Parul University

Fundamentals of Programming Using C - 15101104

Overview of C

Bhavika Vaghela

Assistant Professor

Parul Institute of Computer Application

Faculty of IT & Computer Science

Parul University

Before Staring About C Programming..

Let's understand some basic of computer programming

- What is Program?

A computer program is a collection of instructions that performs a specific task when executed by a computer.

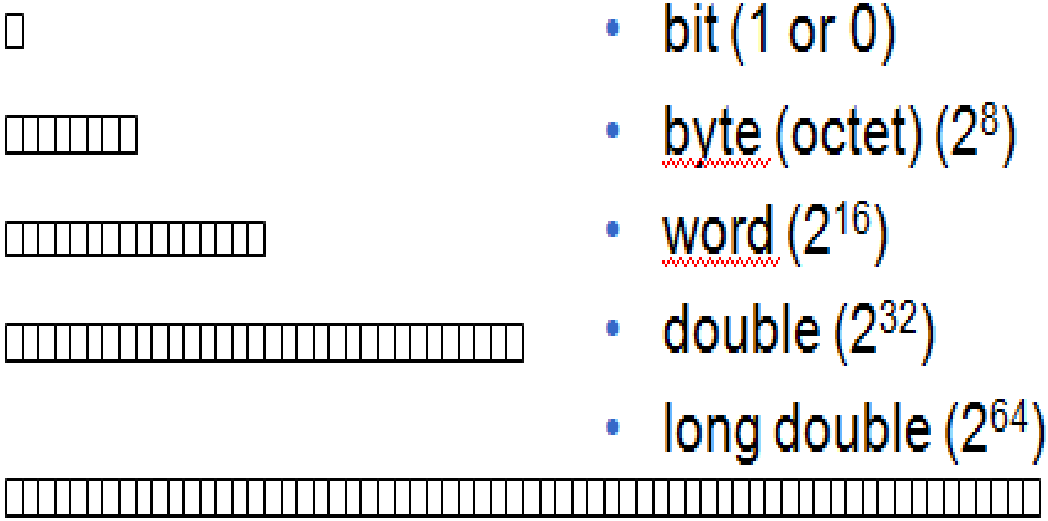
- What Any Computer understand?

It only understand Machine level language which is only combination of 0's and 1's.

- In early days it is written using Machine level language.
- So the C programming is known as one of the high level programming language.

How Computer store information

- 1 bit
- 8 bits
- 16 bits
- 32 bits
- 64 bits



- bit (1 or 0)
- byte (octet) (2^8)
- word (2^{16})
- double (2^{32})
- long double (2^{64})

Levels of Programming Language

High Level Language

```
class Triangle {
  ...
  float surface()
    return b*h/2;
  }
```

Low-level program

```
LOAD r1,b
LOAD r2,h
MUL r1,r2
DIV r1,#2
RET
```

Executable machine code

```
00010010010001010010010011
1011001010110100 1...
```

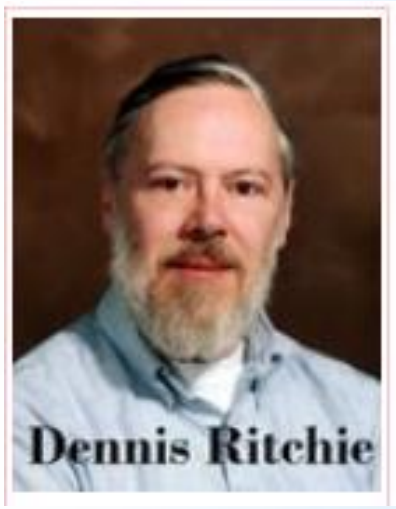
Types of Programming Language

Cont…

- Machine and assembly languages: BINARY, ASSEMBLY
- Algorithmic languages: FORTRAN, ALGOL, LISP, C
- Business Oriented: COBOL, SQL
- Education Oriented: BASIC, PASCAL, LOGO, HYPERTALK
- Object Oriented: C++, ADA, JAVA, VISUAL BASIC
- Declarative: PROLOG
- Scripting Languages: PERL, UNIX
- Web languages: HTML, XML, PHP, CSS, JAVASCRIPT, .NET

History

- C programming language was developed in 1972 by Dennis Ritchie at bell laboratories of AT&T (American Telephone & Telegraph), located in the U.S.A.
- Dennis Ritchie is known as the founder of the c language.



- It was developed to overcome the problems of previous languages such as B, BCPL, etc.
- Initially, C language was developed to be used in UNIX operating system. It inherits many features of previous languages such as B and BCPL.

History

Summary –		
1	B Language Developed By	Ken Thompson
2	Operating System Developed in C	UNIX
3	Developed at	AT & T Bell Laboratory
4	Creator of Traditional C	Dennis Ritchie
5	Year	**1972**

C Programming Language Timeline :

Programming Language	Development Year	Developed by
ALGOL	1960	International Group
BCPL	1967	Martin Richards
B	1970	Ken Thompson
Traditional C	1972	Dennis Ritchie
K&R C	1978	Brain Kernighan and Dennis Ritchie
ANSI C	1989	ANSI Committee
ANSI/ISO C	1990	ISO Committee

Algorithm and Flow Chart

- Algorithm is a procedure or formula or one can also say its step by step procedure to solve any given problem.
- An algorithm (pronounced AL-go-rith-um) is a procedure or formula for solving a problem.
- The word derives from the name of the mathematician, Mohammed ibn-Musa al-Khwarizmi, who was part of the royal court in Baghdad
- It is step by step in sequence to solve problem.
- We can also say an algorithm is a finite set of instruction to solve the problem.
- Algorithms have a definite beginning and a definite end, and a finite number of steps.
- Proper algorithm is very important in efficiency of programming.

Algorithm Cont…

- We use algorithm every day in day to day life style
- Lets see some example of it.

Write an algorithm for add two numbers

Step1 : Start
Step2 : Take input of two number (read num1 and num2)
Step3 : Declare one more variable named as sum
Step4 : Perform sum= num1 + num2 and store it into variable sum
Step5 : Print answer (variable sum)
Step6 : Stop

**** using same concept write algorithm for multiplication of two numbers**

Algorithm Cont…

Write an algorithm to find entered number is odd or even.

Step1 : Start
Step2 : declare one variable
Step3: assign some value to declared variable
Step4: divide that number with 2
Step5: check condition
Step6: if modulo is 0 than its even number else odd number
Step7: print result
Step8: stop

** Based on above write down algorithm to find number is positive or negative.

Algorithm Cont…

Write an algorithm to find entered number is odd or even.

Step 1: Start
Step 2: Read Number n
Step 3: Check if (n % 2 == 0) then go to step 5
Step 4: Display n is Odd Number go to step 6
Step 5: Display n is Even Number
Step 6: Stop

Algorithm cont…

Algorithm to swap value of two numbers

Step 1: Start
Step 2: declare three variable a, b and temp
Step 3: Read 'a' and 'b' value
Step 4: Interchange the values
Step 5: (Create Temporary Variable) temp = a
Step 6: a = b
Step 7: b = temp
Step 8: Display ‘a’ and ‘b’ values.
Step 9: Stop

Algorithm cont…

Algorithm to swap value of two numbers without using temp

Step 1: Start
Step 2: declare two variable a, b
Step 3: Read 'a' and 'b' value
Step 4: Interchange the values
Step 4: a = a + b
Step 5: b= a - b
Step 6: a = a - b
Step 8: Display ‘a’ and ‘b’ values.
Step 9: Stop

**** like can you able to write algorithm for the same using * and / operator?**

Flowchart

- Same as algorithm flowchart is pictorial representation of step by step procedure to solve out problem.
- A flowchart is a type of diagram that represents an algorithm, workflow or process, showing the steps as boxes of various kinds, and their order by connecting them with arrows
- Flowcharts are an important tool across various industries and careers, since they're a clear, concise method of displaying information
- The main purpose of a flowchart is to analyze different processes.
- The purpose of a flow chart is to provide people with a common language or reference point when dealing with a project or process.
- Flowcharts use simple geometric symbols and arrows to define relationships

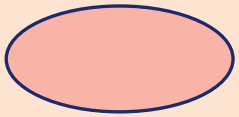
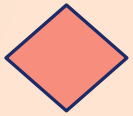
Why to use Flowchart or Advantages

- **Communication**: Flowcharts are better way of communicating the logic of a system to all concerned or involved.
- **Effective analysis**: With the help of flowchart, problem can be analyzed in more effective way therefore reducing cost and wastage of time.
- **Proper documentation**: Program flowcharts serve as a good program documentation, which is needed for various purposes, making things more efficient.
- **Efficient Coding**: The flowcharts act as a guide or blueprint during the systems analysis and program development phase.
- **Proper Debugging**: The flowchart helps in debugging process.
- **Efficient Program Maintenance**: The maintenance of operating program becomes easy with the help of flowchart. It helps the programmer to put efforts more efficiently on that part

