BASIC OF COMPUTER PROGRAMMING (4310702)

IMP QUESTION WITH ANWSER

1. What is a flowchart ? Limitation of flowchart.

- Flowchart is graphical or diagrammatical representation of sequence of anyproblem to be solved by computer programming language.
- Flowchart is a diagrammatic representation of an algorithm.

Limitation

- Complex logic: Sometimes, the program logic is quite complicated. In that case, flowchart becomes complex and clumsy.
- Alterations and Modifications: If alterations are required the flowchart mayrequire re-drawing completely.
- Reproduction: As the flowchart symbols cannot be typed, reproduction of flowchart becomes a problem.
- The essentials of what is done can easily be lost in the technical details of how itis done.
- It is the total perception of the designer who draw the flowchart.
- It's working differ from one compiler to another ,one language to another in sometime.
- It need well-defined requirements.

2. Define Algorithm with example.

- An algorithm is a finite sequence of well defined steps or operations for solving aproblem in systematic manner.
- These are rules for solving any problems in proper manner.
- Instructions are written in the natural language.
- It is also called step by step solution.

Example: **Write an algorithm to find area of circle.**

Step 1: Input R

Step 2: Compute A = 3.14 * R * R

Step 3: Print A

Step 4: Stop

3. List the Feature of C language.

- **C is portable**.: 'C' Program can be run on any hardware.
- **C does not support input/output statements**.: 'C' does not support input/output statement like input and print, but support library functions printf( ) and scanf( ).
- C **supports bitwise operators**.: The C support bitwise operator like AND, OR, NOT etc.

normally these operations do not supported by higher level languages..

- **C is modular language**.: The program written in from of functions. A C program is group of one or more functions. Dividing the program into small functions makes it to easy develop and maintain the programs. Each function is easy to understand as it is small. The method of dividing a program into small functions is called modularity.

4. List out C Token.

- The smallest individual unit in a C program is known as C token.
- C has six types of token as given below:

1. Keywords (float,while)
2. Identifier(main,amount)
3. Constants(-15.2, 10)
4. String("abc", "hello")
5. Special Symbols ([,] , *, ! etc)
6. Operators (+,-, *,/)

5. Explain keyword.

- Keywords are those words whose meaning is predefined.
- The programmer cannot change the meaning of keywords.
- There are 32 keywords available in c.
- All the keywords must be written in lowercase.
- Following is the list of keywords:

Int	char	float	double
Long	short	signed	unsigned
If	else	do	while
For	break	continue	switch
Case	default	consulting	goto
Struct	union	void	return
Static	extern	volatile	register
Auto	enum	sizeof	typedef

6. Explain identifier.

- Identifier is are the words which are defined by the programmer in a program.
- Identifier refers to the name of variable, array, function etc.
- It consists of letters, digits and underscore.
- First character must be an alphabet or underscore.
- Keywords cannot be use as identifier.
- Identifier name are case sensitive.
- The name of identifier should be meaningful.

7. Write the Basic Structure of C Program.

- Documentation section:
  Documentation consists of a set of comment line(/* */) giving the name of the program.
- Header File(Link Section):
  Link section provides instruction to the compiler to link function from the system library(Header Files like #include<stdio.h>).
- Constant Declaration:
  Definition section defines all symbolic constants(e.g.-#define pi 3.14).
- Global declaration section:
  There are some variable that are used more than one function. Such Variables arecalled global variable and that is outside all the function. This section also declares the entire user define functions.
- Main () function section:
  Every c program must have one main () function section. This section contains twoparts. Declaration part and executable part. Declaration part declares all the variables used in the executable part. These two parts must be appearing between opening and closing braces.

8. List out Advantages of 'C' language.

- C language contains rich set of built in functions.
- 'C' operators can be used to write any complex program.
- 'C' is a middle level language, which combines the capabilities of an assemblylanguage and higher level language.
- Programs written in c are efficient and fast, because it contains variety of datatypes and powerful operators.
- 'C' is highly portable, means c program written for one computer can be run onanother with little or no modification.

9. Explain constant in detail.

- Constant means fixed value.
- The value of the constant cannot change during execution of the program.
- C supports following types of constant:
  1. Integer
  2. Real
  3. Character
  4. String

1. Integer constant: Integer constant is a sequence of digits.

- Integer constant can be +ve or –ve number.
- Special symbols such as space, comma, currency etc are not allowed.

