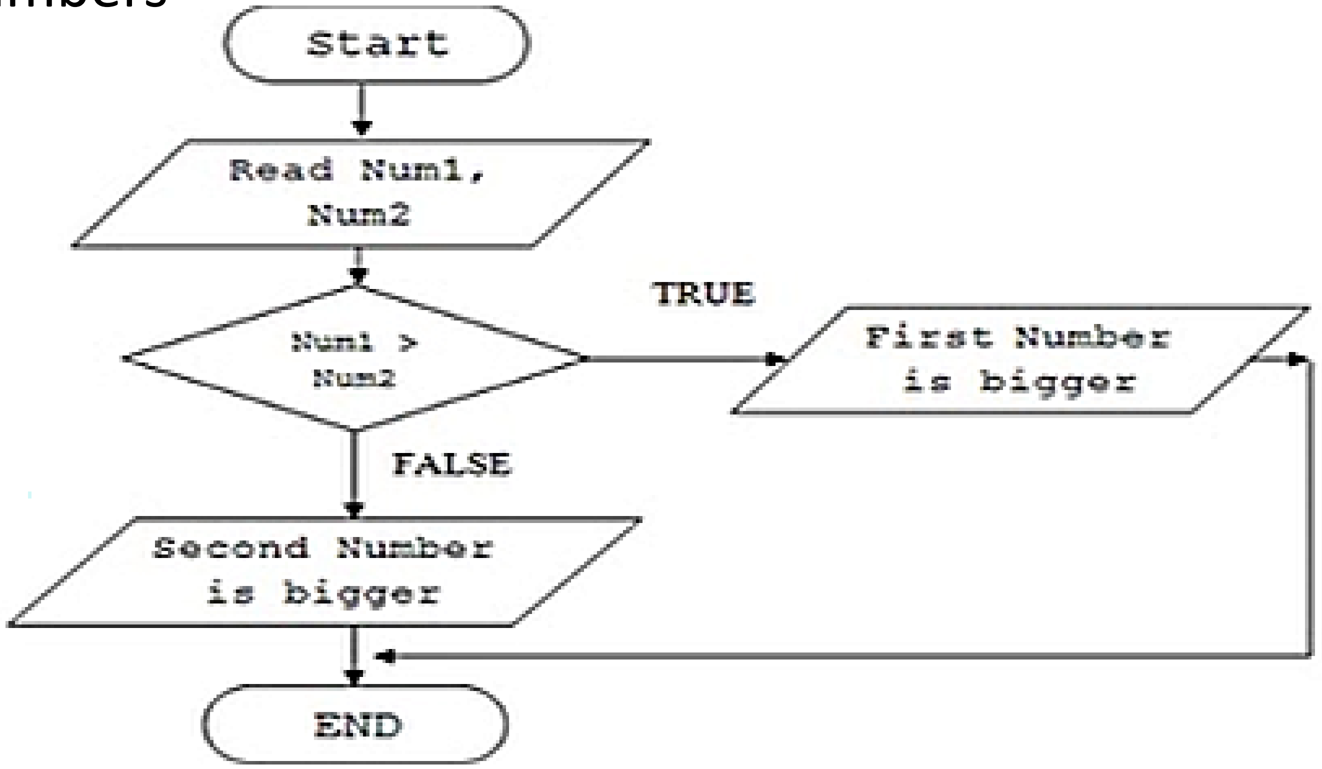
Flowchart



Comparing two numbers

```
Name of the algorithm: Comparing 2 numbers
Step 1:  Start
Step 2: Input num1, num2
Step 3: if num1 > num2 then
            Print num1 is bigger
        else
            Print num2 is bigger
Step 4: end
```