Disadvantage of Flowchart

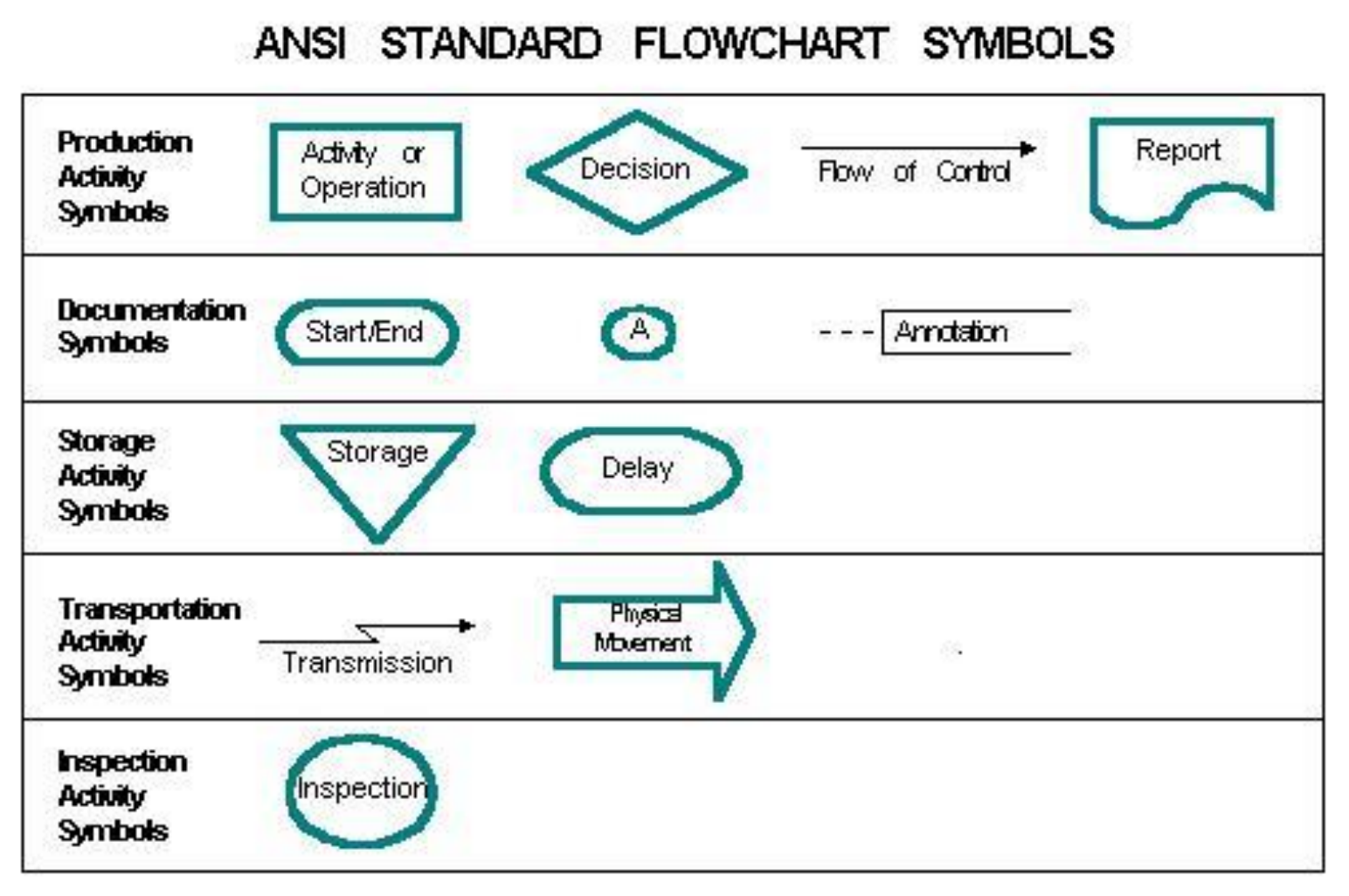
- **Complex logic:** Sometimes, the program logic is quite complicated. In that case, flowchart becomes complex and clumsy. This will become a pain for the user, resulting in a waste of time and money trying to correct the problem
- **Alterations and Modifications:** If alterations are required the flowchart may require re-drawing completely. This will usually waste valuable time.
- **Reproduction**: As the flowchart symbols cannot be typed, reproduction of flowchart becomes a problem.

Symbols of Flowchart

Name	Shape	Purpose	Description
Oval/ Terminator		Start/ End	It use to show staring and end of flowchart.
Rectangle		Process	Is use to show some calculation or process
Diamond		Condition	Is use to check condition.
Parallelogram		Input / output (data)	Is use to take some input from user
Arrow		Connector	It use to connect flow and show relationship between process flow

Flowchart cont..

More Symbols of Flowchart



Flowchart cont…

Terminator
Indicates the beginning or end of a program flow in your diagram.

Process
Indicates any processing function.

Decision
Indicates a decision point between two or more paths in a flowchart.

Delay
Indicates a delay in the process.

Data
Can represents any type of data in a flowchart.

Document
Indicates data that can be read by people, such as printed output.

Multiple documents
Indicates multiple documents.

Subroutine
Indicates a predefined (named) process, such as a subroutine or a module.

Preparation
Indicates a modification to a process, such as setting a switch or initializing a routine.

Display
Indicates data that is displayed for people to read, such as data on a monitor or projector screen.

Manual input
Indicates any operation that is performed manually (by a person).

Manual loop
Indicates a sequence of commands that will continue to repeat until stopped manually.

Loop limit
Indicates the start of a loop. Flip the shape vertically to indicate the end of a loop.

Stored data
Indicates any type of stored data.

Connector
Indicates an inspection point.

Off-page connector
Use this shape to create a cross-reference and hyperlink from a process on one page to a process on another page.

Off-page connector

Off-page connector

Off-page connector

Or
Logical OR

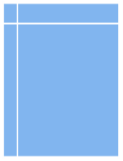
Summing junction
Logical AND

Collate
Indicates a step that organizes data into a standard format.

Sort
Indicates a step that organizes items list sequentially.

Merge
Indicates a step that combines multiple sets into one.

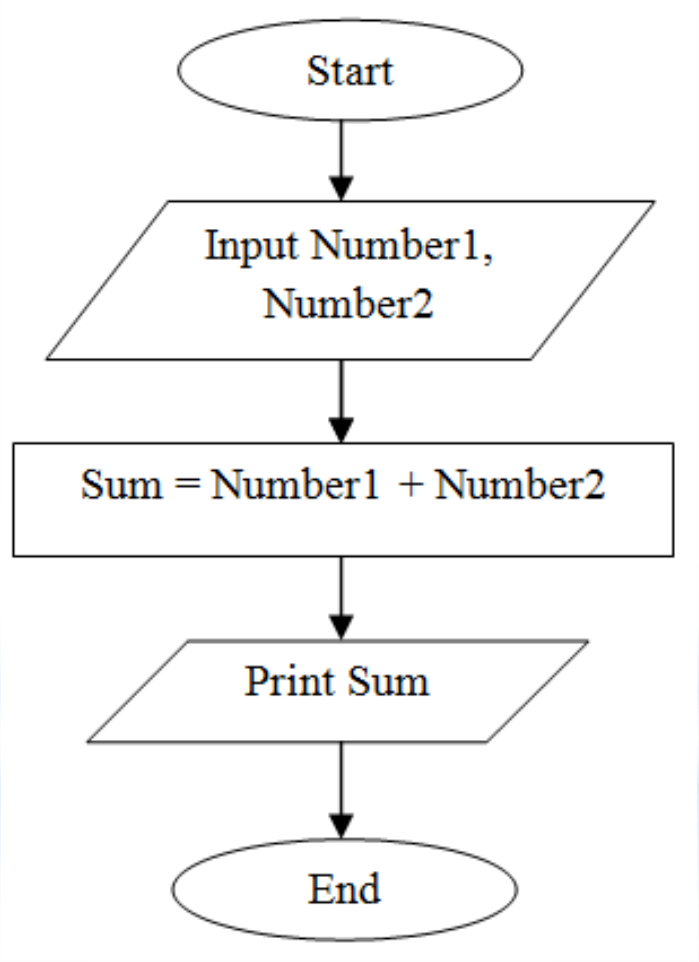
Database
Indicates a list of information with a standard structure that allows for searching and sorting.



Internal storage
Indicates an internal storage device.

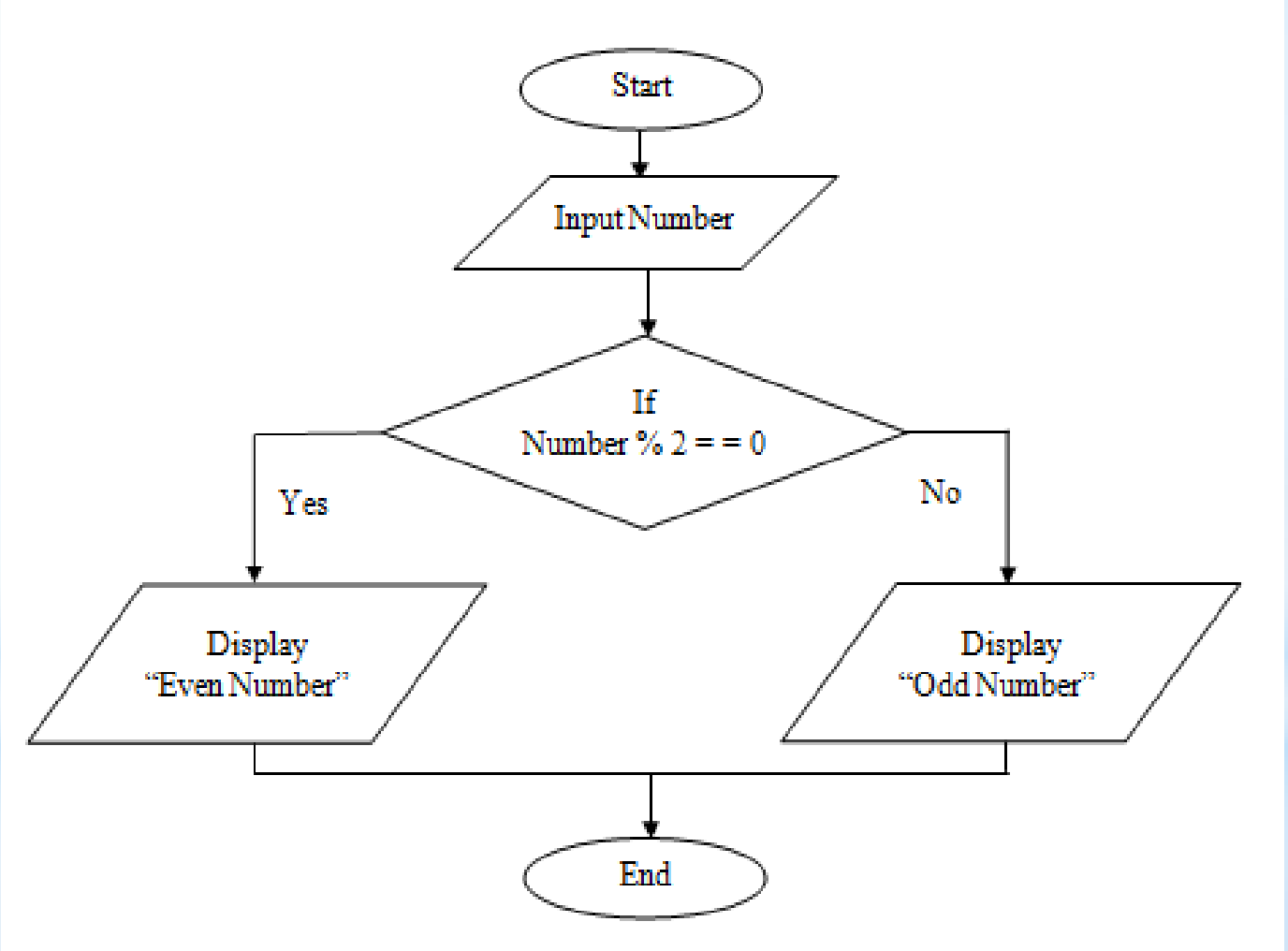
Flowchart cont..