- There are three types of integer.
  Decimal: A decimal integer constant consist of any combination of digits from 0to 9.Example 123,-345,001.
  Octal: An octal integer constant consists of any combination of digits from 0 to 7with a leading 0.Example:01234, 0564.

2. **Real constant:** Real constant is a sequence of digits with decimal points. Such aconstant are used to represent weight, height etc.
3. **Character constant**: Character constant is a single character enclosed in a singlequotation mark.

- Example: 'a'
- Back slash constants are the special type of character constants which actuallyconsists of two characters. These are known as escape sequences.
- The combination of backslash and a character is known as escape sequence.
- Escape sequence start with back slash '\' character.

Escape Sequence	Meaning
'\0'	End of string
'\n'	End of line
'\t'	Horizontal tab

4. **String Constant:** String constant is a sequence enclosed in a double quotationmark.
   Example: ''Hello''
   Constant can be declared in a program using count keyword.For
   Example: const PIE = 3.14**;**

10. Define Variable & explain its rules.

- A variable is a name which is used to store a temporary value.
- The value of the variable may changes during execution of the program.
- Rules for naming variable:
- The variable name may consist of letters (A-Z, a-z), digits (0-9) and underscoresymbol (_).
  Example: int no_1; / int no1; / int NO1;
- The variable name must begin with a letter. Some system permit underscore as a first character.
  Example: int 1NO; is not valid
- The length should not be more than eight characters.
- Variable name should not be a keyword.
  Example: int case is not valid
- A white space is not allowed in the name of variable.
  Example: int NO 1; is not valid.
- Variable is case sensitive. Uppercase and lowercase are different.
- **Example:** int MAX;int max;
  Here both MAX and max are treated as different variable.

11. Define Data Type and their types.

- **Data type** is used to specify which type of value to be stored in variable.

❖ There are four basic data types supported by C.

Type	Size(Bytes)	Size(Bits)	Range	Control String
char	1	8	-128 to +128	%c
int	2	16	-32768 to +32767	%d
float	4	32	3.4 e -38 to 3.4 e +38	%f
double	8	64	1.7 e -308 to 1.7 e +308	%ld

12. Define symbolic constants & list out Rules of defining symbolic constant.

❖ #define is a preprocessor directive which is used to defined symbolic constant.

❖ Symbolic name is a name of the constant, which we want to define.

❖ Value of the constant represents the value, which we want to assign to the constant.

Rules of defining symbolic constant

❖ The rules for naming symbolic constant are same as the rules for naming variables. Symbolic names are written in capital letters to distinguish them from the normal variable.

❖ There should be no blank space between the pound sign '#' and the word defines.

❖ '#' must be the first character in the line.

1. A blank space is required between #define and symbolic name and between thesymbolic name and the value.
2. #define statement must not end with a semicolon.
3. After defining the symbolic constant they should not be assigned any other valuein the program.
4. There is no need to declare data type for the symbolic constant.

13. Define Operator. & explain Arithmetic Operators with example.

❖ An operator is a some special characters (symbol) that tells the computer to perform certain mathematical or logical manipulations.

Arithmetic Operators

Operator	Meaning
+	**Addition**
-	**Subtraction**
*	**Multiplication**
/	**Division**
%	**modulo**

Ex: a+b, a-b, a*b, a%b, a/b
where a & b variables & known as Operand (variable).
Example:

```
int a,b,sum,sub,div,mul,mod;a =
15;
b = 3;
sum = a+b;          // sum = 18
sub = a-b;          // sub = 12
mul = a*b;          // mul = 45
div = a / b;        // div = 5
mod = a%b;          // mod = 0
```

14. Explain Relational Operator with example.

- Compare two quantities & depending on their relation.