Flowchart

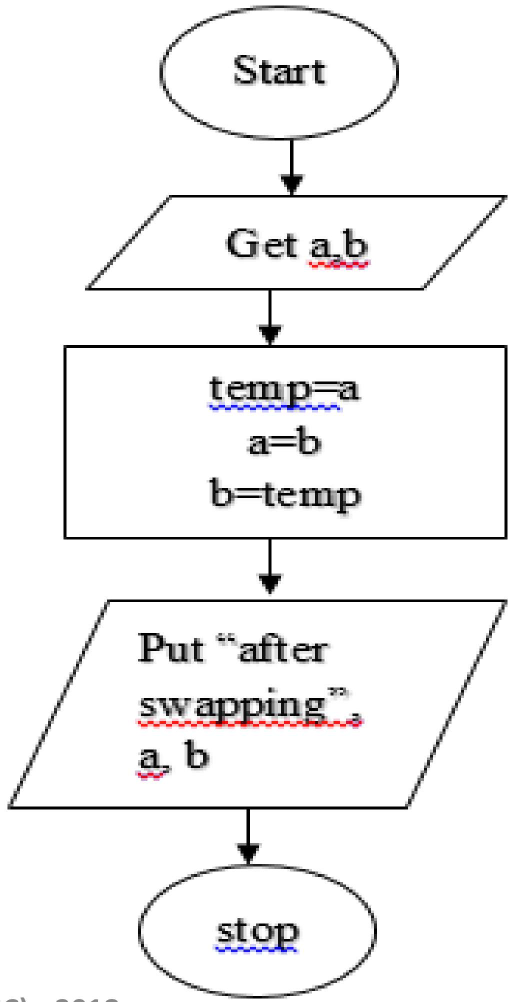


Swapping two numbers

Name of the algorithm: Swapping 2 numbers

```
Step1:  Input  two numbers
Step 2: [swapping]
            temp=a
            a=b
            b=temp
Step 3: [Print]
          Print  'after swapping=', a, b
Step 4: [End of algorithm]
          Stop
```