Flowchart for adding two number



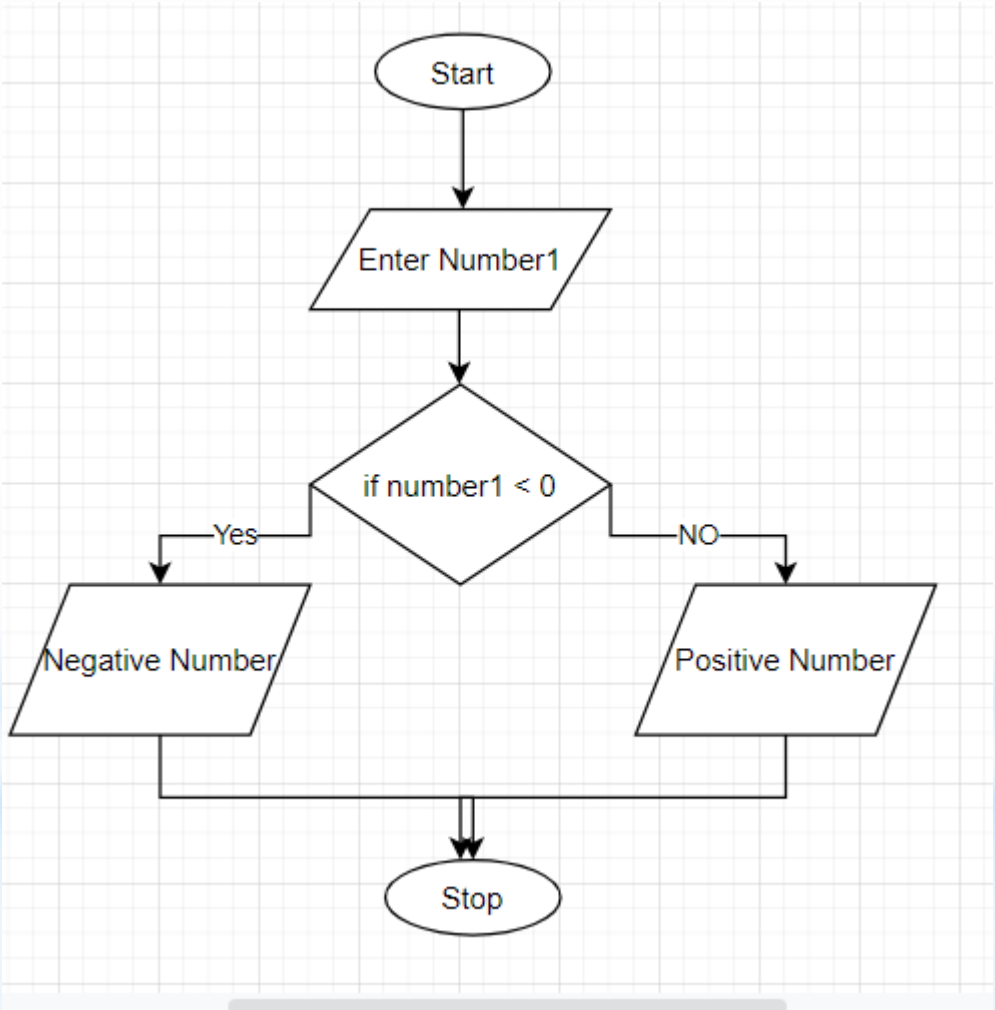
Flowchart cont…

Flowchart to check Number is Odd or Even



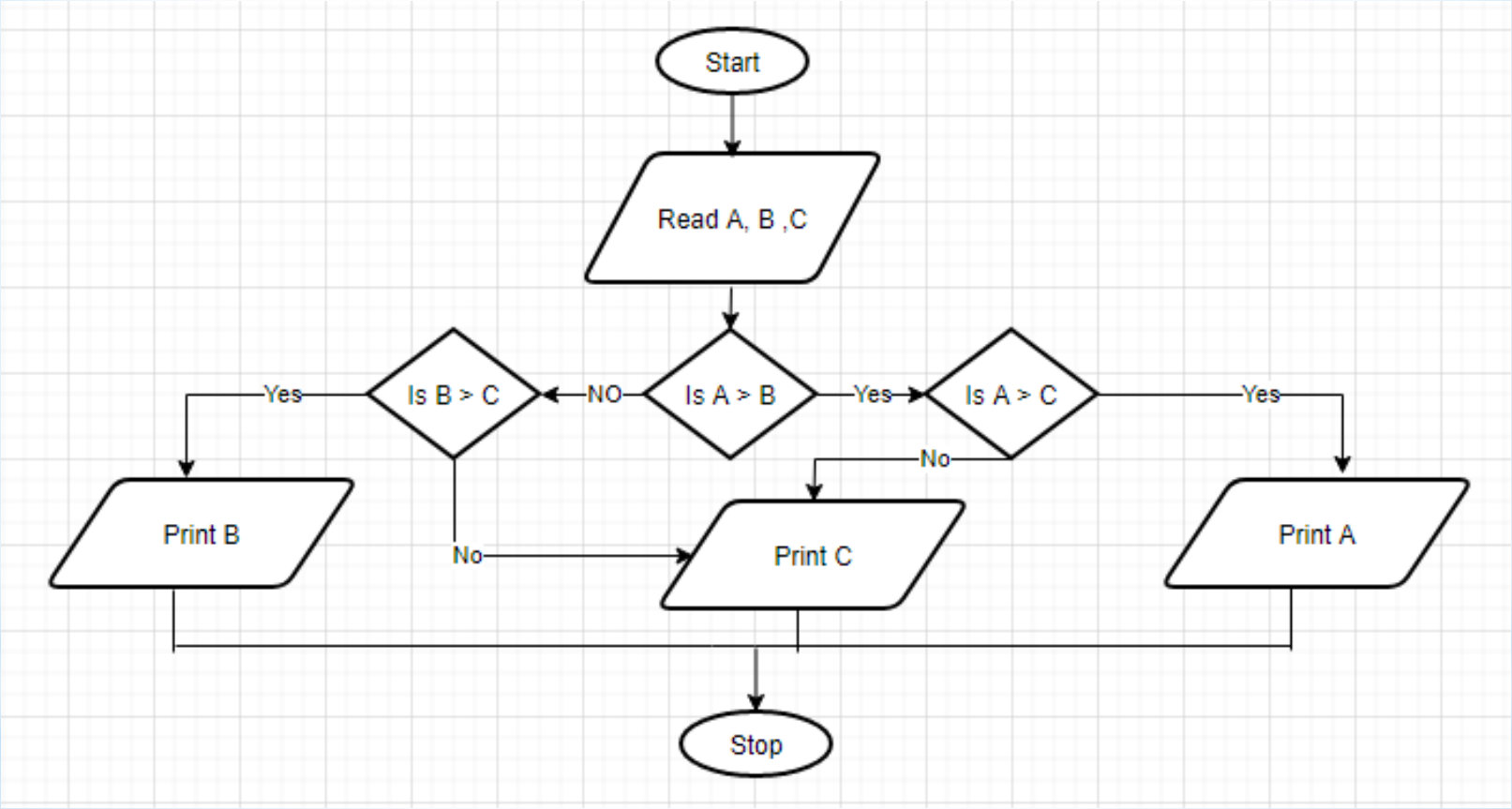
Flowchart Cont…

Flowchart to check number is negative or positive



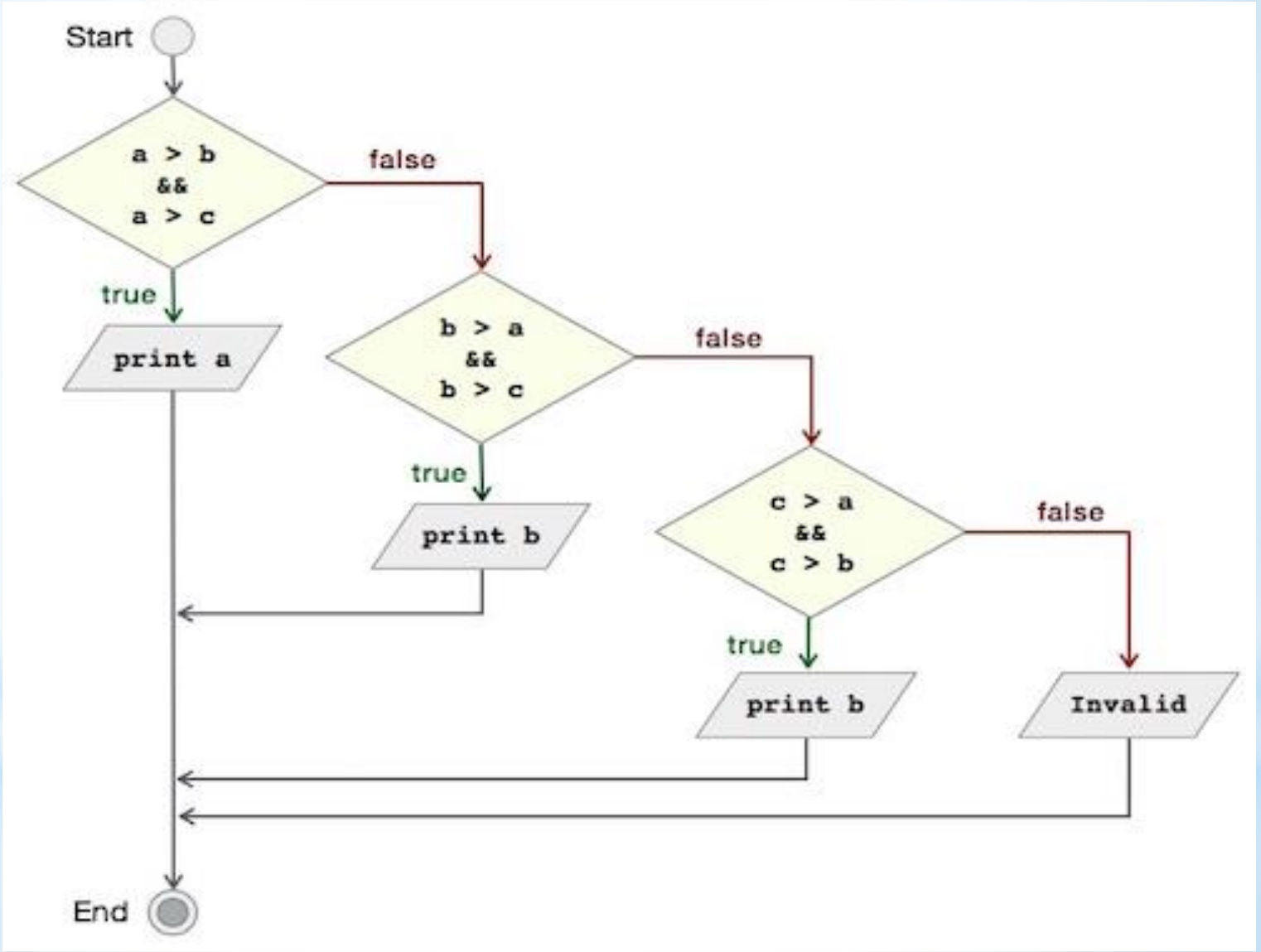
Flowchart Cont…

Flowchart to find maximum number from three numbers



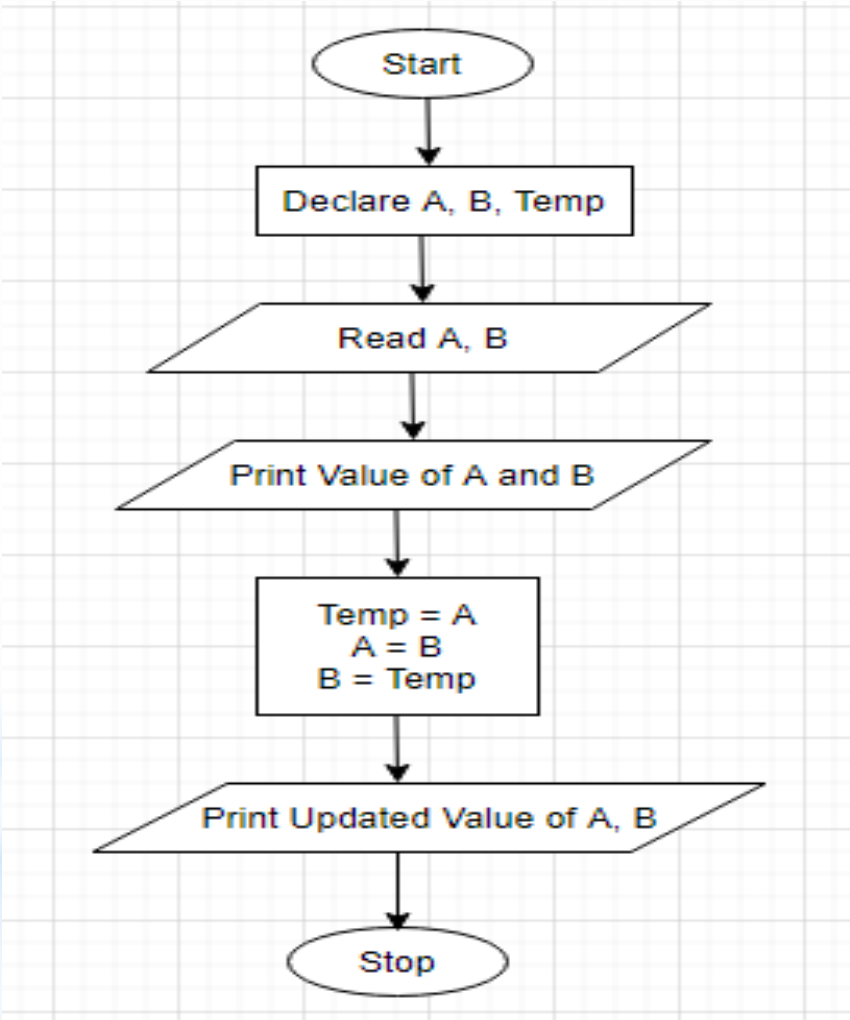
Flowchart cont..

Another way



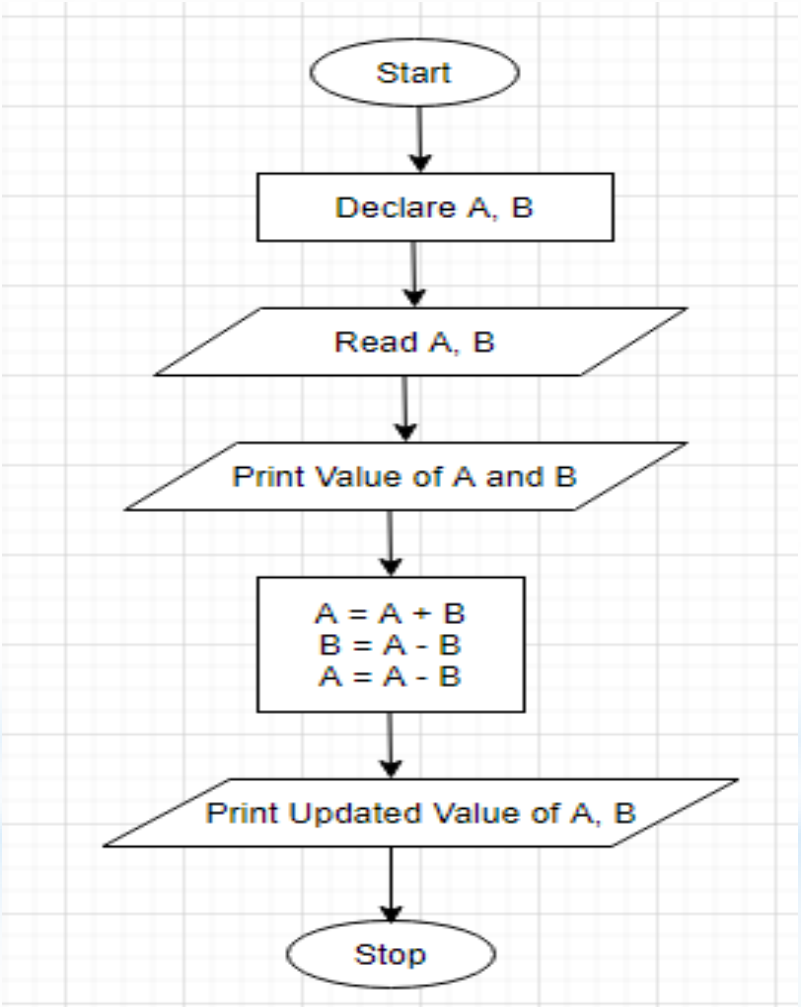
Flowchart cont..

Flowchart to swap value of two variable using temp variable



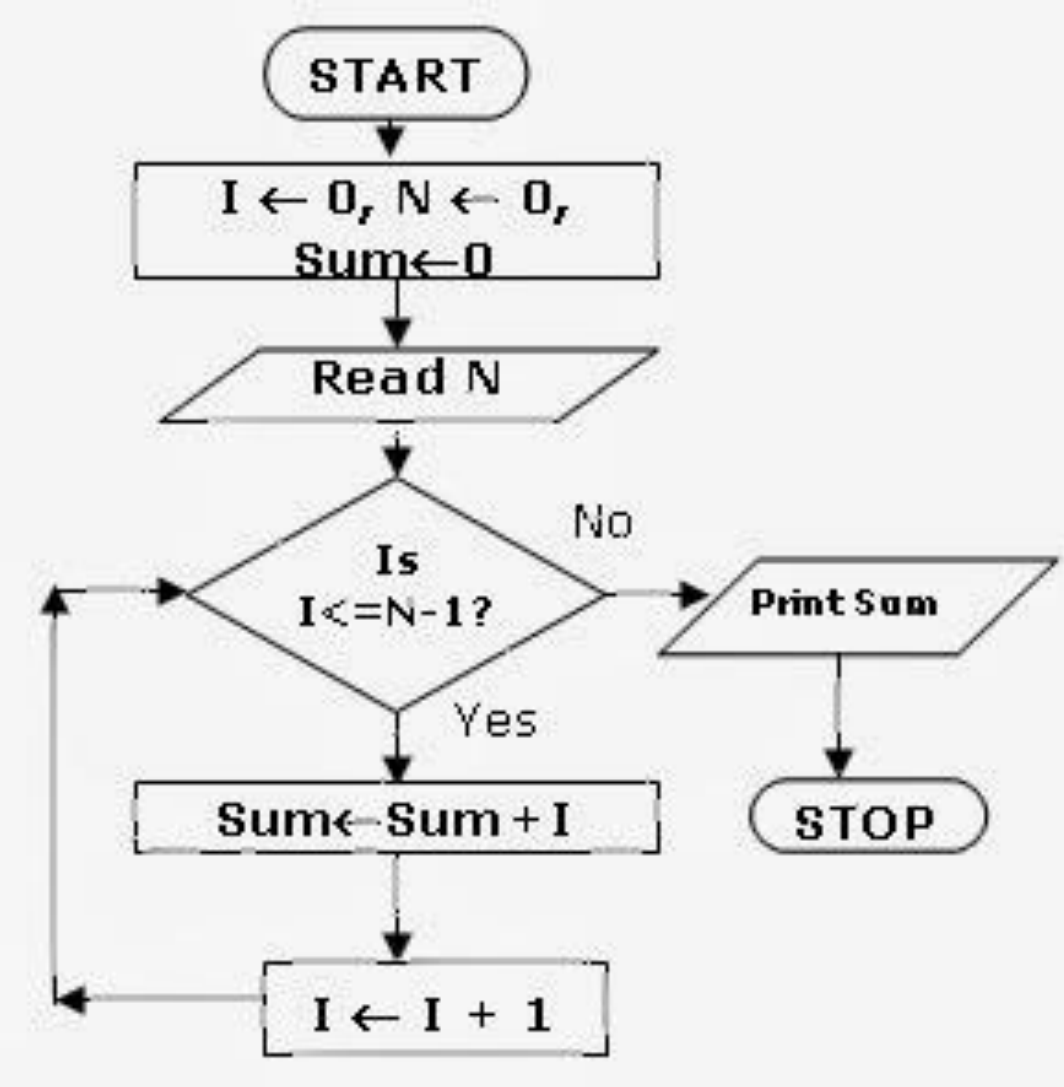
Flowchart cont…

Flowchart to swap value of two variable without using temp variable.



Flowchart cont…

Flowchart to print sum Of first N number using loop.



Structure of C Program

- Any programming language is having its own model or a structure for the program
- In ‘C’ also it has its predefined structure to be followed to develop the program, as below

Documentation	**Comments, Some Description, Programmer Name and any other useful points.**
Link	Provides instruction to the compiler to link function from the library function.
Definition	Consists of symbolic constants.
Global declaration	Consists of function declaration and global variables.
main( ) **{** **}**	Every C program must have a main() function which is the starting point of the program execution.
Subprograms	User defined functions.