Operator	Meaning
<	Is less than
<=	Is less than or equal to
>	Is greater than
>=	Is greater than or equal to
==	Is equal to
!=	Is not equal to

- Relational operators are used in decision statements.
- Such as while & if to decide the course of action of a running program.
- Example:

```
int a = 15, b = 12;
```

Expression	Result
a>b	True
a<b	False
a>=b	True
a<=b	False
a==b	False
a!=b	True

15. Explain Increment & Decrement Operators in detail.

- Increment Operator ++
- Decrement operator --
- The increment operator ++ adds 1 to the operand, while decrement operator --subtracts 1.
- We use the increment and decrement statements in for and while loops.
- A prefix operator first adds 1 to the operand and then the result is assigned to thevariable on the left.
- A postfix operator first assigns the value to the variable on the left and thenincrement the operand.
- Postfix Increment Operator

Example:

```
int a,b;
a=10;
b = a++;
printf("a = %d",a);
printf("b = %d",b);
```

Output:

```
a=11
b=10
```

- Prefix Increment Operator

Example:

```
int a,b;
a=10;
b = ++a;
printf("a = %d",a);
printf("b = %d",b);
```

Output:

```
a=11
b=11
```

- Postfix Decrement Operator
- Example:

```
int a,b;
a=10;
b = a--;
printf("a = %d",a);
printf("b = %d",b);
```

Output:

```
a=9
b=10
```

- Prefix Decrement Operator
- Example:

```
int a,b;
a=10;
b = --a;
printf("a = %d",a);
printf("b = %d",b);
```

Output:

a=9

b=9

16. Explain Conditional Operators (Turnery Operator) in detail.

- Syntax: exp1 ? exp2:exp3
  - where, exp1,exp2 & exp3 are expressions.
- exp1 is evaluated first.
- If it is true then the expression exp2 is evaluated & becomes the value of anexpression.
- If exp1 is false, then exp3 is evaluated and its value becomes the value of theexpressions.
- Example:

```
a=25;
b=32;
ans=(a>b)?a:b;
```

In above example the answer is 35.

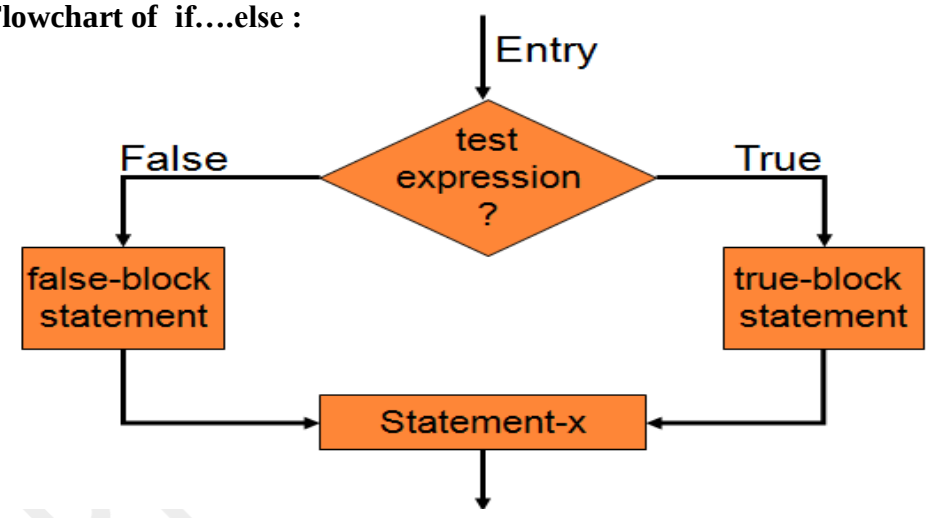
17. Explain the if….else statement with example.

- If the test expression is true then the true-clock statement are executed otherwisethe false-clock statement are executed.

General Form :

```
if( test expression)
{
   true-block statement
}
else
{
  false-block statement
}
```

Flowchart of if….else :



Example:

```
if(n%2==0)
{
printf(" n is even ");
}
else
{
printf(" n is odd ");
}
```