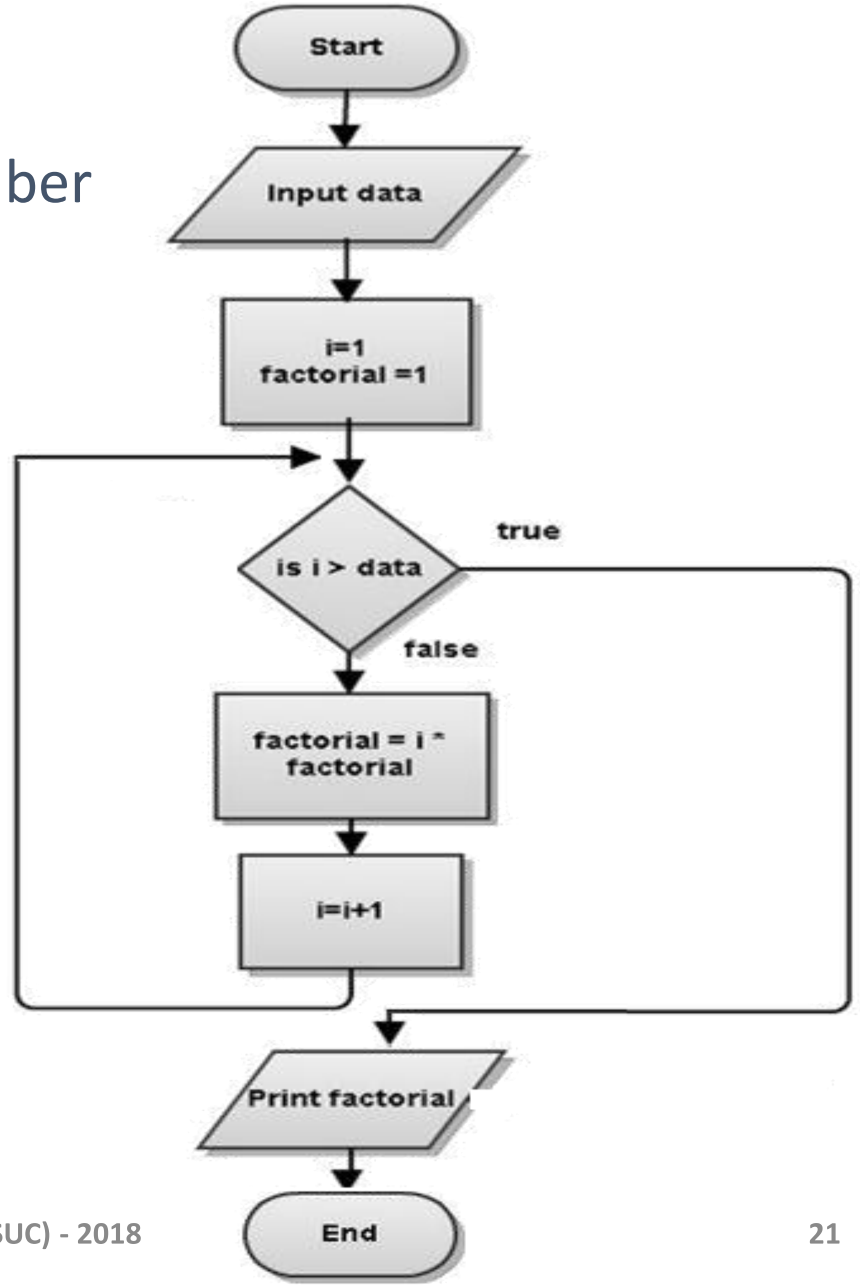
Flowchart



Find Factorial of given no

Name of the algorithm: Compute the factorial of a number

```
Step1:  Start
Step 2: Input data
Step 3: factorial ←1
Step 4: for i=1 to N in step of 1 do
        begin
          factorial←factorial*i
        end
Step 5: Print ‘fact of N=‘, factorial
Step 6: [End of algorithm]
         Stop
```



Key features of flow chart

- ✓Diagrammatic / visual / graphical representation of computation of an algorithm/pseudo code
- ✓Easier to understand and analyze the problem and it’s solution before programming
- ✓Machine independent
- ✓ Well suited for any type of logic

Tutorial

- Write the flowchart to find the area of triangle when three sides are given
- Write the flowchart to add two integers

Summary

✓What is Algorithm and Flowchart

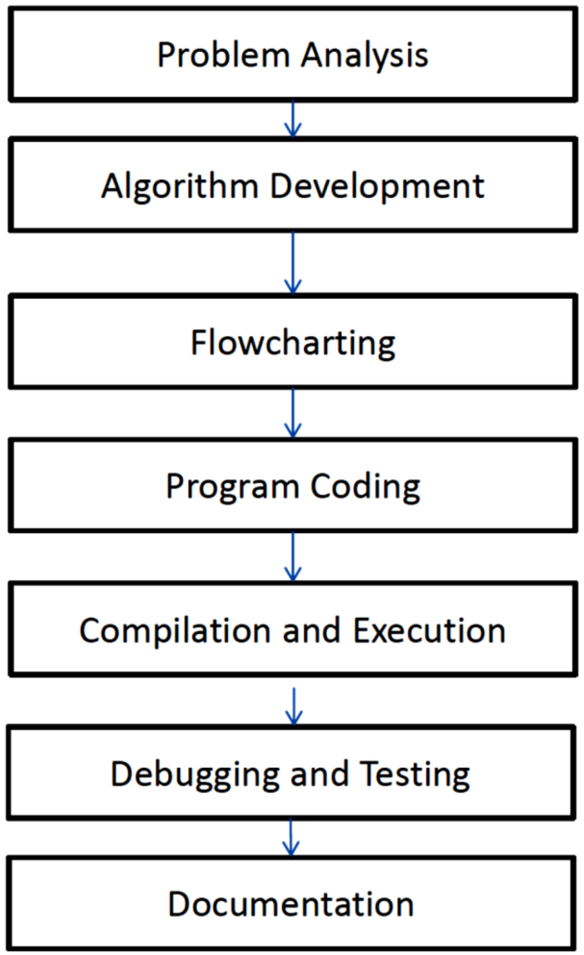
✓Writing algorithms and drawing flowcharts for simple problems

Problem Solving Using Computers

Problem solving is a part of our day to day activity.