Structure of C Program

BASIC STRUCTURE OF A 'C' PROGRAM:

	Example:
Documentation section **[Used for Comments]**	//Sample Prog Created by:Bsource
Link section	#include<stdio.h> #include<conio.h>
Definition section	void fun();
Global declaration section **[Variable used in more than one function]**	int a=10;

main()
{
Declaration part
Executable part
}
	void main()
	{
	clrscr();
	printf("a value inside main(): %d",a);
	fun();
	}

Subprogram section
[User-defined Function]
Function1
Function 2
:
:
Function n
	void fun()
	{
	printf("\na value inside fun(): %d",a);
	}

Structure of 'C' Program with example

```
//This program is to print Hello Students!
#include <stdio.h>
#include <conio.h>
#define PI 3.14;
void sum( );
void main( )
{
   printf("Hello Students");
   show( );
   getch( );
}
void show( );
{
   printf("This is Prof. Bhavika Vaghela");
}
```

Documentation

Link (Header file)

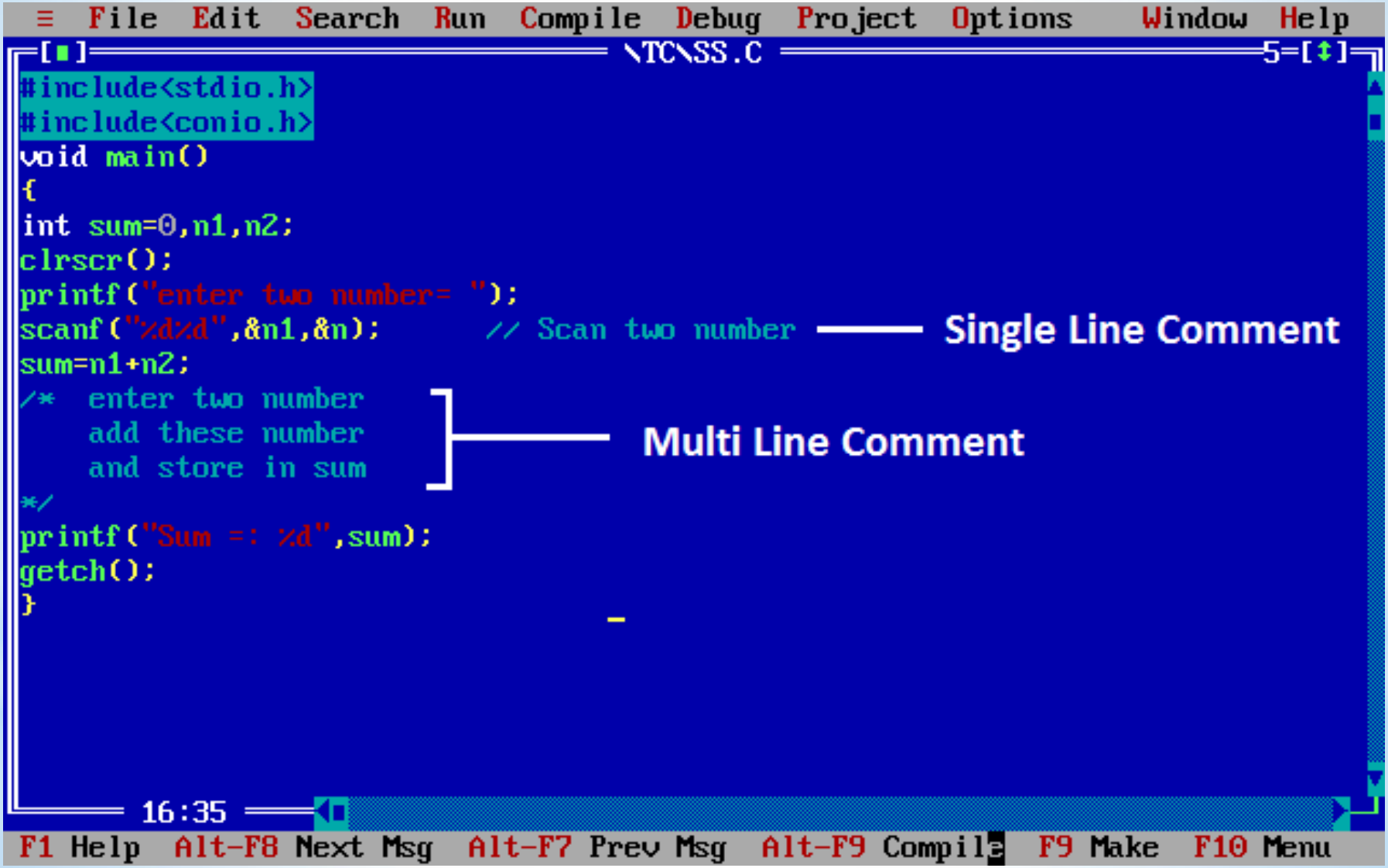
Definition

Global Declaration

Main Function

Sub Programs

Screen of Turbo C



Elements of C

- C Character Set
- Keywords
- Identifiers
- Comments in C Language
- Whitespace
- Constants
- Variables

Character Set in C

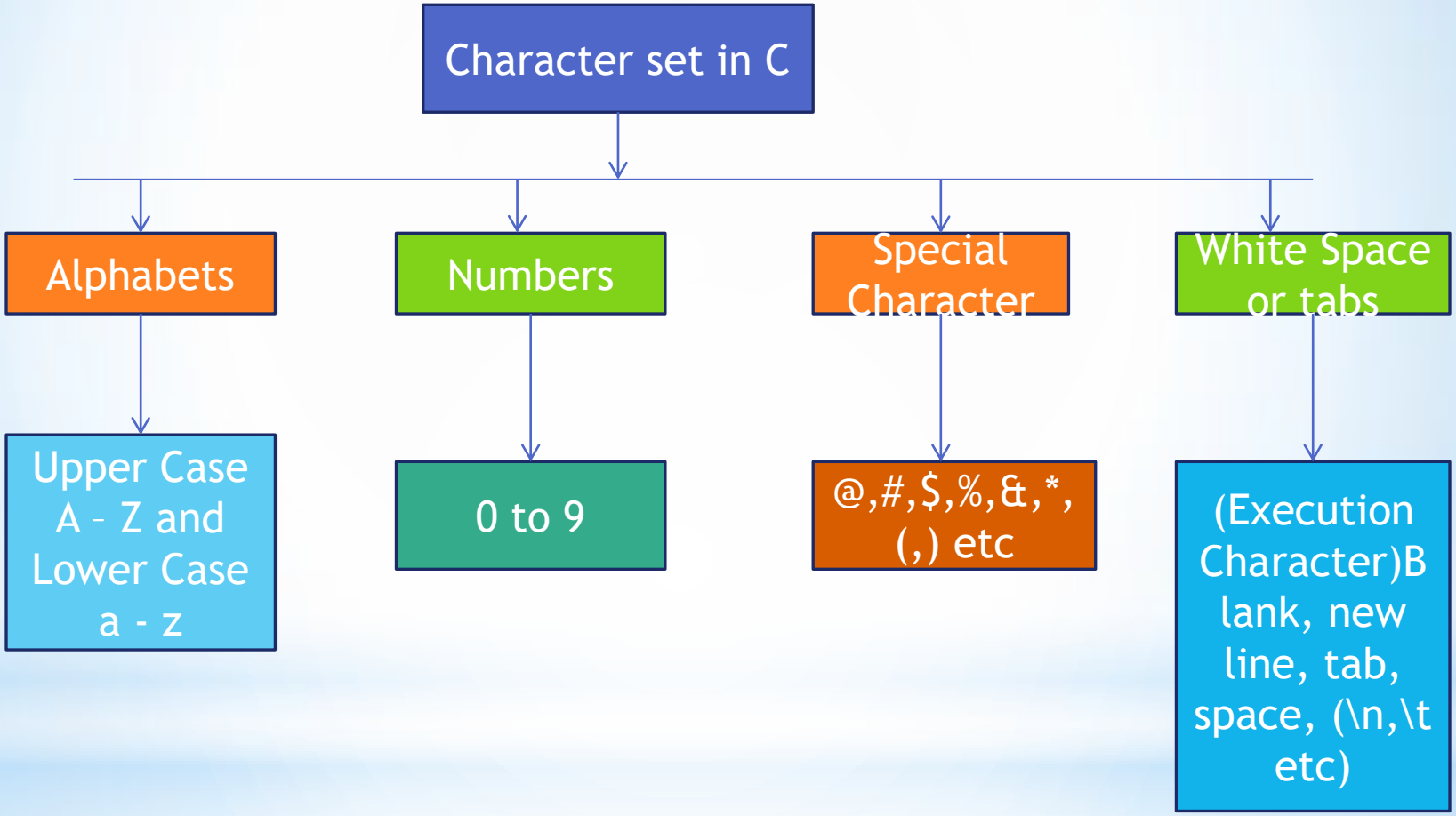
- The C character set consists of upper and lowercase alphabets, digits, special characters and white spaces.
- The alphabets and digits are altogether called as the alphanumeric character.
- Each character is represented by a number. Which is known as ASCII value.
- The ASCII character set, for example, uses the numbers 0 to 127 to represent all English characters as well as special control characters.
- The character set is the fundamental raw material of any language and they are used to represent information.
- ASCII (American Standard Code for Information Interchange).

Character set of C Language:

C compiler allows to use many no. of characters

Lower case letters : a b c … z
Upper case letters : A B C….Z
Digits : 0 1 2 3…9
Other characters : + - * ( ) & % $ # { } [ ]’ “ : ; etc.
White space characters : blank, new line, tab space etc.

Cont...


Character set in C
Alphabets
Numbers
Special Character
White Space or tabs
Upper Case A – Z and Lower Case a - z
0 to 9
@,#,$,%,&,*, (,) etc
(Execution Character)Blank, new line, tab, space, (\n,\t etc)

What is execution Character

- Certain ASCII characters are unprintable, which means they are not displayed on the screen or printer. Those characters perform other functions aside from displaying text. Examples are backspacing, moving to a newline, or ringing a bell. They are used in output statements.
- Escape sequence usually consists of a backslash and a letter or a combination of digits.
- An escape sequence is considered as a single character but a valid character constant.
- These are employed at the time of execution of the program.
- Execution characters set are always represented by a backslash (\) followed by a character.
- Note that each one of character constants represents one character, although they consist of two characters.
- These characters combinations are called as escape sequence.