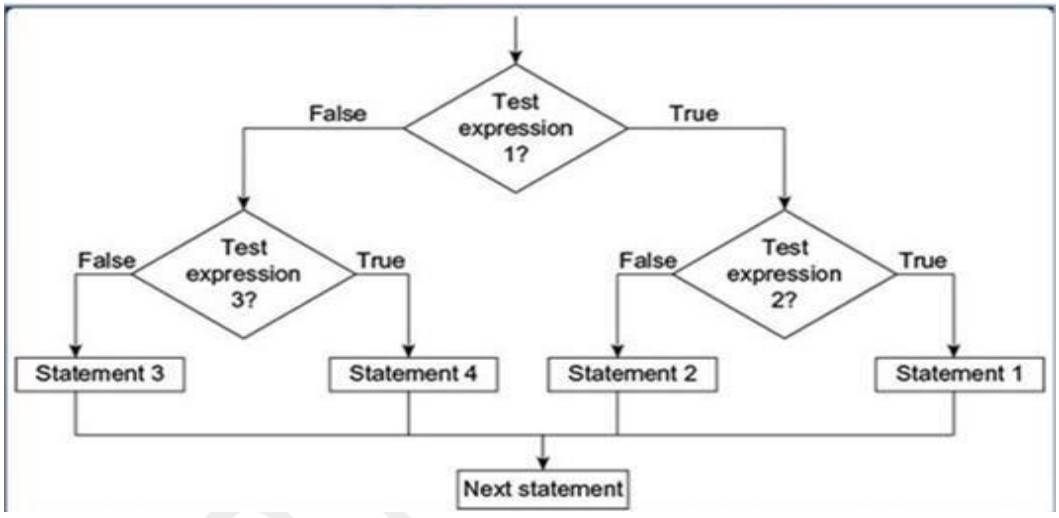
18. Explain the else if ladder with example.

❖ Used when we need multipath decisions.

General Form:

```
if(condition1)
statement1;
else if(condition2)
statement2;
else if(condition3)
statement3;
else
default-statement;
statement-x;
```

THE ELSE IF LADDER



❖ In this, the condition are evaluated from top to downwards.

❖ When all the n conditions are become false, then the final else containing thedefault

statement will be executed.

19. Explain SWITCH Statement with example.

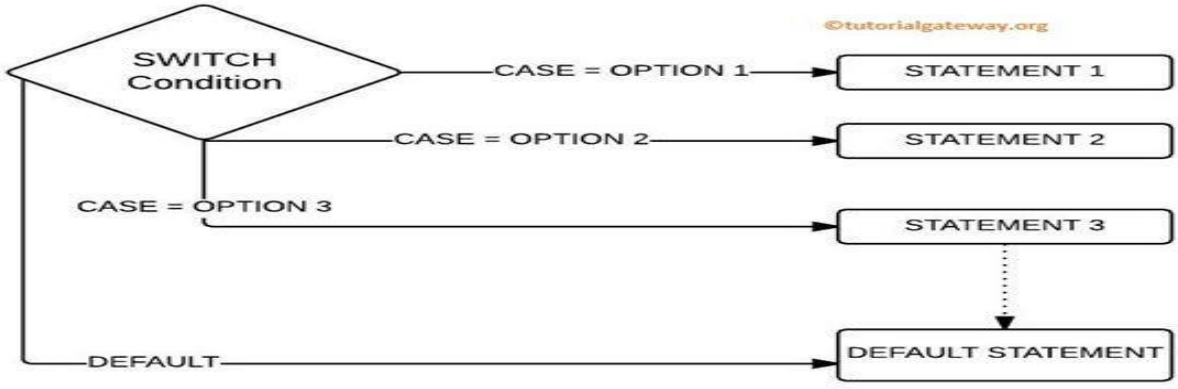
- We uses an if statement to control the selection. However, the complexity ofsuch a program increases when the number of alternatives increases.
- And the program becomes more difficult to read and follow.
- The switch statement tests the value of a given variable against a list of case values and when a match is found, a block of statements associated with that caseis executed.

- General Form

```
switch (expression)
{
   case value1:
                              block-1;
                              break;
   casevalue2:
                              block-2;
                              break;

   default:
                              default-block;
                              break;
}
statement-x;
```

- Where, the expression is an integer expression or characters.
- Value1, value2….. Are constants or constant expression and are known as case labels.
- Case labels end with a colon(:)
- When the switch is executed, the value of the expression is successfully comparedagainst the values. If a case is found whose value matches with the value of the expression, then the block of statement that follows the case are executed.