- **Algorithm**
- **Flowchart**
- **Code**
- **Documenting**

Steps in Problem Solving



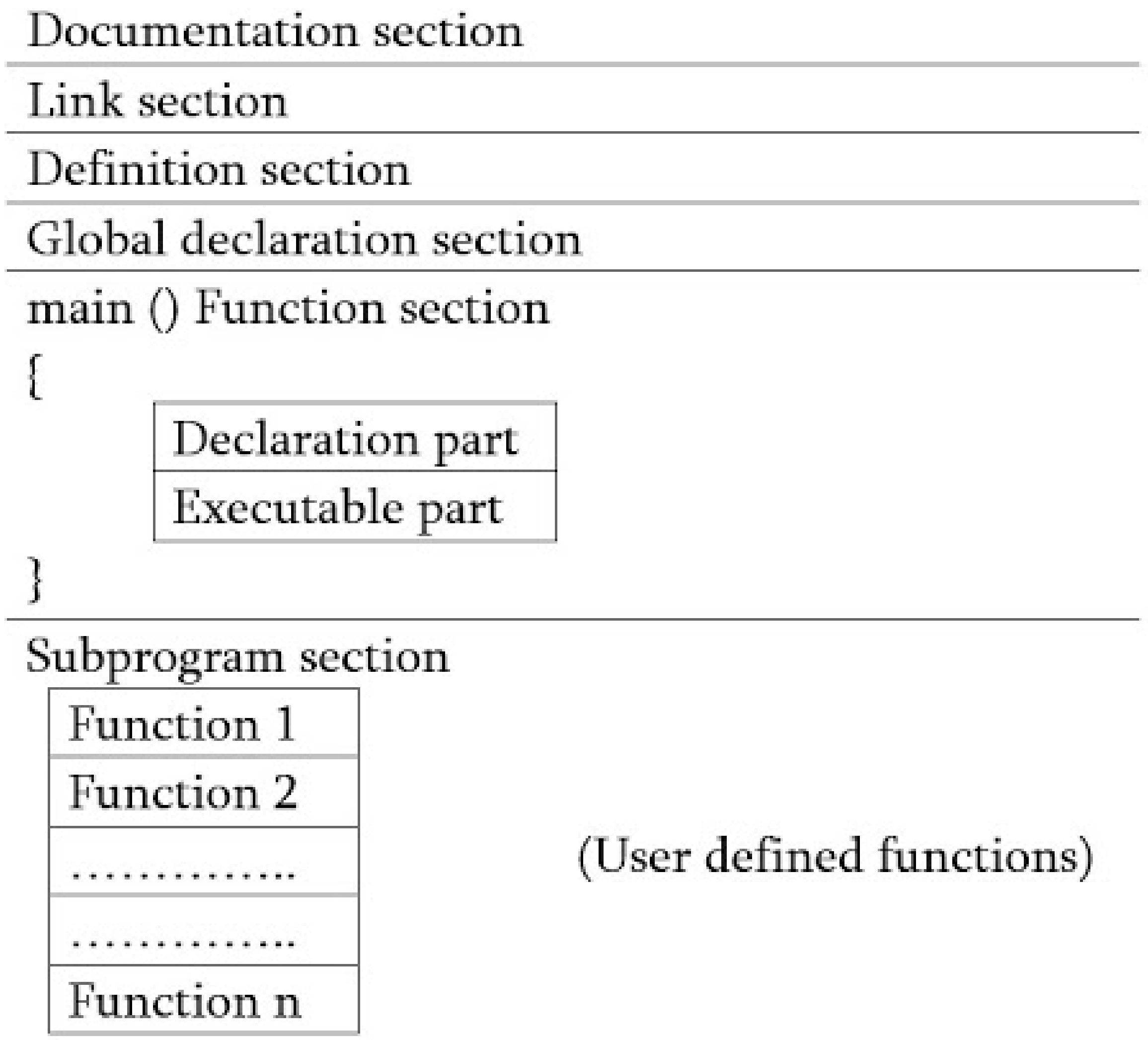
C Program Structure

- An example of simple program in C

```
#include <stdio.h>

void main()
{
   printf(“I love programming\n”);
   printf(“You will love it too once ”);
   printf(“you know the trick\n”);
}
```

