



**ACHARYA**

BENGALURU, INDIA

WHERE THE  
**WORLD COMES  
TO LEARN**

# **Principles of Programming using C**

**Subject code : BPOPS203**

**Prof. Supriya C**  
**Assistant Professor**  
**Department of ISE**  
**Acharya Institute Of Technology**



**ACHARYA**

Dept of ISE

# Syllabus

Course Title:	<b>Principles of Programming using C</b>	
Course Code:	<b>BPOPS103/203</b>	CIE Marks 50
Course Type (Theory/Practical /Integrated )	Integrated	SEE Marks 50
		Total Marks 100
Teaching Hours/Week (L:T:P: S)	2:0:2	Exam Hours 3+2
Total Hours of Pedagogy	40 hours	Credits 03
	<b>Course Objectives:</b>  CLO 1. Elucidate the basic architecture and functionalities of a Computer CLO 2. Apply programming constructs of C language to solve the real-world problems CLO 3. Explore user-defined data structures like arrays, structures and pointers in implementing solutions to problems CLO 4. Design and Develop Solutions to problems using structured programming constructs such as functions and procedures	



**ACHARYA**

Dept. of ISE

# Syllabus



ACHARYA

Dept. of ISE

Module-1 (6 Hours of Pedagogy)	
	<p><b>Introduction to C:</b> Introduction to computers, input and output devices, designing efficient programs. Introduction to C, Structure of C program, Files used in a C program, Compilers, Compiling and executing C programs, variables, constants, Input/output statements in C,</p> <p><b>Textbook: Chapter 1.1-1.9, 2.1-2.2, 8.1 - 8.6, 9.1-9.14</b></p>
<b>Teaching-Learning Process</b>	Chalk and talk method/PowerPoint Presentation/ Web Content: <a href="https://tinyurl.com/4xmrexre">https://tinyurl.com/4xmrexre</a>
Module-2 (6 Hours of Pedagogy)	
	<p>Operators in C, Type conversion and typecasting.</p> <p><b>Decision control and Looping statements:</b> Introduction to decision control, Conditional branching statements, iterative statements, nested loops, break and continue statements, goto statement.</p> <p><b>Textbook: Chapter 9.15-9.16, 10.1-10.6</b></p>
<b>Teaching-Learning Process</b>	Chalk and talk method/PowerPoint Presentation
Module-3 (8 Hours of Pedagogy)	
<p><b>Functions:</b> Introduction using functions, Function definition, function declaration, function call, return statement, passing parameters to functions, scope of variables, storage classes, recursive functions.</p> <p><b>Arrays:</b> Declaration of arrays, accessing the elements of an array, storing values in arrays, Operations on arrays, Passing arrays to functions, two dimensional arrays, operations on two-dimensional arrays, two-dimensional arrays to functions, multidimensional arrays, applications of arrays.</p> <p><b>Textbook: Chapter 11.1-11.10, 12.1-12.10, 12.12</b></p>	

# Syllabus

## Module-3 (8 Hours of Pedagogy)

**Functions:** Introduction using functions, Function definition, function declaration, function call, return statement, passing parameters to functions, scope of variables, storage classes, recursive functions.

**Arrays:** Declaration of arrays, accessing the elements of an array, storing values in arrays, Operations on arrays, Passing arrays to functions, two dimensional arrays, operations on two-dimensional arrays, two-dimensional arrays to functions, multidimensional arrays, applications of arrays.

**Textbook:** Chapter 11.1-11.10, 12.1-12.10, 12.12

<b>Teaching-Learning Process</b>	Chalk and talk method/PowerPoint Presentation
----------------------------------	---

## Module-4 (6 Hours of Pedagogy)

**Strings and Pointers:** Introduction, string taxonomy, operations on strings, Miscellaneous string and character functions, arrays of strings. **Pointers:** Introduction to pointers, declaring pointer variables, Types of pointers, Passing arguments to functions using pointers

**Textbook:** Chapter 13.1-13.6, 14-14.7

## Module-5 (6 Hours of Pedagogy)

**Structure, Union, and Enumerated Data Type:** Introduction, structures and functions, Unions, unions inside structures, Enumerated data type.

**Files:** Introduction to files, using files in C, reading and writing data files. , Detecting end of file

**Textbook:** Chapter 15.1 – 15.10, 16.1-16.5

<b>Teaching-Learning Process</b>	Chalk and talk method/PowerPoint Presentation
----------------------------------	---



ACHARYA

Dept. of ISE

# Syllabus



Dept. of ISE

## Suggested Learning Resources:

### Textbooks

1. Computer fundamentals and programming in c, “Reema Thareja”, Oxford University, Second edition, 2017.

### Reference Books:

1. E. Balaguruswamy, Programming in ANSI C, 7th Edition, Tata McGraw-Hill.
2. Brian W. Kernighan and Dennis M. Ritchie, The ‘C’ Programming Language, Prentice Hall of India.

### Web links and Video Lectures (e-Resources):

1. [elearning.vtu.ac.in/econtent/courses/video/BS/15PCD23.html](http://elearning.vtu.ac.in/econtent/courses/video/BS/15PCD23.html)
2. <https://nptel.ac.in/courses/106/105/106105171/> MOOC courses can be adopted for more clarity in understanding the topics and verities of problem solving methods.



**ACHARYA**

Dept. of ISE

## **Course Outcomes (CO's):**

**After the completion of the course students will be able to**

**CO1: Illustrate basic concepts of Computer and C programming**

**CO2: Design the solution for the given problems and develop the same using C programming language**

**CO3: Apply the concepts of looping, branching, and decision-making statements for a given problem**

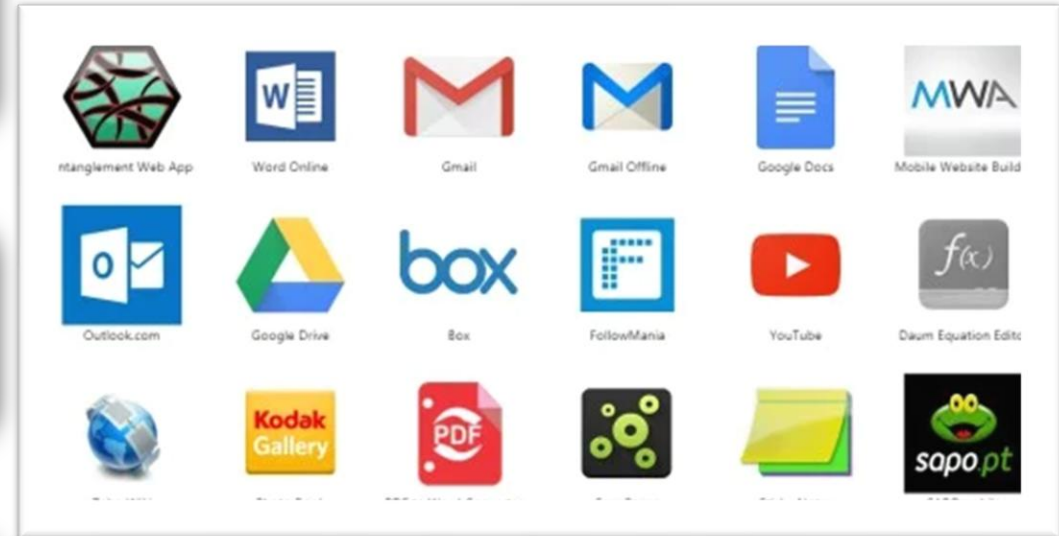
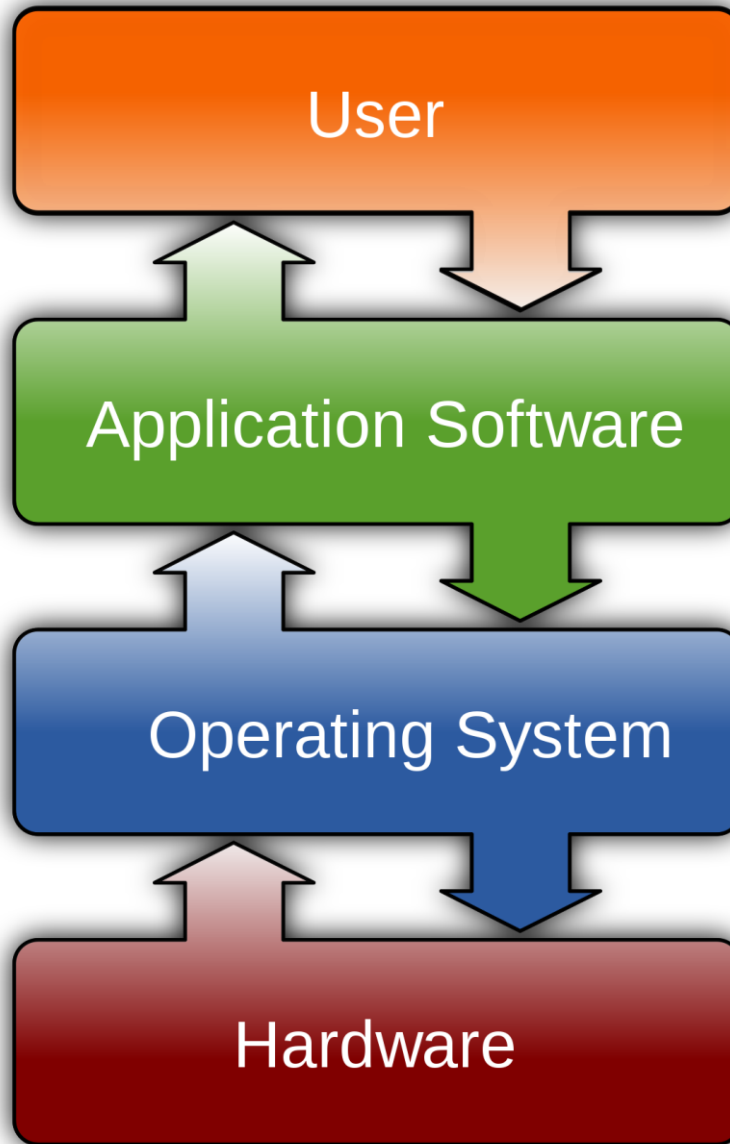
**CO4: Demonstrate the ability to write C programs using pointers, structures, unions and arrays**