FLOWCHART OF SWITCH CASE



Example

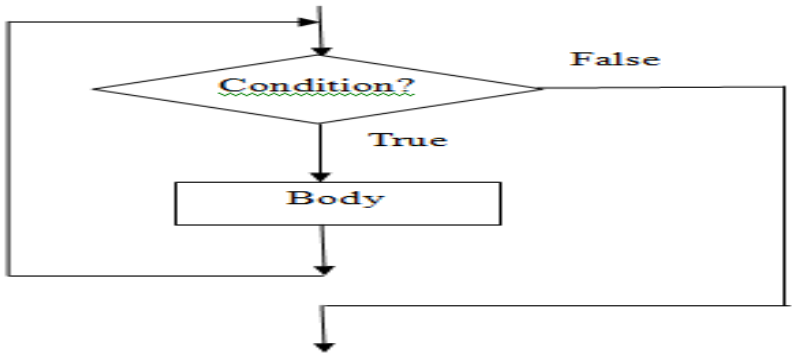
```
switch( op)
        {
                case +:
                                ans = a + b;
                                break;

                case -:
                                ans = a - b;
                                break;
                default:
                                printf(" wrong choice ");
                                break;
}
        printf(" Answer is : %d", ans);
```

20. Explain while loop with example.

- While loop is the entry control loop. Because in while loop first the condition ischecked.
- Syntax:

```
while (condition)
{
Body of the loop;
}
```

- {} is known as body of the loop. If body contain one statement then it is notnecessary to enclose body of the loop in the pair of brace {}.
- In while loop, the condition is evaluated first and if it is true then the statement inthe body of the loop is executed.
- After executing body , the condition is evaluated again and if it is true body isexecuted again. This process is repeated as long as the condition is true. The control move out once the condition is false.
- Flowchart:



Example: **Print 1 to 10 numbers using while loop.**

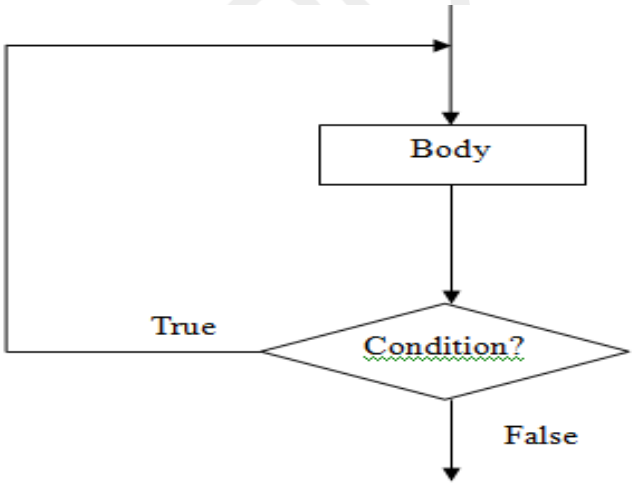
```
i =1;
while (i < = 10)
{
        printf( "% d \n " , i );i =
        i+1;
}
```

21. Explain Do-while loop with example.

- Do..while loop is a exit control loop. Because after the executing the body of theloop the condition is checked.
- **Syntax:**

```
Do
{
Body of the loop
}while(condition);
```

- In do..while loop the body is executed and then the condition is checked.
- If the condition is true the body is executed again and again, as long as thecondition is true.
- The control moves out of loop once, the condition become false.
- The body of the loop is executed at least once even if the condition is false firsttime.
- Flowchart:



Example: Print 1 to 10 numbers using do---while loop.

```
i =1;
do{
printf( "% d \n " , i );i =
        i+1;
    } while (i < = 10);
```

22. Give differences between While & Do-while loop.

While	Do--while
It is entry control loop, means first condition is checked, and if it is true thenbody of the loop executes, otherwise nextstatement after the loop will be executed.	It is exit control loop, first body is executedand then condition is checked and if the condition is true, then again body of the loop will be executed, otherwise next statement after the loop will be executed
At first time, condition is false, thenbody is not executed.	At first time, condition is false, at least oncethe body is executed.

Syntax:
while (condition)
{
Body of the loop;
}
	Syntax:
	Do
	{
	Body of the loop
	}while(condition);

Example:
i =1;
do
{
printf("% d \n " , i);i =
i+1;
} while (i < = 10);
	Example:
	i =1;
	do
	{
	printf("% d \n " , i);i =
	i+1;
	} while (i < = 10);