Execution Character

Escape Sequence	Meaning
\n	New Line
\t	Horizontal Tab
\b	BackSpace
\r	Carriage Return
\a	Audible bell
\'	Printing single quotation
\"	printing double quotation
\?	Question Mark Sequence
\\	Back Slash
\f	Form Feed
\v	Vertical Tab
\0	Null Value
\nnn	Print octal value
\xhh	Print Hexadecimal value

ASCII Value

Parul University

These are Control Characters			These are Printable Characters							
ASCII Value	Character	Meaning	ASCII Value	Character	ASCII Value	Character	ASCII Value	Character
0	NULL	null	32	Space	64	@	96	`
1	SOH	Start of header	33	!	65	A	97	a
2	STX	start of text	34	"	66	B	98	b
3	ETX	end of text	35	#	67	C	99	c
4	EOT	end of transaction	36	$	68	D	100	d
5	ENQ	enquiry	37	%	69	E	101	e
6	ACK	acknowledgement	38	&	70	F	102	f
7	BEL	bell	39	'	71	G	103	g
8	BS	back Space	40	(	72	H	104	h
9	HT	Horizontal Tab	41	)	73	I	105	i
10	LF	Line Feed	42	*	74	J	106	j
11	VT	Vertical Tab	43	+	75	K	107	k
12	FF	Form Feed	44	,	76	L	108	l
13	CR	Carriage Return	45	-	77	M	109	m
14	SO	Shift Out	46	.	78	N	110	n
15	SI	Shift In	47	/	79	O	111	o
16	DLE	Data Link Escape	48	0	80	P	112	p
17	DC1	Device Control 1	49	1	81	Q	113	q
18	DC2	Device Control 2	50	2	82	R	114	r
19	DC3	Device Control 3	51	3	83	S	115	s
20	DC4	Device Control 4	52	4	84	T	116	t
21	NAK	Negative Acknowledgement	53	5	85	U	117	u
22	SYN	Synchronous Idle	54	6	86	V	118	v
23	ETB	End of Trans Block	55	7	87	W	119	w
24	CAN	Cancel	56	8	88	X	120	x
25	EM	End of Medium	57	9	89	Y	121	y
26	SUB	Sunstitute	58	:	90	Z	122	z
27	ESC	Escape	59	;	91	[	123	{
28	FS	File Separator	60	<	92	\	124	|
29	GS	Group Separator	61	=	93	]	125	}
30	RS	Record Separator	62	>	94	^	126	~
31	US	Unit Separator	63	?	95	_	127	DEL

DEL is also Control Characters

Keyword (reserved keyword) in C

- In c all reserved keywords are in lower case.
- One can not take variable name as keyword other wise it fire an error.
- Total 32 keywords are present in C programming language.

C KEYWORDS OR RESERVED WORDS

auto	break	case	char
const	continue	default	do
int	long	register	return
short	signed	sizeof	static
struct	switch	typedef	union
unsigned	void	volatile	while
double	else	enum	extern
float	for	goto	if

Identifier in C

- C identifiers represent the name in the C program, for example, variables, functions, arrays, structures, unions, labels, etc.
- an identifier is a collection of alphanumeric characters that begins either with an alphabetical character or an underscore.

C imposes certain rules for naming identifiers:

- Identifiers should consist of alphabets, digits or underscores (_)only
- First character should be an alphabet or underscore.
- Identifier should not be a Keyword.
- length of the identifiers should not be more than 31 characters.
- Identifiers are case sensitive
- Commas or blank spaces cannot be specified within an identifier.
- Identifiers should be written in such a way that it is meaningful, short, and easy to read.

Cont…

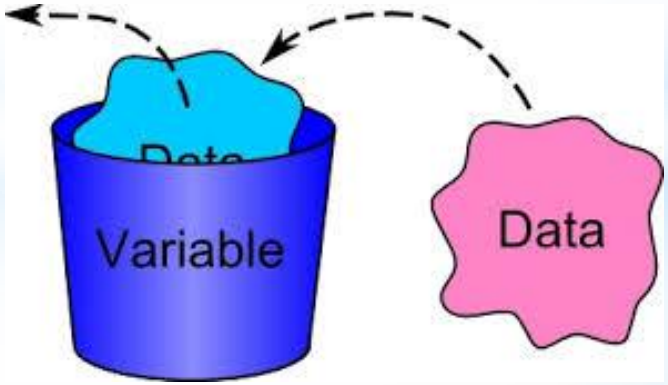
- C is case-sensitive language
  so my_var and MY_VAR are two distinct identifiers.
- Some examples of valid identifiers:

  num , _address , user_name , email_1
- Examples of invalid identifiers:

  1digit , my var , int , some#

Variable in C

- A **variable** is a name of the memory location.
- It is used to store data. Its value can be changed, and it can be reused many times.
- It is a way to represent memory location through symbol so that it can be easily identified.
- For example you have used suitcase to store clothes, match box to store match sticks etc.
- In the same way variables of different data type is used to store different types of data.



Cont..

- Variable have some data type which show which type of data one can store.
- **Syntax** : data type variable_name = value;

  data type variable_name;

Rules to declare variable in c

- A variable can have alphabets, digits, and underscore.
- A variable name can start with the alphabet, and underscore only. It can't start with a digit.
- No whitespace is allowed within the variable name.
- A variable name must not be any reserved word or keyword, e.g. **int, float**, etc.
- There is no rule on how long a variable can be. However, the first 31 characters of a variable are discriminated by the compiler.

Cont…

- In C programming, you have to declare a variable before you can use it.
- It is a common practice in C programming to declare all the variables at the beginning of the program.

Valid Variable name	Invalid Variable name
int _name	int 12num
Int number_1	int float
float value1, value2;	char mes1 mes2;
char first_name	float #num1

Cont…

Types of Variable

- Local variable
- Global variable
- Static variable
- Automatic variable
- External variable

Local Variable : a variable which id declare inside some function or block which is not accessible outside of that block.

Global Variable : A variable that is declared outside the function or block is called a global variable. Any function can change the value of the global variable. It is available to all the functions.

- Global variable must be declare at start of the block. (Top of the program)

Cont…

Static Variable : static variable declare using **“static”** keyword. Like **static int num1=30;**

- Value of static variable incremented on each call so it will retain its value on function call. (**for more please refer program).

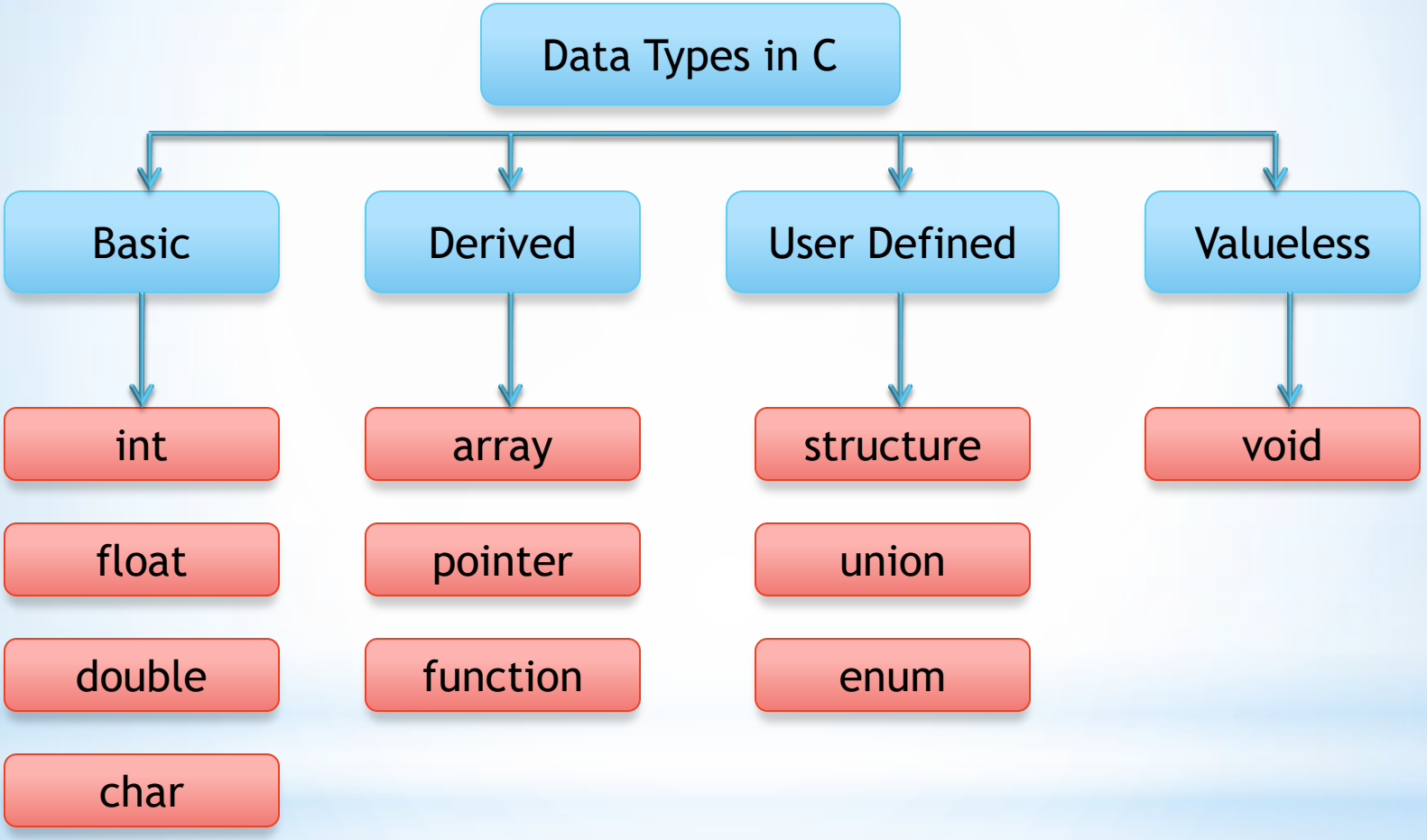
Automatic Variable : All variable in C are auto variable but if some one wants to declare it externally than have to use **“auto”** keyword. Like **auto int num1=20;**

External Variable : External variable is declare using “**extern**” keyword. This kind of variable is accessible in other c program also. Like **extern int value=100;**

Program

```
//Program which show types of variable
#include<conio.h>
#include<stdio.h>
void myfunction();
int num1=20; //global variable
void main()
{
        int num=25; //local variable    or auto variable also
        clrscr();
        printf("\n value of num is : %d",num);
        printf("\n value of num1 in main funciton : %d",num1);
        myfunction();
        printf("\n--------------calling myfunction again--------------");
        myfunction();
        printf("----------Again Calling myfunction---------------");
        myfunction();
        getch();
}
void myfunction()
{
        int num2=40; //local variable of myfunciton
        static int num4=10;     //static variable
        num2=num2+1;
        num4=num4+1;
        printf("\n value of num2 is : %d", num2);
        printf("\n value of num4 is : %d", num4);
        printf("\n accessing global variable here also : %d",num1);
}
```

Data Types in C Programming



Cont…

Type	Storage size	Value range
char	1 byte	-128 **to** 127
unsigned char	1 byte	0 **to** 255
signed char	1 byte	-128 **to** 127
int	2 bytes	-32,768 **to** 32,767
unsigned int	2 bytes	0 **to** 65,535
short	2 bytes	-32,768 **to** 32,767
unsigned short	2 bytes	0 **to** 65,535
long int	4 bytes	-2,147,483,648 to 2,147,483,647
unsigned long int	4 bytes	0 to 4,294,967,295

For more about data types please visit https://www.javatpoint.com/data-types-in-c

Placeholder in C

- Placeholders are used to determine when what type of values are going to be input or displayed, depending on what type of functions that you use.
- **scanf()** function requires input placeholders to allow data to be transferred to the specific variable.
- Output placeholders are used to display data onto the screen. **printf()** function uses the output placeholder.
- Normally we use %d, %s, %c, %f as placeholder while we read or print data. For more please checkout table on next slide.