**CO5: Develop modular applications using C programming language**

**CO 6: Implement programs using conditional and statements for a given problem.**

**CO 7 : Implement programs using structures, strings, Arrays, pointers and files for a given problem**

# Application Software V/S System Software

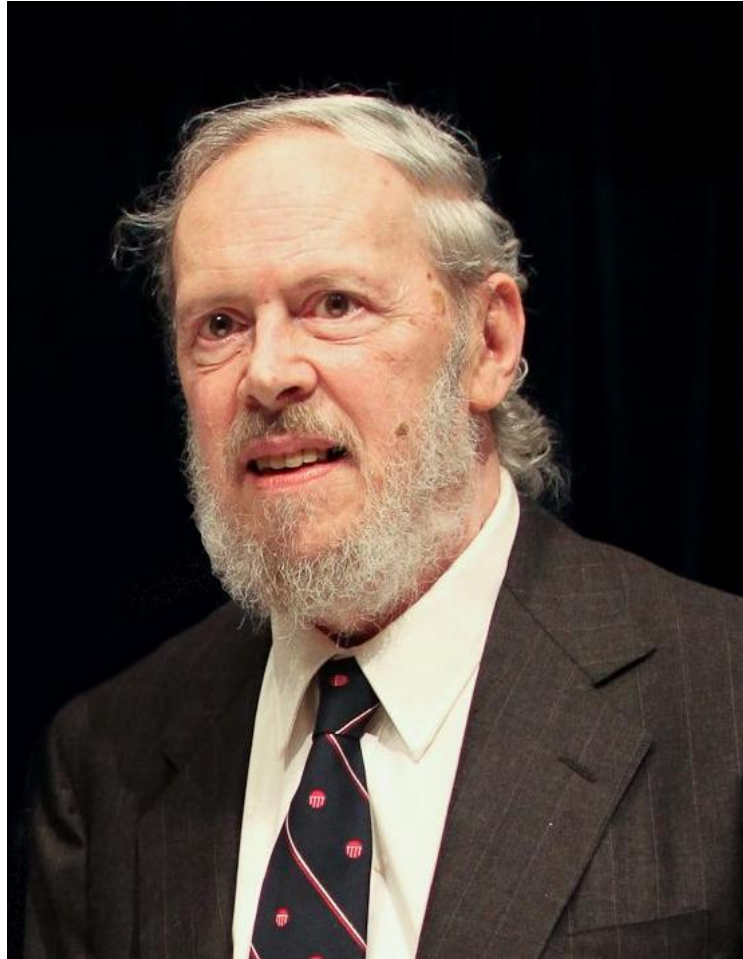


# C PROGRAMMING LANGUAGE

Applications of C Programming Language



## Dennis Ritchie



C, Computer Programming Language developed in the early 1970s by American computer scientist Dennis M. Ritchie at Bell Laboratories (formerly AT&T Bell Laboratories)

## **C Programming Language has following importance:**

1.C is **robust language** and has rich set of built-in functions, data types and operators which can be used to write any complex program

2. C has the capabilities of an **assembly language** (low level features) with the feature of high level language so it is well suited for writing both **system software** and **application software**

3.C is highly **portable language** i.e. code written in one machine can be moved to other which is very important and powerful feature.























4.C supports low level features like bit level programming and **direct access to memory** using **pointer** which is very useful for managing resource efficiently.

5.C has high level constructs and it is **more user friendly** as its syntaxes approaches to English like language.

# Use of C Programming

C language is used to develop **System applications** that forms major portion of **operating systems** such as **Windows, UNIX and Linux.**

- Database systems
- Graphics packages
- Word processors
- Spread sheets
- Operating system development
- Compilers and Assemblers
- Network drivers
- Interpreters

Language Rank	Types	Spectrum Ranking
1. Python	  	100.0
2. C++	  	99.7
3. Java	  	97.5
4. C	  	96.7
5. C#	  	89.4
6. PHP		84.9
7. R		82.9
8. JavaScript	 	82.6
9. Go	 	76.4
10. Assembly		74.1



Dept. of ISE

## Module 1

**Introduction to C:** Introduction to computers, input and output devices, designing efficient programs. Introduction to C, Structure of C program, Files used in a C program, Compilers, Compiling and executing C programs, variables, constants, Input/output statements in C,

**Textbook: Chapter 1.1-1.9, 2.1-2.2, 8.1 - 8.6 ,9.1-9.14**

# Computer

**“An electronic device which accepts the data, performs the required mathematical and logical operations at high speed and outputs the result”**

- Computer is an electronic device, which takes input from the user in the form of data and instructions.
- Once the computer receives the data, it waits for the user's instructions to process the data.
- On receiving the instructions from the user, the computer processes the data and generates the output and displays it to the user.
- The data that is given as an input to the computer can be text, numerals, audio, video and image etc.

## Tasks of the Computer

- **Input:** Sending the data and command to the computer is known as input.
- **Processing:** Work done by the computer with the help of processing hardware and software to produce results is known as processing.
- **Output:** The result displayed by the computer is called as output.
- **Storage:** A place to save results inside or outside the computer is known as storage.

# Characteristics of Computers:

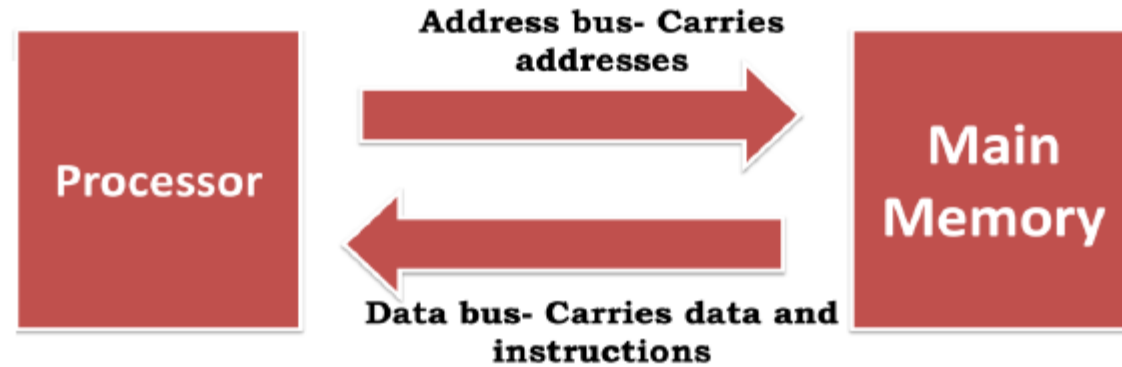
**Speed:** Computers can perform millions of operations per second. The speed of computers is usually given in nanoseconds/picoseconds where  $1\text{ns} = 1 \times 10^{-9}$  seconds and  $1\text{ps} = 1 \times 10^{-12}$  seconds

- **Accuracy:** A computer can always give accurate results, provided the correct input and instructions as input. If the input is wrong, then output will be erroneous. This is called “**garbage-in, garbage-out**”
- **Automation:** Computers performs the task without any human intervention.
- **Diligence:** Unlike humans, computers never get tired of repetitive task.
- **Versatile:** Computers can perform multiple tasks of different nature at the same time.
- **Memory:** Computers have internal or primary memory and External or Secondary memory to store data and programs.
- **No IQ:** Computers do not have any decision making abilities of their own. They need guidance to perform various tasks.
- **Economical:** Computers are considered as short-term investment for achieving long term gains. Reduce Manpower, They save time, money and energy.



Dept. of ISE

# Types of Stored Program Computers



Von Neumann Architecture with shared memory for instructions and data



Von Neumann Architecture with separate memory for instructions and data



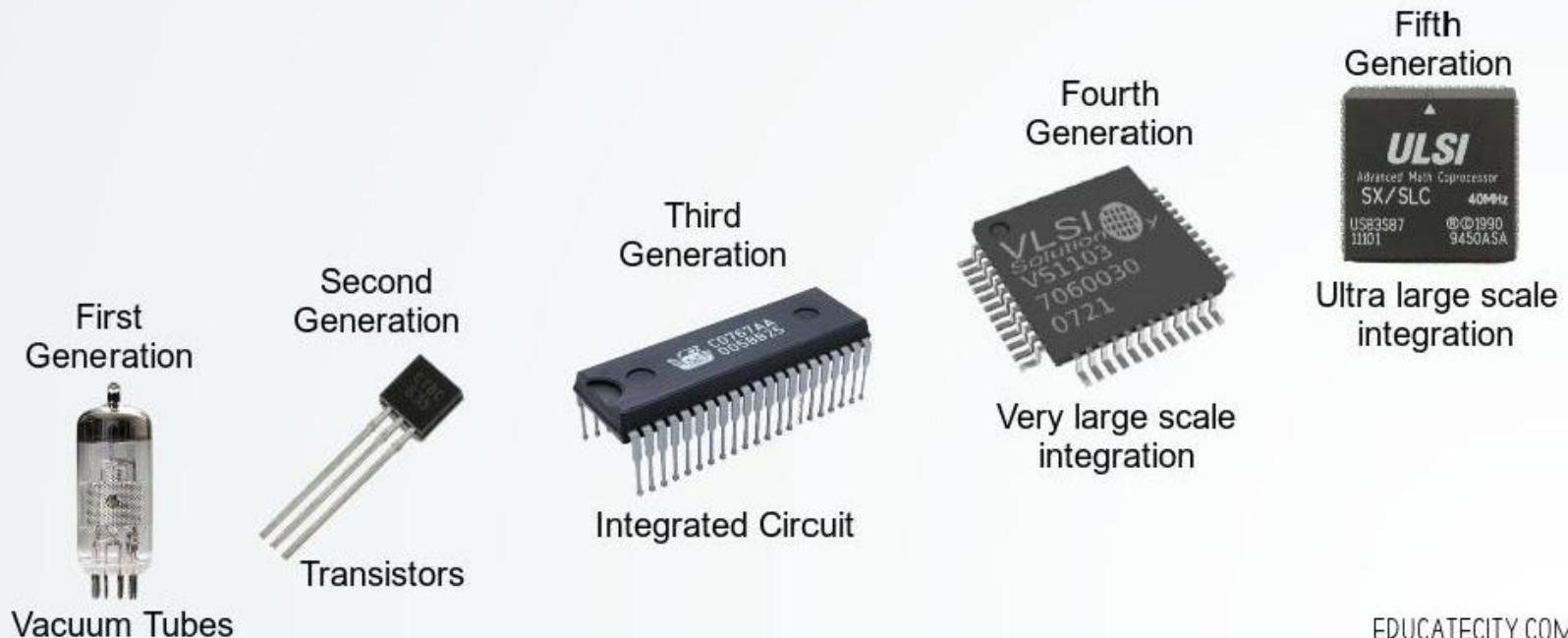
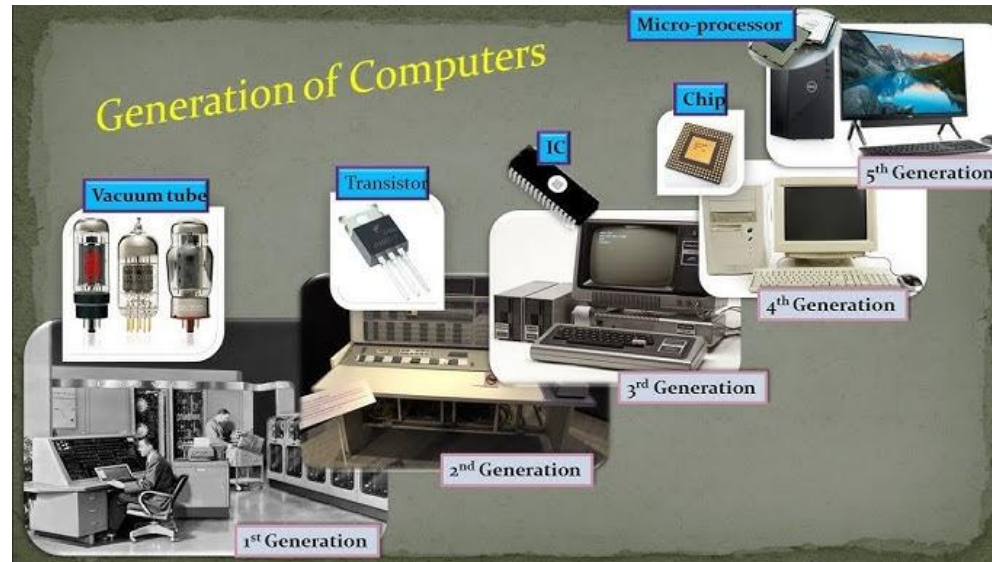
ACHARYA

