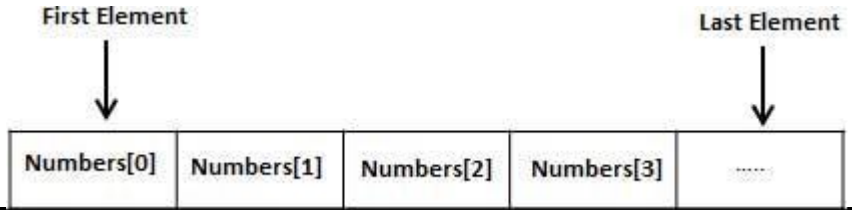
4. ARRAY

4.1 ARRAY BASICS

Array is a simple data structure which can store collection of elements of same data type. All elements are stored in the contiguous memory. An array is used to store a collection of data, but it is often more useful to think of an array as a collection of variables of the same type. Instead of declaring individual variables, such as number0, number1, ..., and number99, you declare one array variable such as numbers and use numbers[0], numbers[1], and ..., numbers[99] to represent individual variables. A specific element in an array is accessed by an index.

All arrays consist of contiguous memory locations. The lowest address corresponds to the first element and the highest address to the last element.



4.1.1 Declaring Arrays

To declare an array, a programmer specifies the type of the elements and the number of elements required by an array as follows:

type arrayName [arraySize];

This is called a single-dimensional array. The **arraySize** must be an integer constant greater than zero and **type** can be any valid C data type. For example, to declare a 10-element array called **balance** of type double, use this statement:

double balance[10];

Now balance is avariable array which is sufficient to hold up to 10 double numbers.

4.1.2 Initializing Arrays

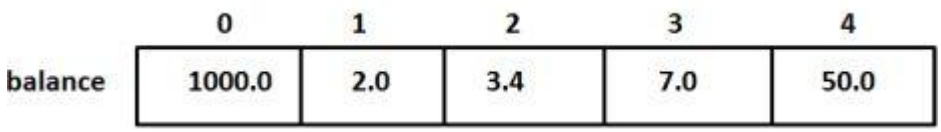
You can initialize array either one by one or using a single statement as follows:

double balance[5] = {1000.0, 2.0, 3.4, 17.0, 50.0};

The number of values between braces { } cannot be larger than the number of elements that we declare for the array between square brackets [ ]. Following is an example to assign a single element of the array:

If you omit the size of the array, an array just big enough to hold the initialization is created. Therefore, if you write:

double balance[] = {1000.0, 2.0, 3.4, 17.0, 50.0};



The above statement assigns element number 5th in the array a value of 50.0. Array with 4th index will be 5th ie. last element because all arrays have 0 as the index of their first element which is also called base index.

4.1.3Accessing Array Elements

An element is accessed by indexing the array name. This is done by placing the index of the element within square brackets after the name of the array. For example:

```
double salary = balance[9];
```

The above statement will take 10th element from the array and assign the value to salary variable.

e.g.

```
#include <stdio.h>

int main ()
{
   int n[ 10 ]; /* n is an array of 10 integers
   */  int i,j;


for ( i = 0; i < 10; i++ )  {
      n[ i ] = i + 100; /* set element at location i to i + 100
   */  }

for (j = 0;j < 10; j++ )  {
   printf("Element[%d] = %d\n", j, n[j] );
     }

 return 0;
}
```

When the above code is compiled and executed, it produces the following result:

```
Element[0] = 100
Element[1] = 101
Element[2] = 102
Element[3] = 103
Element[4] = 104
Element[5] = 105
```

Element[6] = 106
Element[7] = 107
Element[8] = 108
Element[9] = 109

4.2 MULTIDIMENSIONAL ARRAYS

C programming language allows multidimensional arrays. Here is the general form of a multidimensional array declaration:

type name[size1][size2]...[sizeN];

For example, the following declaration creates a three dimensional 5 . 10 . 4 integer array:

int threedim[5][10][4];