General Structure of C program



Documentation section

Link section

Definition section

Global declaration section

main () Function section

{

Declaration part

Executable part

}

Subprogram section

Function 1

Function 2

…………..

…………..

Function n

(User defined functions)

General Structure of C program

1. **Documentation section:**

   The documentation section consists of a set of comment lines giving the name of the program, the author and other details, which the programmer would like to use later.

2. **Link section:** The link section provides instructions to the compiler to link functions from the system library such as using the *#include directive*.

3. **Definition section:** The definition section defines all symbolic constants such using the *#define directive*.

4. **Global declaration section:** There are some variables that are used in more than one function. Such variables are called global variables and are declared in the global declaration section that is outside of all the functions. This section also declares all the *user-defined functions*.

General Structure of C program

5. **main () function section:** Every C program must have one main function section. This section contains two parts; declaration part and executable part

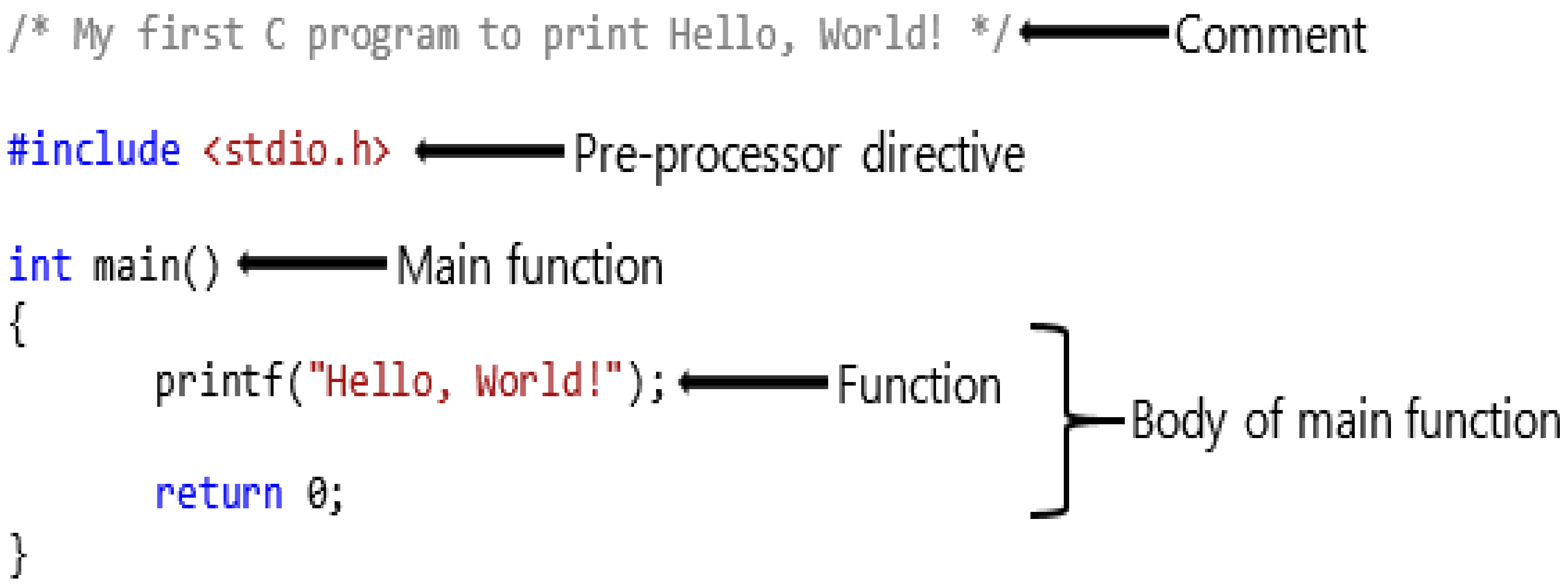
   a) **Declaration part:** The declaration part declares all the *variables* used in the executable part.