Dept. of ISE

# History of Computers



Dept. of ISE



# History of Computers



-	Year	Manufactured using	Primary memory	Storage device
First generation	1942-1955	Vacuum tubes	Electromagnetic relay	Punch cards
Second generation	1955-1964	Transistors	Magnetic Core Memory	Magnetic tapes and disks
Third generation	1964-1975	IC's	Large Magnetic Core Memory	Magnetic tapes and disks
Fourth generation	1975-1989	Microprocessor	Semiconductor memory	Magnetic tapes and floppy disks
Fifth generation	1989-Present	AI	Semiconductor memory	Magnetic tapes and floppy disks



**Vacuum Tubes**

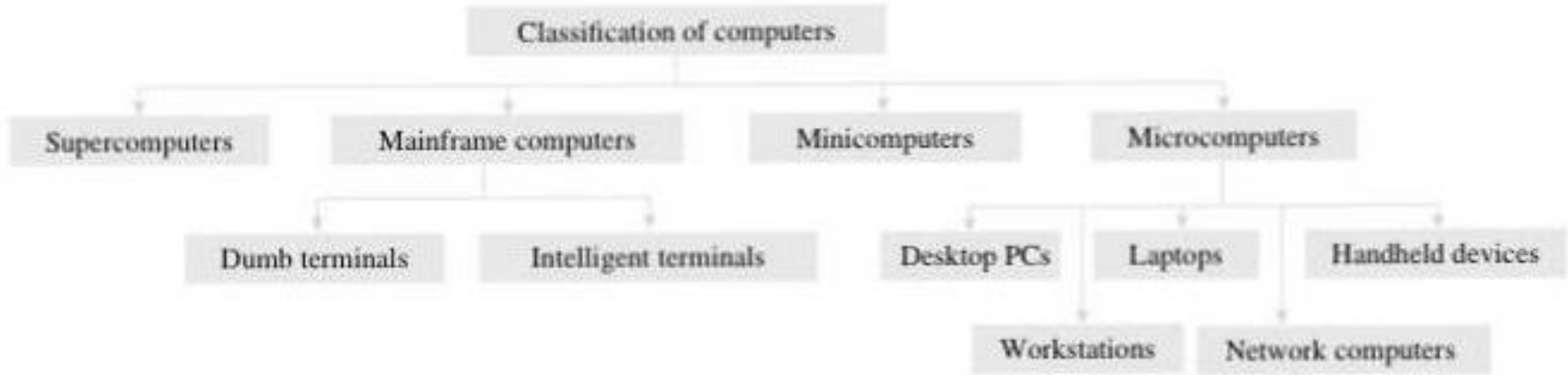


**Transistors**



**Integrated Circuits**

# Classification of Computers



## 1. Supercomputers:

- It is the fastest, most powerful, and most expensive computer.
- It was first developed in 1980s to process large amounts of data and solve complex scientific problems.
- They use parallel processing technology and can perform trillion calculations in a second.
- A single supercomputer can support thousands of users at same time.
- They are used for weather forecasting, nuclear energy research, aircraft design, automotive design, online banking etc.
- Examples of supercomputers are: CRAY-1, CRAY-2, ETA A-10.



ACHARYA

Dept. of ISE

# Classification of Computers

## 2. Mainframe Computers:

- They are large scale computers but smaller than supercomputers.
- These are very expensive and need a very large clean room with air conditioning, making them very costly to deploy.
- Mainframes can also support multiple processors.

### 2a. Dumb terminals:

It consists of only monitor and keyboard and uses mainframe's system CPU and storage device.

### 2b. Intelligent terminals:

It have their own processor and perform processing operations but do not have their own storage space.

- Mainframe computers are typically used as servers on WWW
- They are also used in organizations such as banks, airline companies, universities where large number of people access the data frequently.
- Examples of Mainframe computers are: IBM S/390, Control Data CYBER 176, and Amdahl 580.



ACHARYA

Dept. of ISE

# Classification of Computers

## 3. Minicomputers:

- They are smaller, cheaper, and slower than mainframes.
- They are also called Midrange computers.
- The capabilities of these fall between mainframe and personal computers.
- They are widely used in business, education, hospitals, government organizations, etc.
- Some minicomputers are designed in such a way that it can be either used by single user or multiple users.
- Single-user minicomputers are used for performing complex design tasks.
- The first Minicomputer was introduced by Digital Equipment Corporation in mid-1960's.



# Classification of Computers

## **4. Microcomputers:**

- They are commonly known as PC's.
- The first microcomputer was designed by IBM in 1981 and was named as IBM-PC.

### **4a. Desktop PCs:**

- It is the most popular model of PC's.

### **4b. Laptops:**

- They are small microcomputers that easily fit inside a briefcase
- The memory and storage capacity of a laptop is almost equivalent to a desktop computer.
- They also have hard disk drives.
- For input, laptops have built-in keyboard and touch pad.



ACHARYA

Dept. of ISE

# Classification of Computers

## 4c. Workstations:

- A workstation is nothing but a computer like a personal computer, but with more powerful microprocessor and large, high resolution monitor for better quality of pictures along with large amount of memory.

**Advantages:** High Resolution, Accelerated graphics cards- they are used in video editing and animations, Powerful machines and used for architectural or engineering design.

**Disadvantages:** Workstations are costly, Not Portable.



Dept. of ISE

# Classification of Computers

## 4d. Network computers:

- Network computers have less processing power, memory, and storage than a desktop computer.
- These are specially designed to be used as terminals in a networked environment.
- For example, some network computers are specifically designed to access data stored on a network (including the Internet and intranet).
- Some network computers do not have any storage space and merely rely on the network's server for data storage and processing tasks. The concept of network computers had become popular in the mid-1990s
- Network computers that are specifically designed to access only the Internet or intranet are often known as Internet PCs or Internet boxes.

# Classification of Computers

## **Network computers:**

- Some network computers used in homes do not even have a monitor. Such computers may be connected to a television, which serves as the output device.
- The most common example of a home-based network computer is Web TV, which enables the user to connect a television to the Internet.
- The Web TV is equipped with a special set-top box that is used to connect to the Internet.
- The set-top box also provides controls to enable the user to navigate the Internet, send and receive e-mails, and to perform other tasks on the network while watching television.
- The other reason for the popularity of network computers is that they are cheaper to purchase and maintain than PCs.

# Classification of Computers

## Smart Phones:

- A smart phone is an electronic hand held device that has the functionality of a mobile phone and PDA.
- This is achieved by adding mobile phone features to an existing PDA.

## Tablet Computers:

- A tablet PC is a personal computer and looks like a standard slate.
- The display system can be rotated by 180 degree.
- It has all the features of a laptop and allows the user to write on the screen using digital pen. This pen is called stylus. This pen is also used to tap on the icon so as to select an item.

# Applications of Computers

**1. Word Processing:** Word Processing software enables user to read and write documents.

**2. Internet:** The internet is a network that connects the computers all over the world.

**3. Digital video or Audio Composition:** Computers make Audio and Video editing very simple.

**4. Desktop publishing:** This software enables to create page layouts for entire books.

**5. E-Business:** It is a process of conducting business via Internet.

**6) Bioinformatics:** Bioinformatics is an interdisciplinary field of science that develops methods and software tools for understanding biological data, especially when the data sets are large and complex.



# Applications of Computers

**7) Health care:** In health care, it is used to store health records, surgical procedures for better diagnosis and treatment.

**8) Geographic Information System and Remote Sensing:** It is a computer based tool for mapping and analyzing Earth's features.

**9) Meteorology:** It is the study of atmosphere.

- Weather Forecasting
- Aviation
- Agriculture
- Nuclear

**10) Multimedia and Animation:** which combines still images, moving images, text, sound.

**11) Legal System:** Computers are used by lawyers to shorten the time required to conduct legal precedent and case research.

**12) Retail Business:** Computers are used here to enter orders, calculate costs, and print receipts.



# Applications of Computers

**13) Sports:** Computers are used here to compile statistics, identify weak and strong players, sell tickets etc.

**14) Travel and Tourism:** Computer are used to prepare tickets, monitor the train or airplane route, and guide the plane to a safe landing.

**15) Simulation:** Supercomputers that can process enormous amount of data are widely used in simulation tests.

**16) Astronomy:** Spacecrafts are usually monitored using computers that not only keep a continuous record of the voyage and of the speed, directions, fuel and temperature, but also suggest corrective actions when vehicle make a mistake.