Cont…

Format specifier	Description	Supported data types
%c	Character	char, unsigned char
%d	Signed Integer	short, unsigned short, int, long
%e or %E	Scientific notation of float values	float, double
%f	Floating point	Float
%g or %G	Similar as %e or %E	float, double
%hi	Signed Integer(Short)	short
%s	String	char *
%u	Unsigned Integer	unsigned int, unsigned long

For more please visit
https://codeforwin.org/2015/05/list-of-all-format-specifiers-in-c-programming.html
http://shangyilim.blogspot.com/2009/02/placeholders-placeholders-are-used-to.html
https://www.javatpoint.com/c-format-specifier

Constant in C

- Constant value mean fixed value which is not change or not allow to change through out the program.
- In C one can declare all types of constant value like decimal value, float value, string value etc.
- To declare constant variable or value **“const”** keyword is use.
- So, Constants are fixed value variables, whose value cannot be altered throughout the execution of program.
- They behave like normal variables expect that they are **read-only** (once assigned cannot be modified).
- An identifier also can be defined as a constant.

const double PI = 3.14

Cont…

List of constant in C

Type of Constant	Example
Decimal Constant	10, 20, 450 etc.
Real or Floating-point Constant	10.3, 20.2, 450.6 etc.
Octal Constant	021, 033, 046 etc.
Hexadecimal Constant	0x2a, 0x7b, 0xaa etc.
Character Constant	'a', 'b', 'x' etc.
String Constant	"c", "c program", "c in javatpoint" etc.

We can declare constant in two way

- **const** keyword
- **#define** pre-processor

Cont…

Syntax to define constant

const <data-type> <constant-name> = <constant-value>;

Or

<data-type> const <constant-name> = <constant-value>;

Or

#define <constant-name> <constant-value> //using predecessor

Example

```
const int RATE =10;
const float INTRATE = 6.7
float const PI = 3.14
#define PI 3.14
#define MAX 20
```

**** Note : it is good practice to give constant name in UPPERCASE.**

Cont…

Program using const

```
#include <stdio.h>
#include <conio.h>
int main()
{
        const float PI = 3.14159;
        clrscr();
        float radius, area;
        PI = 3.14; // <-- Will generate error, it must not be modified
        radius = 12;
        area = PI * radius * radius;
        printf("Area = %f", area);
        getch();
        return 0;
}
```

Cont…

Using #define

```
#include <stdio.h>
#include <conio.h>
#define PI 3.14
int main()
{
        float radius, area;
        clrscr();
        //PI = 3.14; // <-- Will generate error, it must not be modified
        radius = 12;
        area = PI * radius * radius;
        printf("Area = %f", area);
        getch();
        return 0;
}
```

Comment in C

- Generally Comments are used to provide the description about the Logic written in program. Comments are not display on output screen.
- When we are used the comments, then that specific part will be ignored by compiler.
- In 'C' language two types of comments are possible
  - Single line comments
  - Multiple line comments

Single line comment declare using "//" like //program for xyz

- Multiple line or multiline comment declare using "/* comment */" like, /*this program is for

addition of two numbers*/

Exercise for algorithm and flowchart

1. Enter length and breadth of a rectangle and find its perimeter
2. Enter length in centimetre and convert it into meter and kilometre
3. Enter temperature in °Celsius and convert it into Fahrenheit.
4. Convert days into years, weeks and days.
5. find power of any number xy (x^y).
6. Find power of any number xy (x^y).
7. Enter any number and calculate its square root.
8. Enter marks of five subjects and calculate total, average and percentage.
9. Enter P, T, R and calculate Simple Interest and Compound Interest

Operator

- An operator is some special symbols which is use to perform mathematical and logical operations on given data.
- C programming support various types of operators like,
  - Arithmetic Operator
  - Increment and decrement operator
  - Relational (conditional) Operators
  - Logical Operators
  - Bitwise Operators
  - Assignment Operators
  - Special Operator

**** All Operator mostly executed in right to left direction only assignment & (Short hand) operator execute in left to right.**

Arithmetic Operator

- An arithmetic operator performs mathematical operations such as addition, subtraction and multiplication on numerical values (constants and variables) **For eg. if A=10 and B=20**

Operator	Description	Example	Output
Addition (+)	Add both the operands or value	A+B	30
Subtraction (-)	Subtract right hand side value from left hand side operand	A-B	-10
Multiplication (*)	Multiply both operands or value	A*B	200
Division (/)	Divide left side operand by right hand side operand and return quotient	B/A	2
Modulo (%)	Divide left side operand by right hand side operand and return reminder	B % A	0

Increment and Decrement Operator

- C Programming support two increment and decrement operator respectively ++ and - -.
- ++ operator is known as increment operator and its increment value by 1.
- - - operator is known as decrement operator and its decrement value by 1.

```
/ /Working of increment and decrement operators
#include <stdio.h>
#include<conio.h>
 int main()
{       int a = 10, b = 100;
        float c = 10.5, d = 100.5;
        printf("++a = %d \n", ++a);
        printf("--b = %d \n", --b);
        printf("++c = %f \n", ++c);
        printf("--d = %f \n", --d);
        getch();
        return 0;
}
```

Relational (Conditional) Operator

- A relational operator checks the relationship between two operands.
- If the relation is **TRUE, it returns 1**; if the relation is **False, it returns value 0.**
- Relational operators are used in **decision making and loops. Example A= 10 B =20**

Operator	Description	Example	Output
==	Check both operand is equal or not	A== B	False
>	Check left operand is grater than right one	A > B	False
<	Check left operand is less than right one	A < B	True
>=	Check left operand is grater than or equal to right one	A >= B	False
<=	Check left operand is less than or equal to right one	A <= B	True
!=	Check left operand is not equal to right one	A != B	True

Logical Operator

- Logical Operator return Boolean value based on condition.
- It will return True (1) or False (0) based on the condition.
- It basically use for decision making in C programming. Let's A=10, B=5 and C=12

Operator	Description	Example	Output
and (&&)	It will return true if both the condition become true else false	(A>B) && (A<C)	True (1)
or (||)	It will return true if any one the condition become true else false	(A>B) || (A>C)	True (1)
not(~)	It will return true if condition false and false if condition is true	~ (A > B)	False (0)

Bitwise Operator

- The bitwise operators perform bit by bit operation on the values of the two operands.
- If data is not in binary than it convert it first in binary and than perform operation.
- It returns decimal as a result.

Binary AND(&) : it perform bit by bit and operation and if both bit at same place are 1 than put 1 else it will put 0.

Binary OR(|) : it perform bit by bit or operation and if both bit at same place is 0 than put 0 else it will put 1.

Binary XOR(^) : it perform bit by bit XOR (exclusive or) operation. It will put 0 if both bit at the same place is same else put 1

Cont…

- **One's Complement(~) / binary negative** : It flips the bits. It will return 1 if bit is 0 and return 0 if bit is 1.
- **Left shift (<<)** : The left operand value is moved left by the number of bits present in the right operand. **E.g. 10<<n (n is number of bits and 10 is value).**
- **Right shift (>>)** : The left operand value is moved right by the number of bits present in the right operand. **E.g. 10>>n (n is number of bits and 10 is value).**

Cont...

p	q	p & q	p | q	p ^ q
0	0	0	0	0
0	1	0	1	1
1	1	1	1	0
1	0	0	1	1

Assume if A = 60; and B = 13; now in binary format they will be as follows:

A = 0011 1100

B = 0000 1101

A&B = 0000 1100

A|B = 0011 1101

A^B = 0011 0001

~A = 1100 0011

<<	Binary Left Shift Operator. The left operands value is moved left by the number of bits specified by the right operand.	A << 2 will give 240 which is 1111 0000
>>	Binary Right Shift Operator. The left operands value is moved right by the number of bits specified by the right operand.	A >> 2 will give 15 which is 0000 1111

Shorthand (Assignment) Operator

- An assignment operator is used for assigning a value to a variable. The most common assignment operator is "=". Lets A = 10 and B = 2.

Operator	Example	Same As	Output
=	A = B	A = B	A assign value 2
+=	A += B	A = A + B	A assign value 12
-=	A -= B	A = A – B	A assign value 8
*=	A *= B	A = A * B	A assign value 20
/=	A /= B	A = A / B	A assign value 5
%=	A %= B	A = A % B	A assign value 0

Other Operator

Operator	Description
sizeof()	sizeof() operator is use to find size of variable or (array, structure or constant) in byte.
Address (&)	Returns the address of a variable.
Pointer (*)	Pointer to a variable.
Conditional (? :)	If Condition is true ? then value X : otherwise value Y
Member Operator Dot (.) and arrow →	Is use for reference to object or structure and also use to reference to member of class, union, structure.

Compiler and Interpreter

- Source code : A Computer program normally written down in high level programming language which is human readable and understandable language which known as source code.
- So computer or machine only understand the 0 and 1. it not understand high level programming language so for converting source we use compile or interpreter.
- Both compiler and interpreter is use to convert code into machine level programming language.

What is compiler?

- Compiler is computer program which convert high level programming into machine readable code mean into machine level programming language.

Cont…

What is interpreter?

- An interpreter is a computer program, which coverts each high-level program statement into the machine code.
- The compiler and interpreter both did the same job but the only difference is compiler read entire program once and than convert it. Where as interpreter read code line by line and it convert code line by line into machine level.
- Please find the difference between compiler and interpreter on next slide.