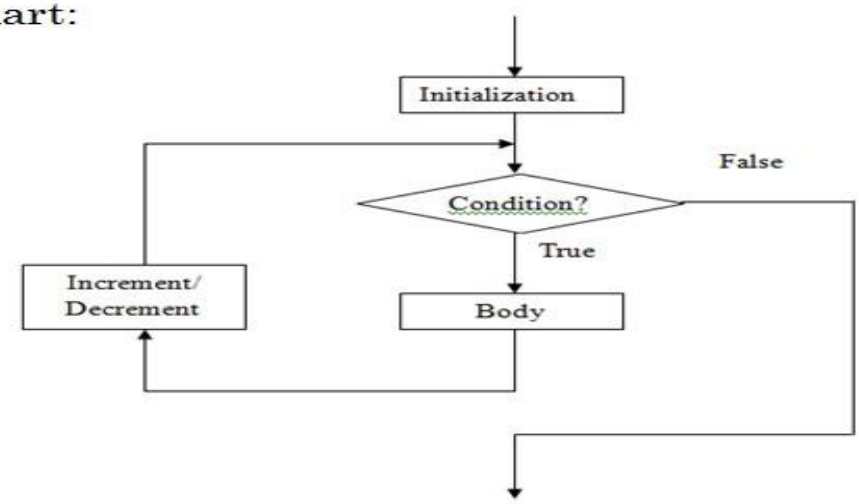
23. Explain for loop with example.

- **Syntax:**

```
For(Initialization ; Condition ; Increment or decrement)
{
        Body of the loop;
}
```

- **Initialization:** We require one variable which is used to control the loop, which is called as control variable. It executes only once. The control variable is initialized in the initialization expression.
- **Condition:** The condition statement checks the value of the control variable. Ifthe condition statement is true, the body of the loop is executed.
- **Increment/Decrement:** The increment/decrement of the control variable is donein this part. After the incrementing/decrementing the value of control variable, it is tested using condition if condition is true than again the body of loop is executed and this process is repeated until the condition become false.
- If the body of the loop contains only one statement, then { } is not required.

Flowchart:



Example: Print 1 to 10 numbers using for loop.

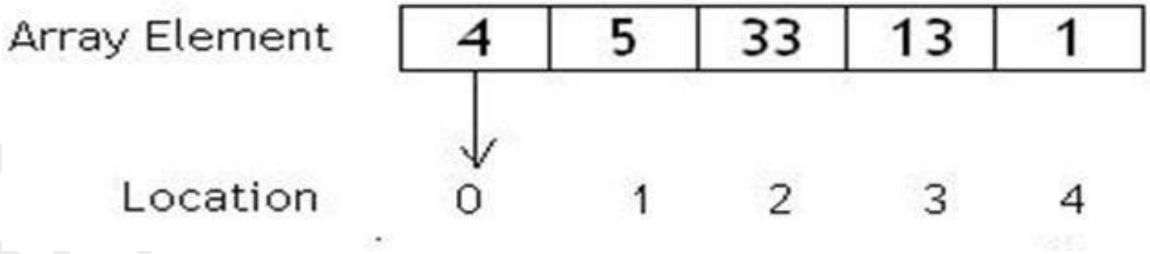
```
for(i = 1; i < =10 ; i++)
{
        printf(" %d  \n ", i ) ;
 }
```

24. Define Arrays with Syntax & example.

- An array is a collection of variables of same data type known by a same name.

Syntax:

- Data type arrayname [size] ;
- Data type: it defines the type of the array element, like int, float, char,double etc.
- Array name: it is the name of variable which represent array.
- Size: which is represents within[ ] symbol represent the size of the array.



The above array is declared as int a[5];a[0] =
4;
a[1] = 5;
a[2] = 33;
a[3] = 13;
a[4] = 1;
4,5,33,13,1 are actual data items … (in our case Roll numbers)
0,1,2,3,4 are index variables ..(similar to 'i' in for loop / Subscript variable)

25. How to Declaration Array.

- Array has to be declared before using it in C Program. Array is nothing but thecollection of elements of similar data types.
- **Syntax**
- <data_type> array_name [size1][size2] ..... [sizen];

Syntax Parameter	Significance
data_type	Data Type of Each Element of the array
Array_name	Valid variable name
size	Dimensions of the Array

Examples : Declaring 1D array

int roll[10]; // Array of 10 integer roll numbers

In the above example we are declaring the integer array of size 10. Array is singledimensional

26. Explain Advantage & disadvantage of Array.

Advantages of Array

1. We can use one name for similar objects and save them with same name butdifferent indexes.
2. 2- D array are used to represent matrices.
3. It can be used to implement other data structure like linked list, stack, queue ,tree, graph etc.