**17) Education:** Teachers use computers to develop instructional material.

**18) Industry and Engineering:** Computers are found in all kind of industry like thermal power plants, oil refineries, chemical industry, Computer-aided designing, Computer aided manufacturing.



ACHARYA

Dept. of ISE

# Applications of Computers

**13) Sports:** Computers are used here to compile statistics, identify weak and strong players, sell tickets etc.

**14) Travel and Tourism:** Computer are used to prepare tickets, monitor the train or airplane route, and guide the plane to a safe landing.

**15) Simulation:** Supercomputers that can process enormous amount of data are widely used in simulation tests.

**16) Astronomy:** Spacecrafts are usually monitored using computers that not only keep a continuous record of the voyage and of the speed, directions, fuel and temperature, but also suggest corrective actions when vehicle make a mistake.

**17) Education:** Teachers use computers to develop instructional material.

**18) Industry and Engineering:** Computers are found in all kind of industry like thermal power plants, oil refineries, chemical industry, Computer-aided designing, Computer aided manufacturing.



ACHARYA

Dept. of ISE

# Applications of Computers

**18) Industry and Engineering:** Computers are found in all kind of industry like thermal power plants, oil refineries, chemical industry, Computer-aided designing, Computer aided manufacturing.

**19) Decision support systems:** Computers help managers to analyze their organization data to understand the present scenario of the business.

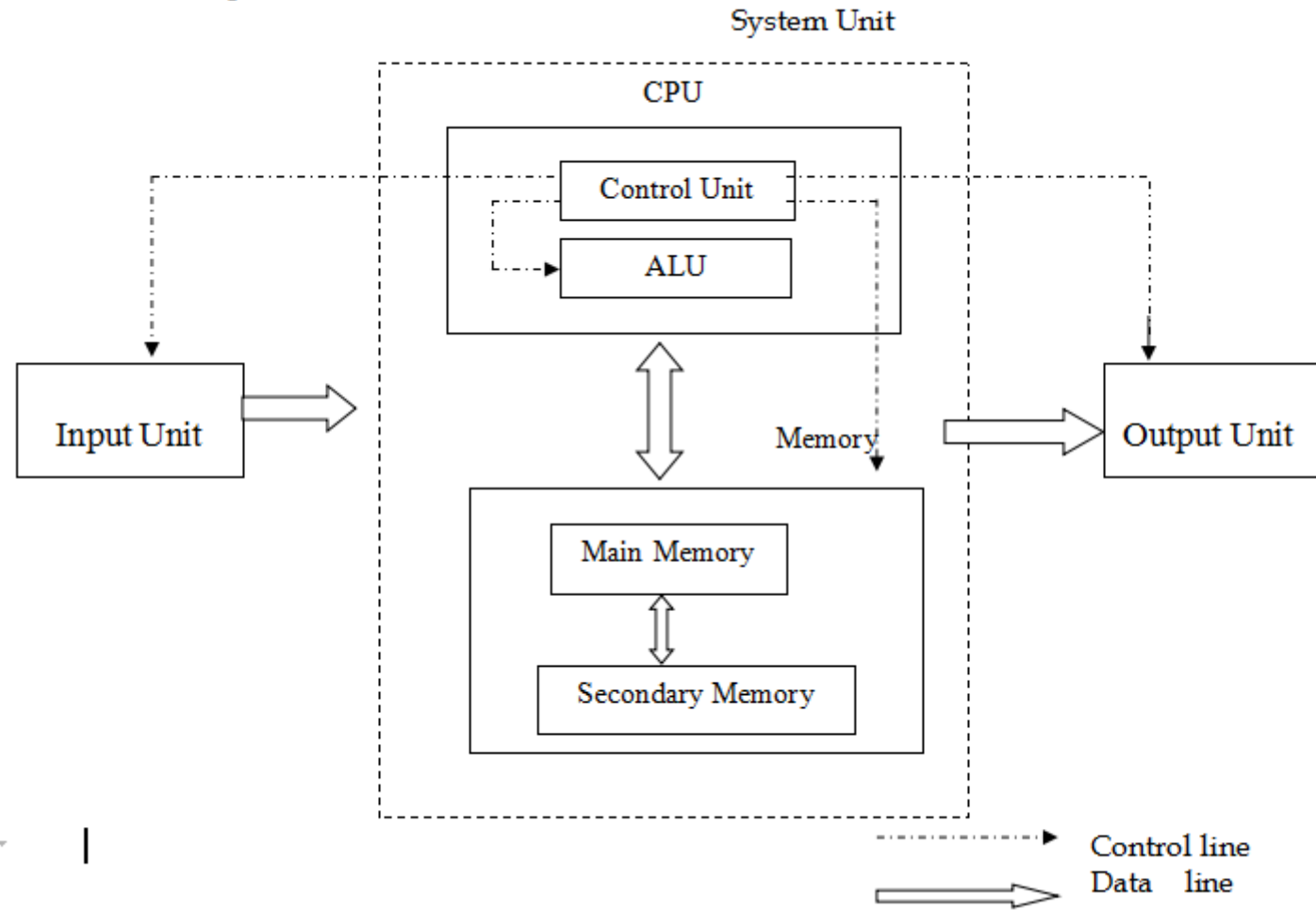
**20) Expert systems:** Expert systems are used to automate the decision-making process in a specific area, such as analyzing the credit histories for loan approval, diagnosing a patient's condition for prescribing an appropriate treatment.



ACHARYA

Dept. of ISE

# Basic Organization of Computers



# Input Unit

- It is an external device that is connected to CPU
- It is used in transferring data/information to the system unit for solving the problems
- The control unit sends the signals to the input devices to receive the data or instructions from the user

Ex: Keyboard, Mouse, Scanner, Light pen, Joystick etc.



# Output Unit

- Output is the process of giving the result of data processing to the outside world (external to the computer system).
- The results are given through output devices such as monitor, and printer.
- Since the computer accepts data only in binary form and the result of processing is also in binary form, the result cannot be directly given to the user.
- The output devices, therefore, convert the results available in binary codes into a human-readable language before displaying it to the user.



MONITOR



PRINTER



SPEAKER



HEADPHONE



PROJECTOR



ACHARYA

Dept. of ISE



# Memory Unit

## PRIMARY MEMORY

Primary memory is temporary.

Primary memory is directly accessible by Processor/CPU.

Nature of Parts of Primary memory varies, RAM- volatile in nature. ROM- Non-volatile.

Primary memory devices are more expensive than secondary storage devices.

The memory devices used for primary memory are semiconductor memories.

Primary memory is also known as Main memory or Internal memory.

**Ex:** RAM, ROM, Cache memory, PROM, EPROM, Registers, etc.

## SECONDARY MEMORY

Secondary memory is permanent.

Secondary memory is not directly accessible by the CPU.

It's always Non-volatile in nature.

Secondary memory devices are less expensive when compared to primary memory devices.

The secondary memory devices are magnetic and optical memories.

Secondary memory is also known as External memory or Auxiliary memory.

**Ex:** Hard Disk, Floppy Disk, Magnetic Tapes, etc.

# Control Unit

- The control unit (CU) is the central nervous system of the entire computer system. It manages and controls all the components of the computer system.
- It is the CU that decides the manner in which instructions will be executed and operations performed. It takes care of the step-by-step processing of all operations that are performed in the computer.
- Note that the CPU is a combination of the arithmetic logic unit (ALU) and the CU. The CPU is better known as the brain of the computer system because the entire processing of data is done in the ALU, and the CU activates and monitors the operations of other units (such as input, output, and storage) of the computer system.



ACHARYA

Dept. of ISE



# Content

- Basic Structure of C program
- Executing a C Program
- C Tokens
- Constant and Variable
- Data types
- Operators and Expressions

# Basic Structure of C Program

```
#include<stdio.h>
main( )
{
    printf("Welcome to CPPS");
}
```

- Any statement that begins with a # is a **pre-processor directive** used for instructing the compiler.
- #include tells the compiler to include standard library function from a header file that will be used in the program.
- stdio.h is the header file for standard input and output. This is useful for getting the input from the user( Keyboard ) and output result to the monitor(screen)





Dept. of ISE

# Program to find area of a Circle

```
#include<stdio.h>

#define PI 3.14

void main( )
{
    int r = 10;
    printf("Area of circle = %f", PI*r*r);
}
```

- **#define** is **preprocessor directive**, It is used to define a value for a symbolic constant in the program. Whenever a symbolic name is encountered, the compiler substitutes the value associated with the name automatically.
- `int r = 10` is used for declaring an integer variable 'r' and storing a value 10 in it.
- `printf` is used for printing the area of the circle

# Program to find area of a circle using functions

```
#include<stdio.h>

#define PI 3.14
float area(int r);

void main( )
{
    int r = 10;
    printf("Area of circle = %f", area( r ) );
}

float area(int r)
{
    return PI*r*r ;
}
```

- In this program, the job of finding area of circle is entrusted to another function named **area**
- **int area(int r);** is used to declare a **user defined function** for finding area of a circle
- After the main function, the area function is defined.
- The function area finds the area of the circle as **PI\*r\*r** and **returns** it to the main function and hence its return type is float



ACHARYA

Dept. of ISE

## Program to find area of a circle using functions and global variables

```
#include<stdio.h>
#define PI 3.14
int r = 10;
float area(int r);

void main( )
{
    printf("Area of circle = %f", area(r));
}

float area( )
{
    return PI*r*r ;
}
```

- In the previous program, the variable 'r' was declared in the main function and was a local variable to the main function. Local variables are accessible only in the function in which it is defined, hence it was sent as parameter for the function area to compute the area of the circle.
- In this program, the variable 'r' is declared outside the main function well before it and is a **global variable**. A global variable is accessible by all the functions of the program and hence it need not be sent as a parameter.