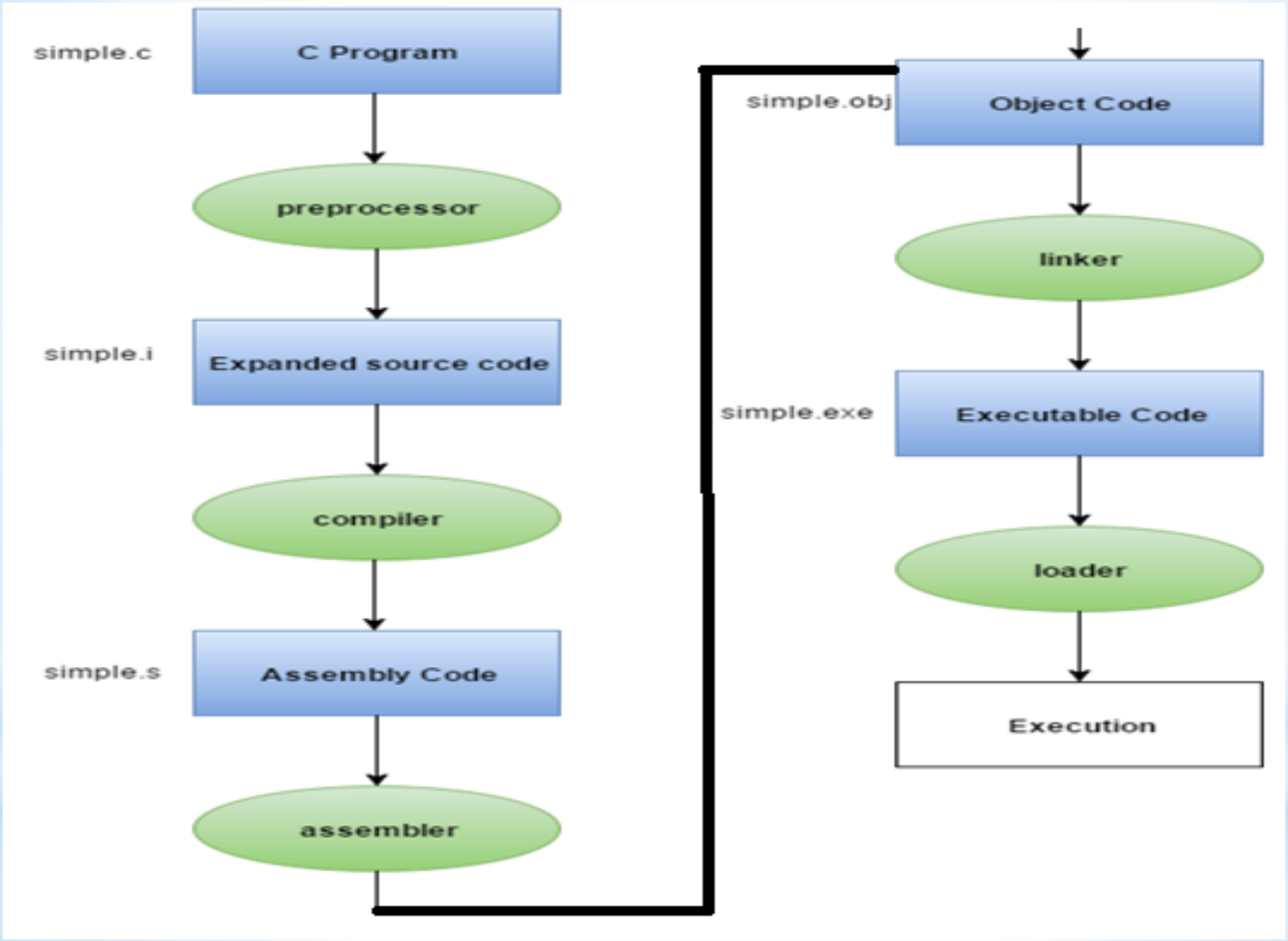
Cont…

Difference between compiler and interpreter

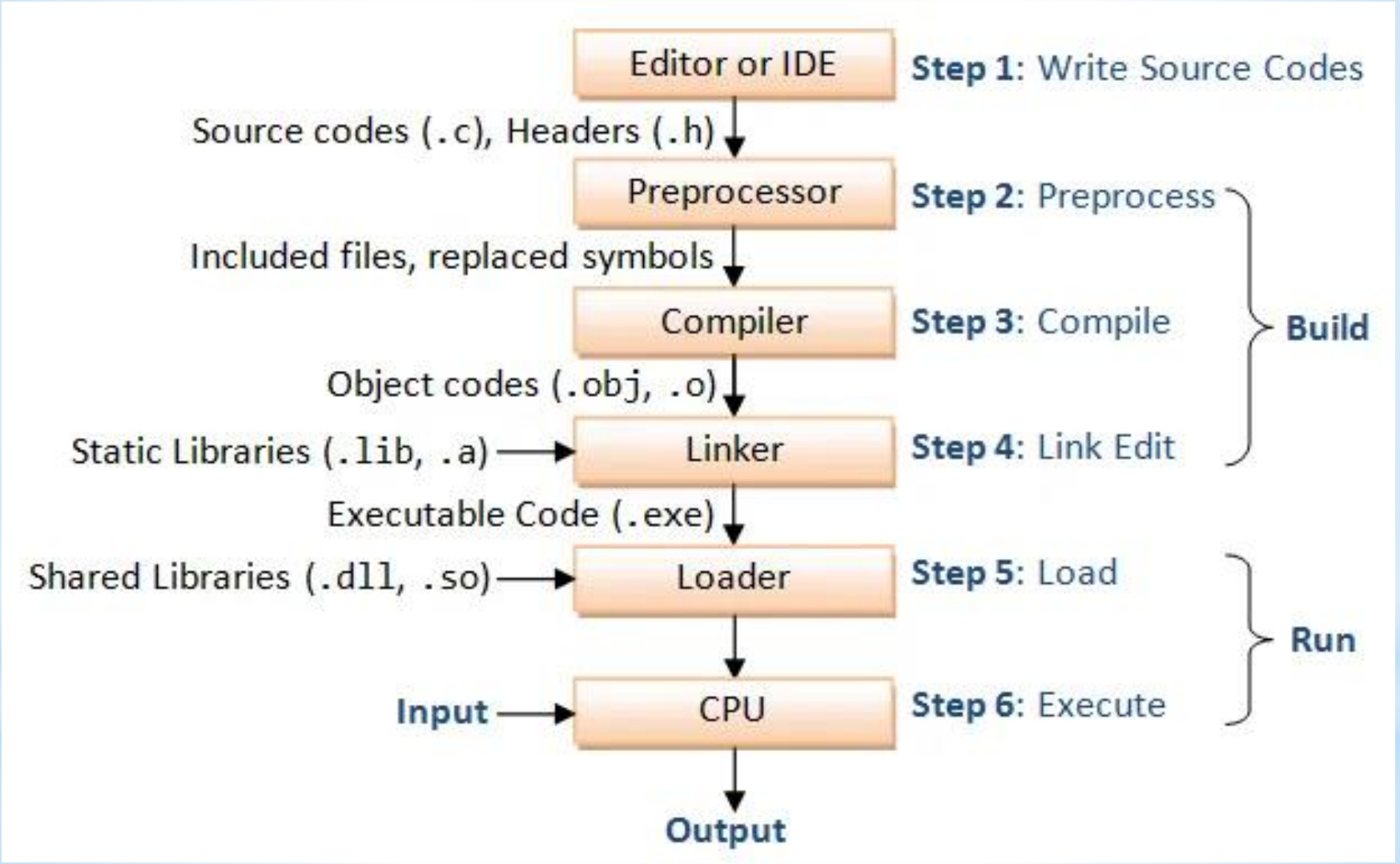
Compiler	Interpreter
Compiler transforms code written in a high-level programming language into the machine code, at once, before program runs	an Interpreter coverts each high-level program statement, one by one, into the machine code, during program run.
Compiled code runs faster	interpreted code runs slower.
Compiler displays all errors after compilation	the Interpreter displays errors of each line one by one.
Compiler is based on translation linking-loading model	Interpreter is based on Interpretation Method.
Compiler takes an entire program	Interpreter takes a single line of code

For more : https://www.guru99.com/difference-compiler-vs-interpreter.html
https://www.programiz.com/article/difference-compiler-interpreter

Compilation Process in C



Cont…


Editor or IDE
Step 1: Write Source Codes
Source codes (.c), Headers (.h)
Preprocessor
Step 2: Preprocess
Included files, replaced symbols
Compiler
Step 3: Compile
Build
Object codes (.obj, .o)
Static Libraries (.lib, .a)
Linker
Step 4: Link Edit
Executable Code (.exe)
Shared Libraries (.dll, .so)
Loader
Step 5: Load
Run
Input
CPU
Step 6: Execute
Output

Header File in C

- Header file in C Programming is those file which have ".h" extension.
- In header file all the function or micro as well some constant variable is declare which we normally use for writing C program.
- To add this header file in C programming one must have to use Pre-Processor directive "#include" followed by <name of header file> in C program. E.g. #include<stdio.h>
- A simple practice in C or C++ programs is that we keep all the constants, macros, system wide global variables, and function prototypes in the header files and include that header file wherever it is required.

Cont..

Name of Header file	Description
stdio.h	Input / Output Functions
conio.h	console input/output
assert.h	Diagnostics Functions
ctype.h	Character Handling Functions
cocale.h	Localization Functions
math.h	Mathematics Functions
setjmp.h	Nonlocal Jump Functions
signal.h	Signal Handling Functions
stdarg.h	Variable Argument List Functions

For more about header file please visit links

https://www.improgrammer.net/header-file-list-functions-c-language/

https://www.tutorialspoint.com/cprogramming/c_header_files.htm

User defined Header file

- One can also write user defined header file.
- It also store with ".h" extension.
- In that file one can include some user defined function name, constant variable, or other functionality which they want to use again and again in any C program.
- Same as inbuilt header file you can include user define header file using pre processor directive "#include" followed by user defined header file like ,

#include"user defined header file.h"

Let's discuss it with program.

User defined Header file

```
//user defined header file for arithmatic operation
extern int number1=100;
void addition(int n1,int n2)
{
        printf("\n addition of two number is : %d",(n1+n2));
}
void subtraction(int n1,int n2)
{
        printf("\n subtraction of two number is : %d",(n1-n2));
}
```

** like this you can write your user defined header file

For more please visit : https://www.geeksforgeeks.org/write-header-file-c/

C program which use UDF header file

```
//program to perform arithmetic operation using user defined header file
#include<stdio.h>
#include<conio.h>
#include"myheader.h“     //this is user defined header file
void main()
{
        int num1,num2;
        printf("\n enter value of number1 : ");
        scanf("%d",&num1);
        printf("\n enter value of number2 : ");
        scanf("%d",&num2);
        addition(num1,num2);
        subtraction(num1,num2);
        printf("\n accessing value of external variable of header file : %d",number1);
        getch();
}
```

Reference

https://www.tutorialspoint.com/cprogramming/index.htm

https://data-flair.training/blogs/applications-of-c/

https://www.w3schools.in/category/c-tutorial/

https://fresh2refresh.com/c-programming/c-interview-questions-answers/what-is-header-file-in-c-language/

https://www.javatpoint.com/c-programming-language-tutorial