Disadvantages of Array

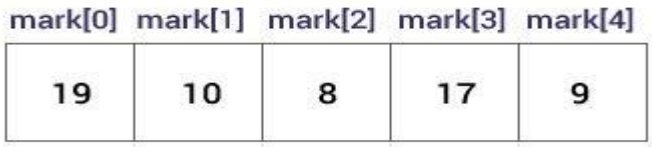
1. We must know in advance how many elements are to be stored in array.
2. If we allocate more memory than requirement than the memory space will bewasted.

27. Explain Accessing an array with example.

- Array elements are randomly accessed using the subscript variable.
- Array can be accessed using array-name and subscript variable written inside pairof square brackets [ ].

int mark[5] = {19, 10, 8, 17, 9};

- mark[0] is equal to 19
- mark[1] is equal to 10
- mark[2] is equal to 8
- mark[3] is equal to 17
- mark[4] is equal to 9



❖ Introduction to a String:

String:

A string is a sequence of zero or more characters followed by a NULL '\0' character.String is always terminating with NULL '\0' character.

Declaration and initialization of string:

Syntax:

char stringname[size];

where size is the number of characters.

Example:

char city[10] ;

Initialization of string variable:

(1) Compile time:

```
char city[9]= " NEW YORK";
```

(2) Run time:

```
char city[9];
scanf("%s", city);
OR
char city[9];
gets(y);
```

❖ Pointer

Introduction to Pointer:

Pointer is a variable which store the address of another variable.

Advantage of pointer:

- Pointer reduces the code and improves the performance.
- We can return multiple values from a function using the pointer.
- It makes you able to access any memory location in the computer's memory.

Characteristics of Pointers:

- Pointer is a variable which can hold the address of another variable.
- A pointer is a derived data type.
- It contains memory addresses as their values.
- If a C pointer is assigned to the null value, it points nothing.
- The asterisk symbol, * is used to retrieve the value of the variable
- & ampersand symbol is used for retrieving the address of a variable.

❖ Types of pointer :

void pointer

Syntax:
void *variable name

- Void pointer is a pointer which has no specified data type.
- It is also known as generic pointer.
- Void pointer can be pointed to any data type.
- When void pointer is declared two bytes of memory is assign to it.
- It is used when return type or parameter is unknown.
- void pointers that have addresses, can be further typecast into other data types very easily.

Example:

```
include<stdio.h
>void  main()
{
int a = 10;
void  *p;
p = &a;
printf("%d", *(int *)p);
}
```

NULL Pointer:

- A pointer that is not assigned any value but NULL is known as the NULL pointer.
- If you don't have any address to be specified in the pointer at the time of declaration, you canassign NULL value. It will provide a better approach.

int *p=NULL;

❖ Introduction to Functions:

Definition of Function: Function is a group of statement to perform a specific task.
Types of functions:
1. Library function
2. User-defined Function

❖ Types of Functions: Built-in and user defined Functions

Library function(Built-in):

- The function which is develop by system itself is known as Library function .
- Example: printf(),scanf(),sqrt(),ceil() etc …
- Library function is all ready exist in system.

User-defined Function:

- The function which is develop by user itself is known as User-defined function.
- Example: main()
- User-defined function is created by the user at the time of writing of program.

❖ Call by Value and call by Reference

Call By Value with example:

When a function is called using the value of variables, then it is known as call by value.

- The value of variables, which are to be passed, will be copied from the variables of the calling functions to the variables of the called functions.
- All the process done on the duplicate variables rather then actual variables.

Example:

```
#include <stdio.h>
void swap(int , int);      // function  prototype
int main()
{
   int a = 10;
   int b = 20;
   printf("Before swapping the values in main a = %d, b = %d\n",a,b);
   swap(a,b);    // function  call
   printf("After swapping values in main a = %d, b = %d\n",a,b); //actual parameters a=10, b=20
}
}
void swap (int a, int b)
{
   int temp;
   temp = a;
   a=b;
   b=temp;
   printf("After swapping values in function a = %d, b = %d\n",a,b); // Formal parameters, a=20, b=10
}
```

Call By Reference with example:

- When a function is called using the address of variables, then it is known as call by reference.
- Instead of passing the value of variables from calling function to the called function, addresses of the variables are passed.

- All the process done on the actual variables.