# Basic Structure of a 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)

```
/*Pgm to find area of a circle*/
```

```
#include<stdio.h>
```

```
#define PI 3.14
```

```
int r = 10;
```

```
float area(int r);
```

```
void main( )
```

```
{
```

```
printf("Area of circle = %f", area(r));
```

```
}
```

```
float area( )
```

```
{
```

```
return PI*r*r ;
```

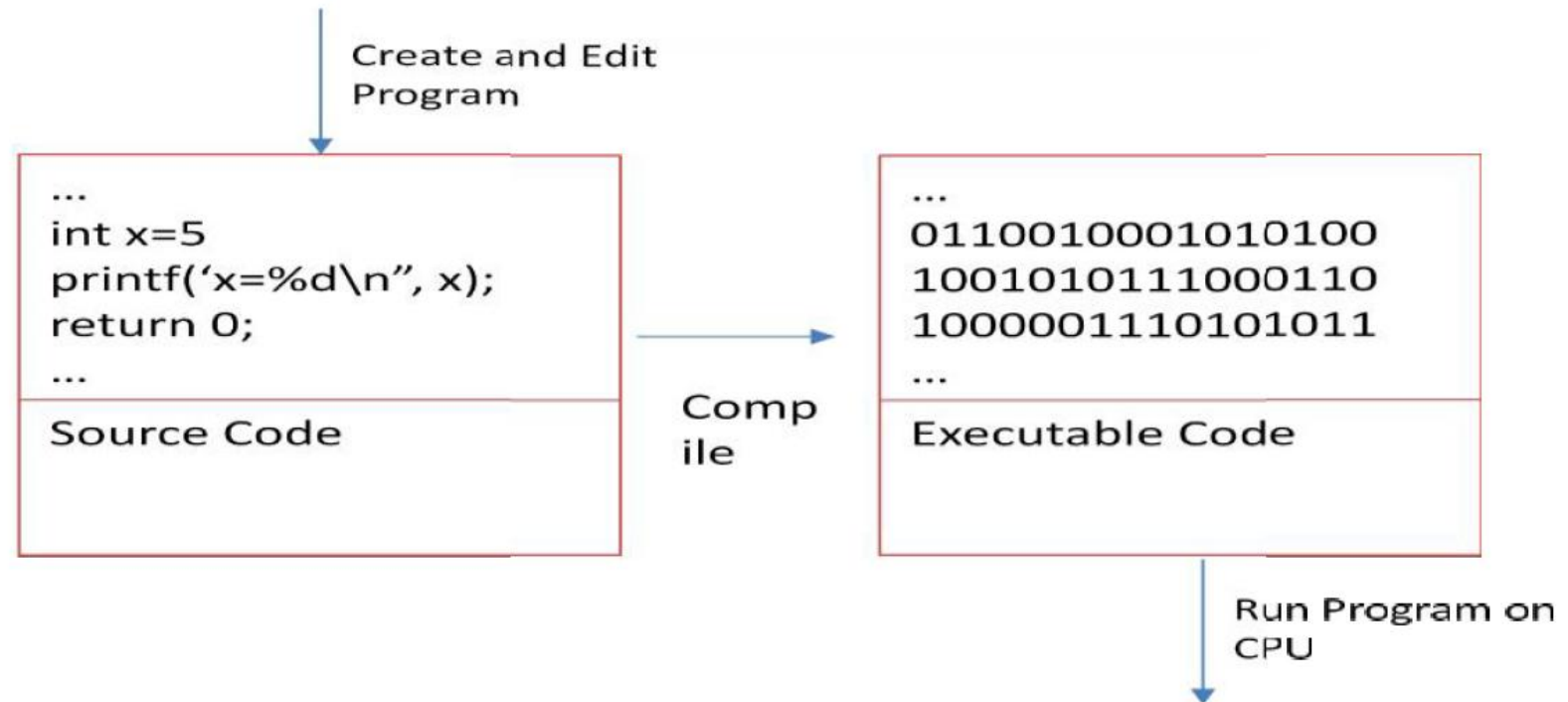
```
}
```

# Source Code v/s Executable Code



ACHARYA

Dept. of ISE

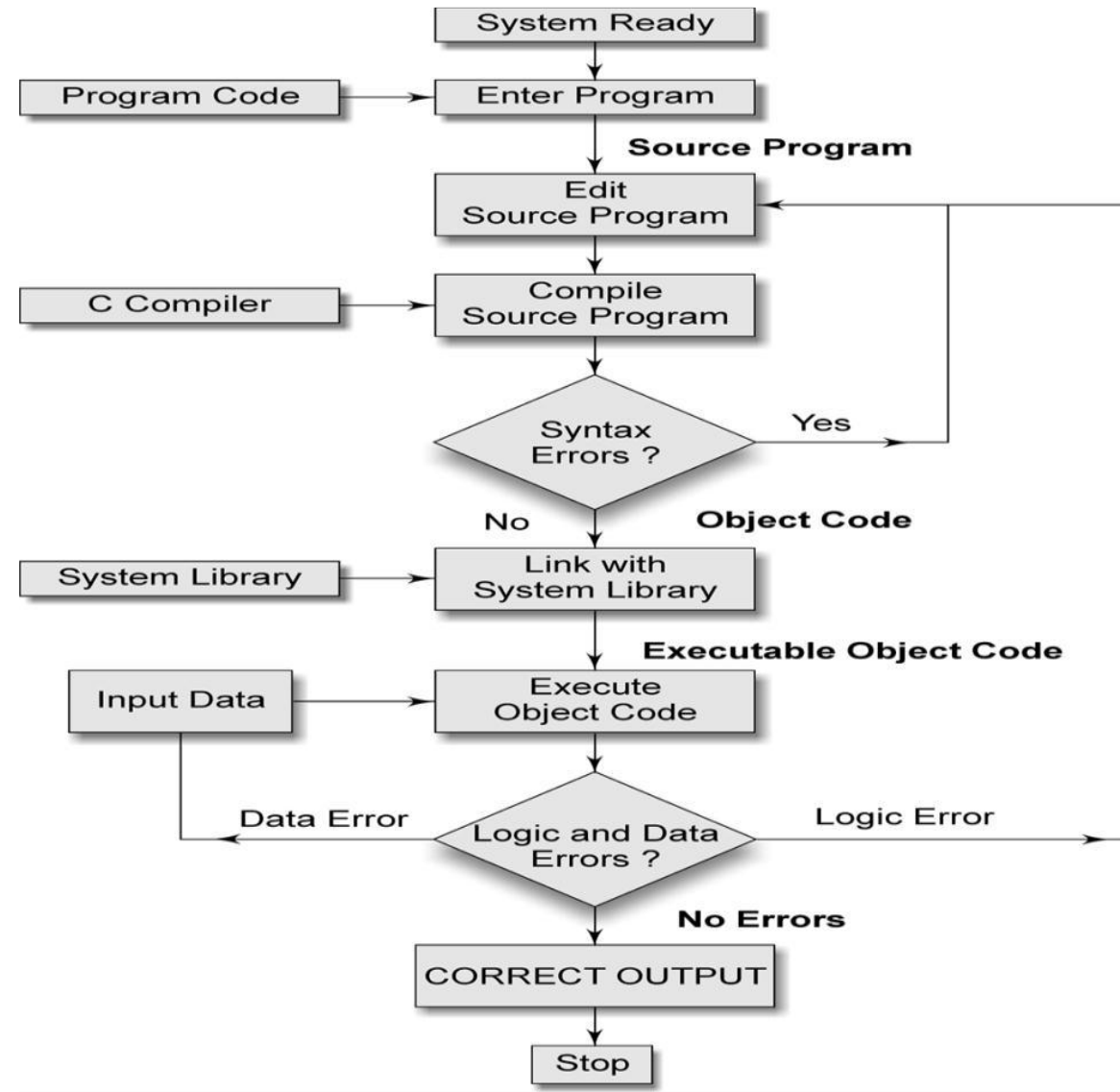




# Content

- Basic Structure of C program
- **Executing a C Program**
- C Tokens
- Constant and Variable
- Data types
- Operators and Expressions

# Execution of C Program



ACHARYA

Dept. of ISE

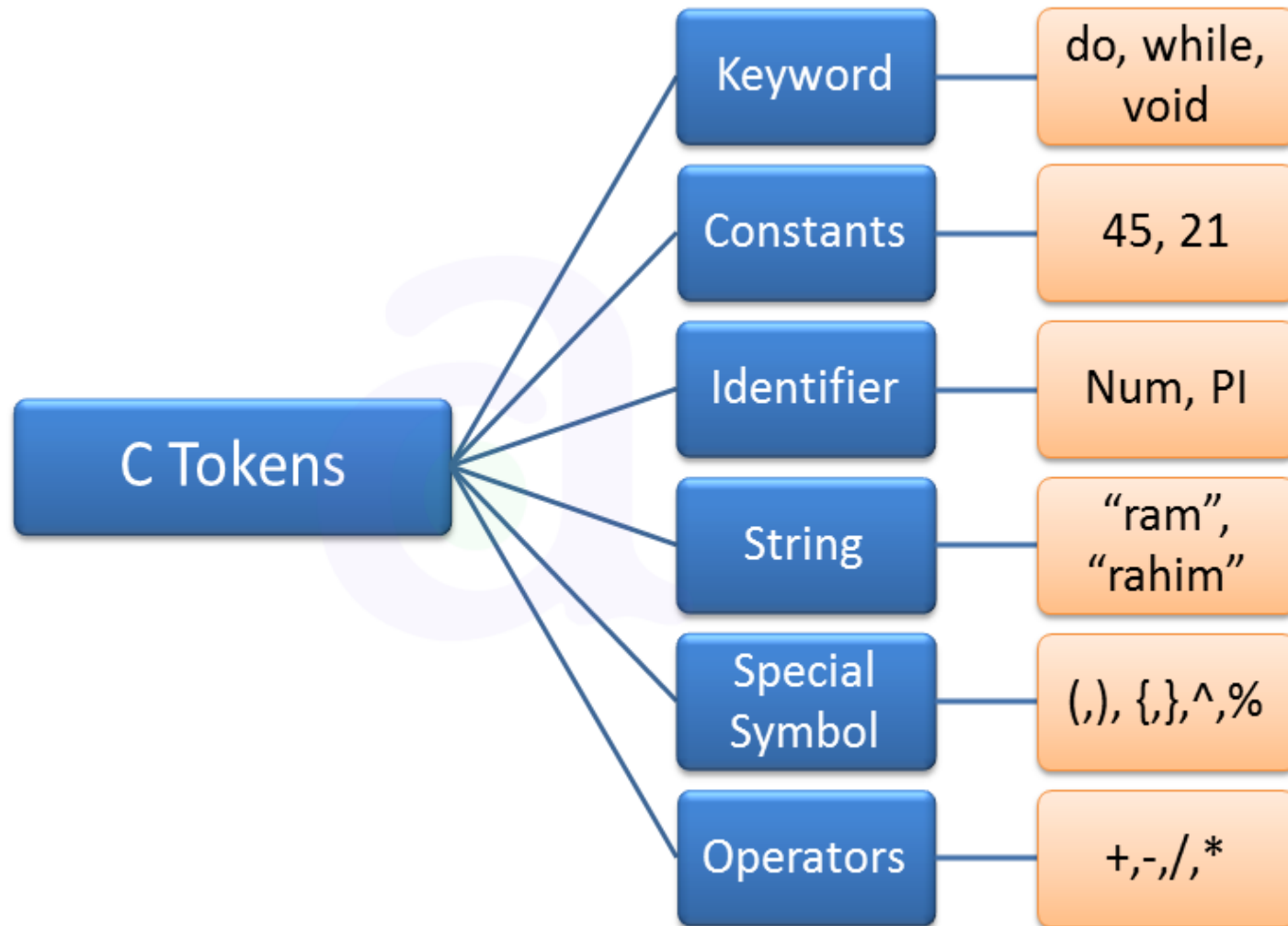


# Content

- Basic Structure of C program
- Executing a C Program
- **C Tokens**
- Constant and Variable
- Data types
- Operators and Expressions

# C Tokens

A token is a smallest or basic unit of any program.