   b) **Executable part:** There is at least one statement in the executable part. These two parts must appear between the opening and closing braces. The *program execution* begins at the opening brace and ends at the closing brace. The closing brace of the main function is the logical end of the program. All statements in the declaration and executable part end with a semicolon.

6. **Subprogram section:** If the program is a *multi-function program* then the subprogram section contains all the *user-defined functions* that are called in the main () function. User-defined functions are generally placed immediately after the main () function, although they may appear in any order.

All sections, except the main () function section may be absent when they are not required.

Hello world program



```
/* My first C program to print Hello, World! */

#include <stdio.h>

int main()
{
        printf("Hello, World!");

        return 0;
}
```

Formatting in C

- Statements are terminated with semicolons.
- Indentation to be used for increased readability.
- Free format: white spaces and indentation is ignored by compiler.
- New line is represented by **n** (Escape sequence).

Formatting in C

- **C is case sensitive** – pay attention to lower and upper case letters when typing !
  - All keywords and standard functions are lower case
  - Typing **MAIN, Main** etc instead of **main** is a compiler error
- Strings are placed in double quotes

C program for reading a number and display it on the screen

```
#include<stdio.h>
int main()
{
 int num;
 printf("\nEnter the number: ");
 scanf("%d", &num);
 printf("The number read is: %d", num);
return(0);
}
```

scanf()

- ***scanf ( )*** is used to obtain the value from the user through an input device (keyboard).

- Eg: **scanf(“*%d*”, &number);**

```
int main(){
 int num;
 printf("\nEnter the number: ");
 scanf("%d", &num);
 printf(“The number read is: %d", num);
return(0);
}
```

C program essentials: *Format specifiers*

Character group	Meaning
%c	Read a single character
%d	Read a decimal integer
%e	Read a floating point number
%f	Read a floating point number
%g	Read a floating point number
%h	Read a short int number
%i	Read a decimal or hexadecimal or octal number
%o	Read an octal number
%p	Read a pointer
%s	Read a string
%u	Read a unsigned integer
%x	Read a hexadecimal number

Similarly, the format specifiers are used with *printf( )* function for printing/displaying.

Input/ Output functions

- C provides the ***printf( )*** to display the data on the output device (monitor).
- It is included in stdio.h
- Examples are:
  - printf("**programming is an art**");
  - printf("*%d*", number);
  - printf("*%f%f*", p, q);

```
int main() {
 int num;
 printf("\nEnter the number: ");
 scanf("%d", &num);
 printf("The number read is: %d", num);
return(0);
}
```

Adding two integers

```
#include <stdio.h>
int main( void )
{/* start of function main */
            int sum; /* variable in which sum will be stored */
            int integer1; /* first number to be input by user */
            int integer2; /* second number to be input by user */
            printf( "Enter first integer\n" );
            scanf( "%d", &integer1 ); /* read an integer */
            printf( "Enter second integer\n" );
            scanf( "%d", &integer2 ); /* read an integer */
            sum = integer1 + integer2; /* assign total to sum */
            printf( "Sum is %d\n", sum ); /* print sum */
            return 0; /* indicate that program ended successfully */
} /* end of function main */
```

Syntax and Logical errors

- **Syntax errors:** violation of programming language rules (grammar).
  - Detected by the compiler
  - Eg: printf ("hello world")   // semicolon missing

- **Logical errors:** errors in meaning:
  - Programs are syntactically correct but don't produce the expected output
  - User observes output of running program

Summary

✓Problem solving !

✓Program structure

✓Formatting

✓Input / Output

✓Simple C programs