Example:

```
#include <stdio.h>
void swap(int *, int *);          // function  prototype
 void  main()
{
   int a = 10;
   int b = 20;
   printf("Before swapping the values in main a = %d, b = %d\n",a,b);
   swap(&a,&b);          // function  call
    printf("After swapping values in main a = %d, b = %d\n",a,b); // actual parameters a = 20, b = 10
}
void swap (int *a, int *b)
{
   int temp;
   temp = *a;
   *a=*b;
   *b=temp;
   printf("After swapping values in function a = %d, b = %d\n",*a,*b); // Formal parameters a=20, b=10
}
```

❖ Recursion

- Function call itself is called Recursion.
- Recursive call to a function may be conditional or unconditional.
- In conditional recursive call, function is terminated when condition is false.
- In unconditional recursive, function is called infinitely.
- It is used to find the factorial of a given number.
- Example :
- void main()
- {
- printf(“Recursion”);
- main();
- }
- Example :

```
void main()
        {
            ------------
             function1();
            ----------
        }
        function1()
        {
            ------------
             function1();

        }
```

Advantages of Recursion:

- The recursion is very flexible in data structure like stacks.
- Using recursion, the length of the program can be reduced.

Disadvantages of Recursion:

- It requires extra storage space.
- The recursion function is not efficient in execution speed and time.
- Proper termination is required; otherwise it leads to infinite loop.

❖ Storage Classes: -auto, static, register and extern

Storage Classes in C:

Storage classes in C are used to determine the lifetime, visibility, memory location, and initial value of avariable. There are four types of storage classes in C

- Automatic
- External
- Static
- Register

Storage Classes	Storag ePlace	Default Value	Scope	Lifetime
auto	RAM	Garbage Value	Local	Within function
extern	RAM	Zero	Global	Till the end of the main program Maybe declaredanywhere in the program

static	RAM	Zero	Local	Till the end of the main program, Retains valuebetween multiple functions call
register	Register	Garbage Value	Local	Within the function

❖ Introduction to Structures and Declaration, Initialization and accessing of Structures:

- Structure is a collection of logically related data items with different datatype.
- Structure must be defined first, that may be used later to declare structure.

Syntax:

Struct structure_name
{
Data type
member1;
data type
member2;

Data type member N;
} ;

- Struct is a keyword, it declares structure to hold data.
- Structure_Name is the name of structure(structure tag).
- member1,member2,…are members of structure.

Example:

Struct student
{
Char name[20];
int roll_no;
float per;
}s1;

Here **s1** is a structure variable.

Structure initialization:

Example:

struct student
{
char name[20];
int roll_no;
float per;
}s1={"Arav",20,85.5}**;**

❖ User-defined Datatypes: enum, typedef

Enumerated Data Type:

- Enumerated data type is user defined data type.
- Using enumerated data type we can create more than one symbolic constant at a time.

Syntax: **enum identifier{ value1, value2,value3};**

Here, enum is a keyword to declare enumerated data type. Identifier is a user defined enumerated data type.

Fo rExample:

```
#include<stdio.h>
enum week{Mon=1,Tue,Wed,Thur,Fri,Sat,Sun};
void main()
{
enum week day;
day = Wed;
printf("%d",day);
getch();
}
```

Output:2

Typedef:

The typedef is a keyword that is used in C programming to provide existing data types with a new name. typedef keyword is used to redefine the name already the existing name.

When names of data types become difficult to use in programs, typedef is used with user-defined data types.

Syntax:
typedef<existing_name><alias_name>

Example:
typede flong mylong

❖ Files

Introduction to text Files

File:

A file is a group of related data which is stored in a disk. File operations:

(1) Naming a file

(2) Opening a file

(3) Reading data from a file
(4) Writing data to a file
(5) Closing a file

fopen() and fclose():

fopen():

Create a new file or open an existing file.

Syntax:

```
FILE *fp;
fp=fopen("filename","mode");
```

Here, fp is a file pointer to the data type FILE.

Second statement open a file named file name and mode specify the opening of this file.

Example:

```
FILE *fp;
fp=fopen("E:\\abc.txt","r");
```

fclose():

Close the file

A file must be closed as soon as all operation on it have been completed.

Syntax:

```
fclose(filepointer);
```

Example:

```
FILE *fp;
fp=fopen("E:\\abc.txt","r");
      --------------
fclose();
```