# C Tokens

A token is a smallest or basic unit of any program.

Token	Meaning
<b>Keyword</b>	The words which have predefined meaning in C language are called keywords. <b>Ex: int, float, for, while , if, for ....</b>
<b>Constant</b>	The constants refer to fixed values that the program may not alter during its execution (int, float, char, string values) <b>Ex: pi=3.14, a=5 ....</b>
<b>Identifier</b>	A C identifier is a name used to identify a variable, function, or any other user-defined item. <b>Ex: sum, a, b, area( ), .....</b>
<b>Variables</b>	Variables are simply names used to refer to some location in memory <b>Ex: sum, a, b .....</b>
<b>String</b>	Sequence of characters <b>Ex: "Acharya" .....</b>
<b>Special Symbol</b>	Symbols other than the Alphabets and Digits and white-spaces <b>Ex: @ , { , } , [ , #, .....</b>
<b>Operators</b>	A symbol that represents a specific mathematical or non-mathematical action <b>Ex: +, *, -, /, &amp;&amp;,    , &lt; , =, .....</b>



ACHARYA

Dept. of ISE



# Content

- Basic Structure of C program
- Executing a C Program
- C Tokens
- **Identifier, Constant and Variable**
- Data types
- Operators and Expressions



ACHARYA

Dept. of ISE

# Identifiers

- A C identifier is a name used to identify a **variable**, **function**, or any other **user-defined item**
- An identifier starts with a **letter** A to Z or a to z or an **underscore** \_ followed by zero or more letters, underscores, and **digits (0 to 9)**

Here are some examples of acceptable identifiers:

- ✓ add ( )
- ✓ a
- ✓ \_abc
- ✓ a123

# Constants

- The constants refer to **fixed values** that the program may not alter during its execution
- These fixed values are also called **literals**
- Constants can be of any of the basic data types like an **integer** constant, a **floating** constant, a **character** constant, or a **string** literal.

## ➤ Integer Constant

Decimal(0-9)

:100,-67,989 etc.,

Octal ( 0-7 with a prefix 0)

:010,0777,-065 etc.,

Hexadecimal (0-9 along with A-F)

:0XAB, 0XA123 etc.,

## ➤ Floating point constant

Fractional form

:0.5, -0.99, -6 -9.,+.9

Exponent notation

: 634e-5

## ➤ Character Constant

: '9', 'a', 'm'

## ➤ String constant

: "Acharya" , "ISE"

# Variables

- Variables are simply names used to refer to some **memory location** that holds a value
- Before a C program can utilize memory to store a variable it must claim the memory needed to store the values for a variable
- This is done by **declaring variables**

```
int abc;  
float _abc;  
int abc12;
```

## Rules for defining variables

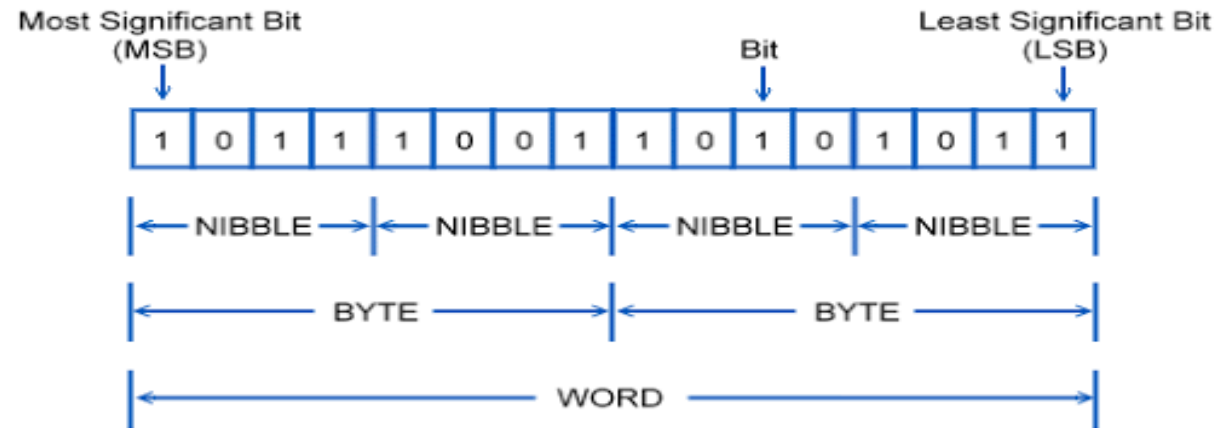
- The first character is the identifier should be letter or '\_'
- No extra symbols are allowed
- The length of an identifier should be within 32 character
- Keywords are not allowed



# Content

- Basic Structure of C program
- Executing a C Program
- C Tokens
- Identifier, Constant and Variable
- **Data types**
- Operators and Expressions

# Memory Measurement Units



SYMBOL	FULL FORM	QUANTITY
1 BIT	BINARY DEGIT	1 CELL , BINARY 0 OR 1
4 BITS	NIBBLE	1/2 BYTE
8 BITS	BYTE	1 BYTE
1024 BYTE	KILOBYTE	1 KILOBYTE
1024 KILOBYTE	MEGABYTE	1 MEGABYTE
1024 MEGABYTE	GIGABYTE	1 GIGABYTE
1024 GIGABYTE	TERABYTE	1 TERABYTE
1024 TERABYTE	PETABYTE	1 PETABYTE
1024 PETABYTE	HEXABYTE	1 HEXABYTE
1024 HEXABYTE	ZEETABYTE	1 ZEETABYTE



ACHARYA

Dept. of ISE

# Basic Data types in C

The data type defines the type of data stored in memory location. It determines how much memory should be allocated for a variable associated with the data type

**All C compilers support 5 primitive data types, namely**

- character (**char**)
- integer (**int**)
- floating point (**float**)
- double precision floating point (**double**)
- **void**

## Data types on a 16-bit Machine

Data Type	Keyword	Memory	Range	Format Specifier for input / output
Character	char	1 byte	$0 \text{ to } 2^8 - 1$ 0 to 255 (unsigned)	%c
Integer	int	2 bytes	$-2^7 \text{ to } +2^7 - 1$ -128 to 127 (signed)	%d
floating point	float	4 bytes	$0 \text{ to } 2^{16} - 1$ 0 to 65535 (unsigned)	%f
Double precision floating point	double	8 bytes	$-2^{15} \text{ to } +2^{15} - 1$ -32768 to 32767 (signed)	%lf

# Variables

- A variable is a name given to the memory location where the data can be stored. Using this name the data can be further accessed or manipulated easily
- The following is the general syntax for declaring variables  
**data type** var<sub>1</sub>, var<sub>2</sub>, var<sub>3</sub>,....., var<sub>n</sub>;

## Example:

```
int a, b, c;      // declaration of 3 integer variables
float x, y;       // declaration of 2 floating point variables
```

- With this declaration, memory for 3 integers and 2 floating point variables will be allocated by the compiler. The memory map is as shown below

a		b		c		x				y			
1000	1001	1002	1003	1004	1005	1006	1007	1008	1009	1010	1011	1012	1013
GV		GV		GV		GV				GV			





# Variables

- Declaration of variables will only allocate memory, however, the contents inside the memory will be unknown values (also referred to as garbage values).
- Values to variables can be assigned using assignment operator =
- Example: After the above declaration, values can be assigned like  
a = 20;  
b = 100;  
x = 3.14;

With this initialization, the memory map is as shown below.

a		b		c		x				y			
1000	1001	1002	1003	1004	1005	1006	1007	1008	1009	1010	1011	1012	1013
20		100		GV		3.14				GV			



# Content

- Basic Structure of C program
- Executing a C Program
- C Tokens
- Identifier, Constant and Variable
- Data types
- **Input and Output function in C**
- Operators and Expressions



Dept. of ISE

# Input and Output Functions in C

- Input functions accept the data from input devices (keyboard) and store them in the memory locations.

**Example:** scanf(), getchar(), gets() etc.

- Output functions retrieve the data from memory locations and send them to output devices (monitor, printer).

**Example:** printf(), putchar(), puts() etc.

## Formatted I/O Functions

### 1) printf( ) function

printf( ) can be used to print only text or text with values

### 2) To output only text

Write the text within double quotes inside printf.

EX: printf("Hello 2024 Batch AE");

# Output Functions in C

- **To output values**

- Write text (optional) along with format specifiers within double quotes, followed by a comma, followed by the names of variables separated by comma inside printf.

- The general syntax of printf ( ) is

printf("control string", var1, var2, var3,.....,varN);

**EX: Consider the following declarations,**

int a = 10;

int b = 20;

float c = 3.14;

float d = 7.25;



ACHARYA

Dept. of ISE



# Output Functions in C

**EX: Consider the following declarations,**

```
int a = 10;  
int b = 20;  
float c = 3.14;  
float d = 7.25;
```

<code>printf("%d %d", a, b);</code>	10 20
<code>printf("The value of a is %d", a);</code>	The value of a is 10
<code>printf("The value of a is %d and b is %d", a, b);</code>	The value of a is 10 and b is 20
<code>printf("%d %d %f %f", a, b, c, d);</code>	10 20 3.14 7.25
<code>printf("%f %d %f %d", c, a, d, b);</code>	3.14 10 7.25 20
<code>printf("The value of c is %f and a is %d", c, a);</code>	The value of c is 3.14 and a is 10

# Input Functions in C

## scanf( ) function

- To read values
  - Write the format specifiers within double quotes, followed by a comma, followed by the address of variables separated by comma inside scanf

EX: `scanf("%d %d %f", &a, &b, &c);`

- The general syntax of scanf ( ) is

`scanf("control string", &var1, &var2, &var3,.....,&varN);`



ACHARYA

Dept. of ISE



ACHARYA

Dept. of ISE

# Input Functions in C

**EX: Consider the following declarations,**

```
int a;
```

```
int b;
```

```
float c;
```

```
float d;
```

To read a	<code>scanf("%d", &amp;a);</code>
To read a and b	<code>scanf("%d%d",&amp;a,&amp;b);</code>
To read b and c	<code>scanf("%d%f",&amp;b,&amp;c);</code>
To read c followed by a followed by d followed by b	<code>scanf("%f%d%f%d",&amp;c,&amp;a,&amp;d,&amp;b);</code>



## Simple Programs

1. Write a C program to find the sum of two numbers
2. Write a C program to find the area of a rectangle
3. Write a C program to find Simple Interest
4. Write a C program to swap two numbers

# Algorithms

Algorithm is a step by step procedure to solve a given problem in a finite number of steps by accepting a set of inputs and producing the desired output.

The **criteria's** or properties that an algorithm must be satisfy are

1. **Input:** Zero or more quantities are externally supplied.
2. **Output:** At least one quantity is produced.
3. **Definiteness:** Each instruction must be clear and unambiguous.
4. **Finiteness:** The algorithm always terminates after a finite number of steps and uses finite sources.

## Steps to write an algorithm

- 1) The name of the algorithm must be written first
- 2) Each instruction must have a step number
- 3) Each step must be specified with sufficient amount of explanation within [ ] before writing the instruction.
- 4) The end of the algorithm must be specified as the the last



## Write an algorithm to find the sum of two numbers

Algorithm : Addition of two numbers

Step 1: [ Input the value of a and b]

Read a, b

Step 2: [Compute the addition of a and b]

$\text{sum} = a + b$

Step 3: [Output the value of sum]

print the addition of a and b is sum

Step 4: [Finished] Stop

## Write an algorithm to find the area of a circle

Algorithm : Area of Circle

Step 1: [ Input the value of radius r]

Read r

Step 2: [Compute the area of circle]

$\text{area} = 3.14 * r * r$

Step 3: [Output the area of circle]

print the area

Step 4: [Finished] Stop

## Write an algorithm to find the area of a rectangle

Algorithm : Area of Rectangle

Step 1: [ Input the value of length l and breadth b]

Read l, b

Step 2: [Compute the area of rectangle]

$\text{area} = l * b$

Step 3: [Output the area of rectangle]

print the area

Step 4: [Finished] Stop

# Write an algorithm to find the Simple Interest

Algorithm :Simple Interest

Step 1: [ Input the value of P, T and R]

Read P, T and R

Step 2: [Compute the Simple Interest]

$SI = P * T * R / 100;$

Step 3: [Output the Simple Interest]

print SI

Step 4: [Finished] Stop

## Write an algorithm to swap or exchange 2 numbers

Algorithm : Swap 2 numbers

Step 1: [ Input the value of a and b]

Read a, b

Step 2: [Print the values of a and b before swapping]

print a, b

Step 3: [Exchange the values of a and b using a temp variable]

temp = a

a = b

b = temp

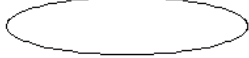


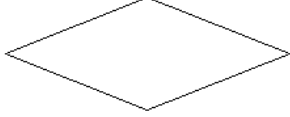



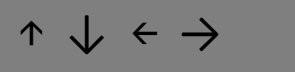
Step 4: [Print the values of a and b after swapping]

print a, b

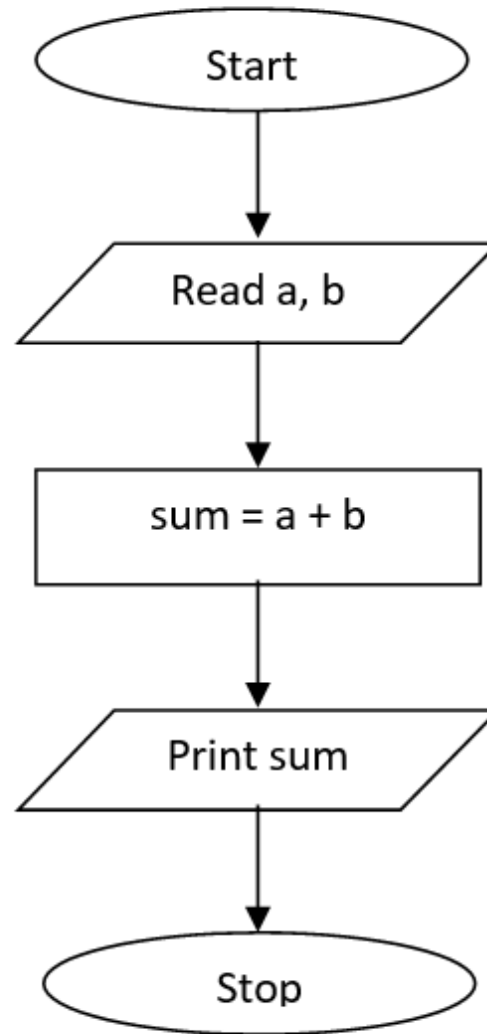
Step 5: [Finished] Stop

# Flowchart

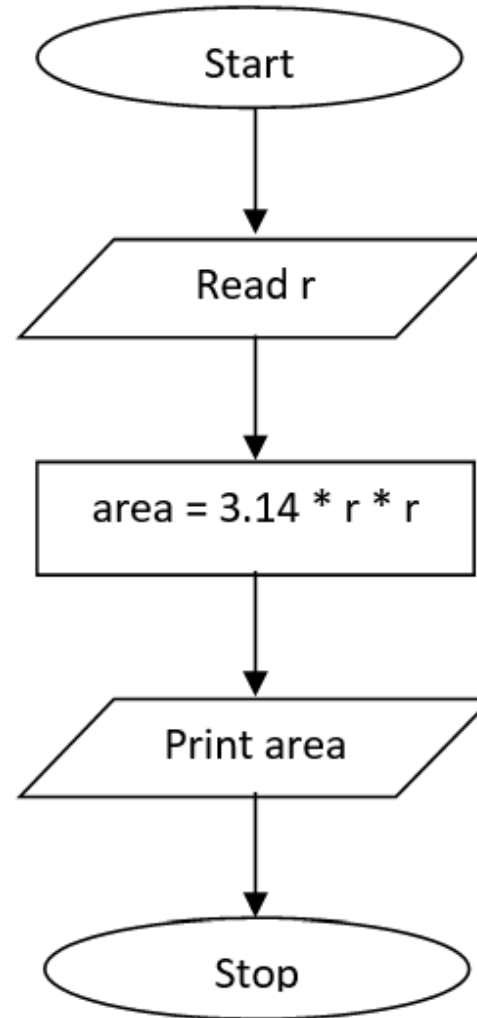
It is a pictorial representation of an algorithm.

Action or Symbol Name	Geometric Shape	Description
Start and Stop Oval		Denotes start or end of algorithm
Input and Output Parallelogram		Input & output statements must be written inside
Processing or Computation Rectangle		All computation and processing statements must be written in this.
Decision Rhombus		All comparisons or decision making statements must be written in this.
Connector Circle		Used to show the continuity in next page
Repetition or Loop Hexagon		Repetitive statements (group of instructions need to be executed a number of times) must be written in this
Predefined Process or function Rectangle with double struck vertical edges		User defined functions are written in this
Control flow Arrows		Arrows are used to show the flow of control

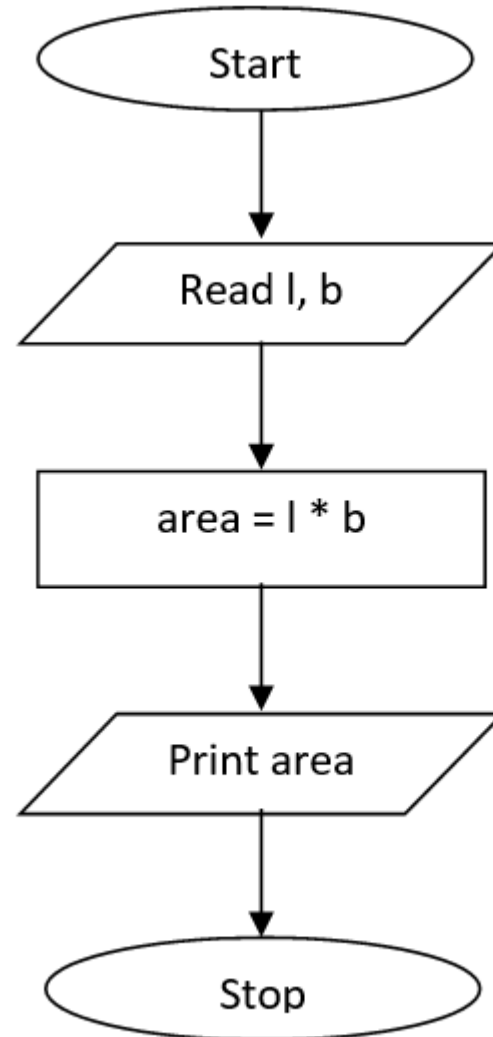
## Write a Flowchar to find the sum of two numbers



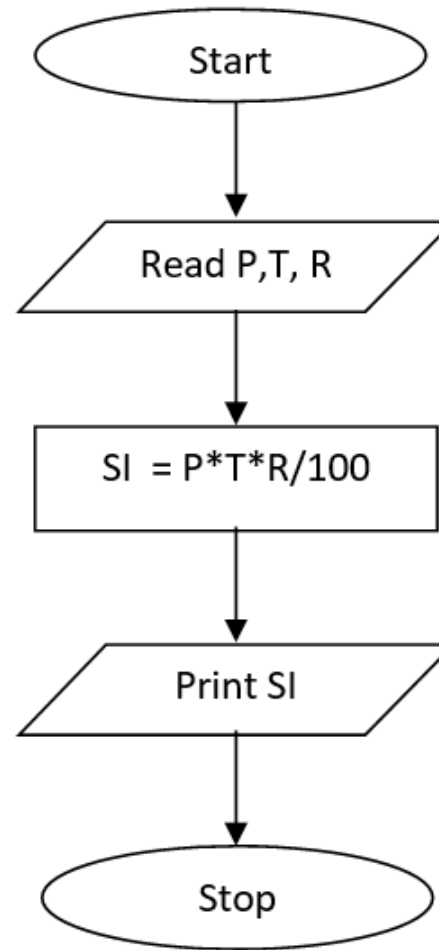
## Write a flowchar to find the area of a circle



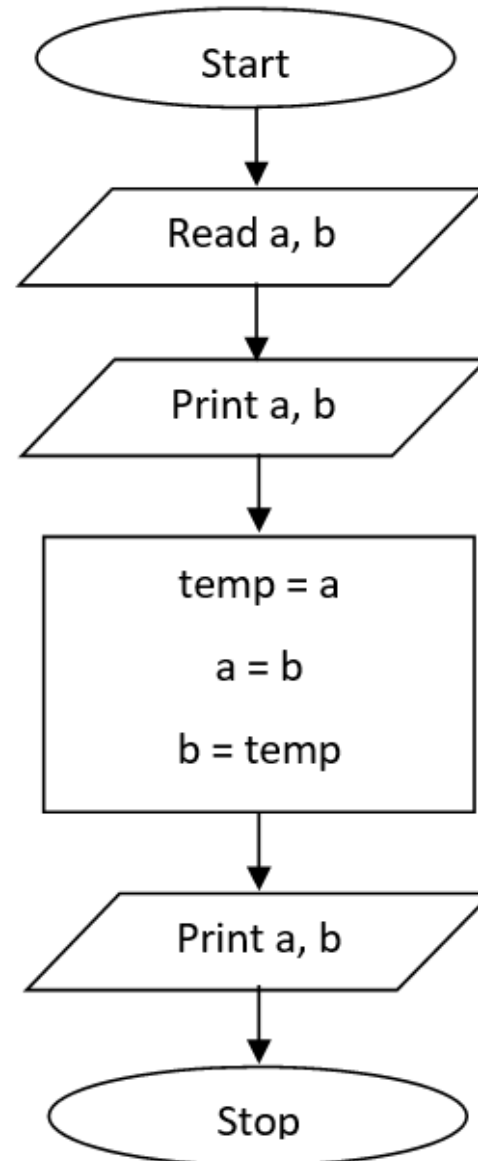
## Write a flowchar to find the area of a rectangle



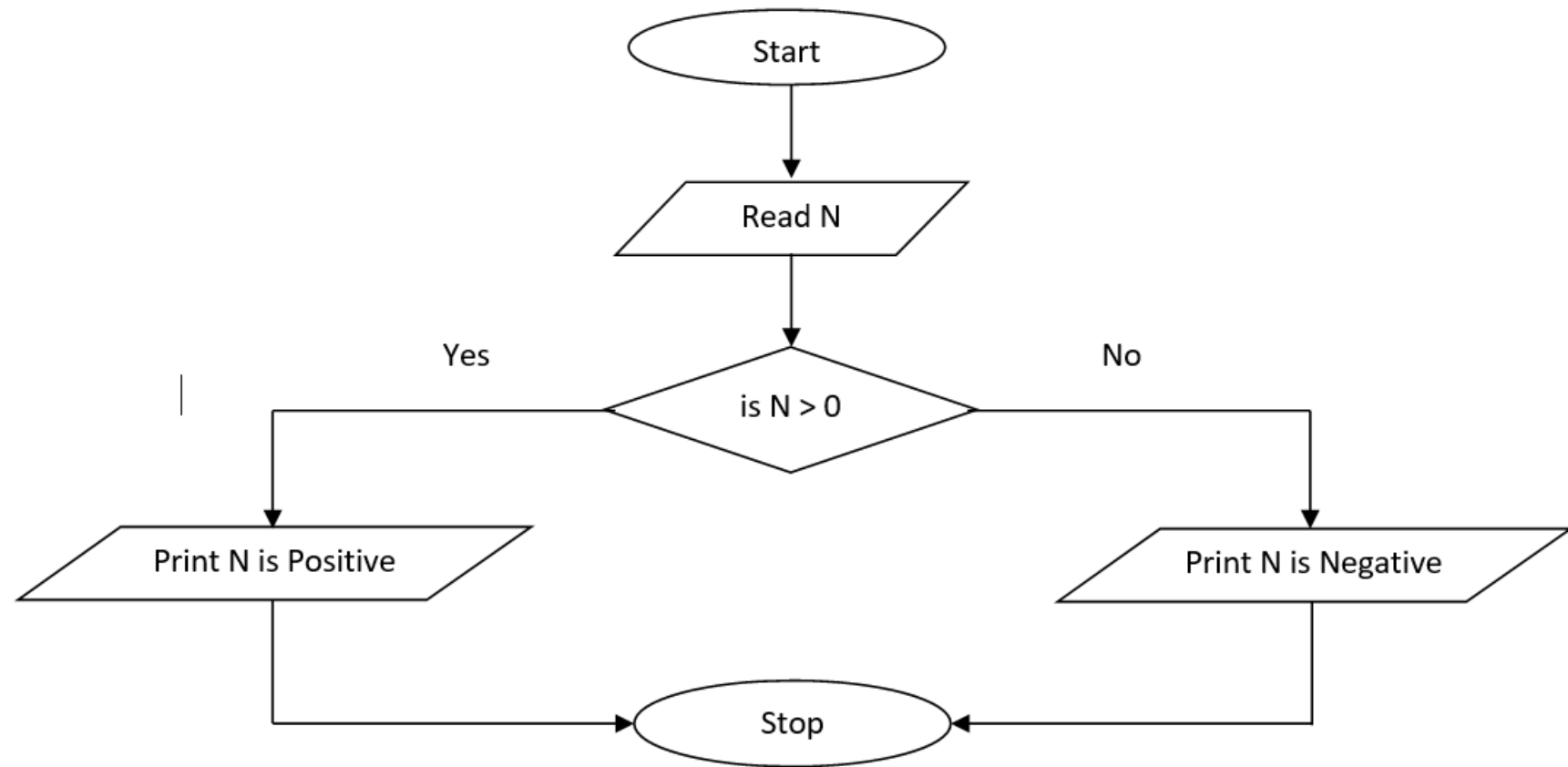
## Write a flowchar to find the Simple Interest



## Write a flowchar to swap or exchange 2 numbers



# Write a flowchart to find whether a given number is positive or negative



# TYPES OF ERRORS

While writing programs, very often we get errors in our program. These errors if not removed will either give erroneous output or will not let the compiler to compile the program.

Run-time Errors occur when the program is being run executed. Such errors occur when the program performs some illegal operation like

- Dividing a number by zero
- Opening a file that already exists
- Lack of free memory space
- Finding square or logarithm of negative numbers

Run-time errors may terminate program execution, so the code must be written in such a way that it handles all sorts of unexpected errors rather terminating it unexpectedly. This ability to continue operation of a program despite of run-time errors is called robustness.



Dept. of ISE



Dept. of ISE

1. **Compile-time Errors** occur at the time of compilation of the program. Such errors can be further classified as follows:
2. **Syntax Errors** Syntax error is generated when rules of C programming language are violated. For example, if we write `int a:` then a syntax error will occur since the correct statement should be `int a;`
3. **Semantic Errors** Semantic errors are those errors which may comply with rules of the programming language but are not meaningful to the compiler. For example, if we write, `a * b = c;` it does not seem correct. Rather, if written like `c = a * b` would have been more meaningful.



Dept. of ISE

**4. Logical Errors** are errors in the program code that result in unexpected and undesirable output which is obviously not correct.

Such errors are not detected by the compiler, and programmers must check their code line by line or use a debugger to locate and rectify the errors.

Logical errors occur due to incorrect statements. For example, if you meant to perform  $c = a + b$ ; and by mistake you typed  $c = a * b$ ; then though this statement is syntactically correct it is logically wrong.

**5. Linker Errors** occur when the linker is not able to find the function definition for a given prototype. For example, if you write `clrscr()`; but do not include `conio.h` then a linker error will be shown.

# Testing Approaches

*Testing is an activity that is performed to verify correct behavior of a program. It is specifically carried out with an intent to find errors.*

1. Unit testing is applied only on a single unit or module to ensure whether it exhibits the expected behavior.
2. Integration Tests are a logical extension of unit tests. In this test, two units that have already been tested are combined into a component and the interface between them is tested. This process is repeated until all the modules are tested together. The main focus of integration testing is to identify errors that occur when the units are combined.
3. System testing checks the entire system. For example, if our program code consists of three modules then each of the module is tested individually using unit tests and then system test is applied to test this entire system as one system.



Dept. of ISE

# Testing Approaches

*Testing is an activity that is performed to verify correct behavior of a program. It is specifically carried out with an intent to find errors.*

1. Unit testing is applied only on a single unit or module to ensure whether it exhibits the expected behavior.
2. Integration Tests are a logical extension of unit tests. In this test, two units that have already been tested are combined into a component and the interface between them is tested. This process is repeated until all the modules are tested together. The main focus of integration testing is to identify errors that occur when the units are combined.
3. System testing checks the entire system. For example, if our program code consists of three modules then each of the module is tested individually using unit tests and then system test is applied to test this entire system as one system.



ACHARYA

Dept. of ISE



Dept. of ISE

**Problem statement :** To develop an automatic system that accepts marks of a student and generate his/her grade.

**Requirement Analysis:** Ask the users to enlist the rules for assigning grades.

```
#include<stdio.h>
{
    int marks;
    char grade;

    printf("\n Enter the marks of the student:");
    scanf("%d",&marks);
    if(marks<0 || marks>100)
    {
        printf("\n Not Possible");
        exit(1);
    }
    if(marks>=75)
        grade = 'O';
    else if(marks>=60 && marks<75)
        grade = 'A';
    else if(marks>=50 && marks<60)
        garde = 'B';
    else if(marks>=40 && marks<50)
        grade = 'C';
    else
        garde = 'D';
    printf("\n GRADE = %c",grade);
}
```

# RESULT ANALYSIS

Test CASE ID	Input	Expected Output	Actual Output
1	-12	Not possible	Not possible
2	112	Not Possible	Not Possible
3	32	D	D
4	46	C	C
5	54	B	B
6	68	A	A
7	91	O	O
8	40	C	C
9	50	B	B
10	60	A	A
11	75	O	O
12	100	O	O
13	0	D	D



Dept. of ISE



Dept. of ISE

# Thank you