4.3 TWO-DIMENSIONAL ARRAYS

The simplest form of the multidimensional array is the two-dimensional array. A two-dimensional array is, in essence, a list of one-dimensional arrays. To declare a two-dimensional integer array of size x,y you would write something as follows:

type arrayName [x][y];

Where **type** can be any valid C data type and **arrayName** will be a valid C identifier. A two dimensional array can be think as a table which will have x number of rows and y number of columns. A 2-dimentional array **a** which contains three rows and four columns can be shown as below:

	Column 0	Column 1	Column 2	Column 3
Row 0	a[0][0]	a[0][1]	a[0][2]	a[0][3]
Row 1	a[1][0]	a[1][1]	a[1][2]	a[1][3]
Row 2	a[2][0]	a[2][1]	a[2][2]	a[2][3]

Thus, every element in array a is identified by an element name of the form **a[i][j]**, where a is the name of the array, and i and j are the subscripts that uniquely identify each element in a.

4.3.1 Initializing Two-Dimensional Arrays

Multidimensional arrays may be initialized by specifying bracketed values for each row. Following is an array with 3 rows and each row has 4 columns.

```
int a[3][4] = {
 {0, 1, 2, 3} ,   /*  initializers for row indexed by 0 */
 {4, 5, 6, 7} ,   /*  initializers for row indexed by 1 */
 {8, 9, 10, 11}   /*  initializers for row indexed by 2
 */
```

The nested braces, which indicate the intended row, are optional. The following initialization is equivalent to previous example:

```
int a[3][4] = {0,1,2,3,4,5,6,7,8,9,10,11};
```

4.3.2 Accessing Two-Dimensional Array Elements:

An element in 2-dimensional array is accessed by using the subscripts ie. row index and column index of the array. For example:

```
int val = a[2][3];
```

The above statement will take 4th element from the 3rd row of the array. You can verify it in the above diagram. Let us check below program where we have used nested loop to handle a two dimensional array:

```
#include <stdio.h>

int main ()
{
   /* an array with 5 rows and 2 columns*/
   int a[5][2] = { {0,0}, {1,2}, {2,4}, {3,6},{4,8}};
   int i, j;

for ( i = 0; i < 5; i++ ) {
for ( j = 0; j < 2; j++ ) {
printf("a[%d][%d] = %d\n", i,j, a[i][j] ); }
   }
return 0;
}
```

When the above code is compiled and executed, it produces following result:

```
a[0][0]: 0
a[0][1]: 0
a[1][0]: 1
a[1][1]: 2
a[2][0]: 2
a[2][1]: 4
a[3][0]: 3
a[3][1]: 6
a[4][0]: 4
a[4][1]: 8
```

ONE WORD ANSWERS

Q1) What is an array?

Ans: Array is collection of homogeneous data.It stores data in contigeous memory location.

Q2) How many types of arrays exist and what are they?

Ans: There are 3 types of arrays they are

a) one dimensional array
b) two dimensional array
c) multi dimensional array

Q3) What is an one dimensional array?

Ans: A list of items given in one variable name using only one subscript is known as one dimensional array.

Example: int a[3]

Q4) What is two dimensional array?

Ans: Collection of homogeneous elements in rows and columns is known as two-dimensional array.

Example: int matrix[3][3]

Q5) What is the general format of declaring an array?

Ans: datatype Arrayname[**size**]

note: the size must be an integer.

Q6) What is the value of starting index of an array?

Ans: zero

Q7) What is the use of const qualifier?

Ans: The compiler allows to associate the const qualifier with variables whose values will not be changed by the program.

Q8) What value is automatically assigned to those array elements that are not explicitly initialized?

Ans: All of array elements automatically set to zero except those that have been explicitly initialized with in array definitions.

Q9) State the rule that determines the order in which initial values are assigned to multi dimensional array elements?

Ans: the rule is that the last (right most) subscript increases most rapidly and the first (left most) increases least rapidly.

MULTIPLE CHOICE QUESTIONS

1. A array an example of ---?
   a. derived types
   b. fundamental types
   c. user-defined types

d. none of the above

2. Which of the following is not a data structure?
   a. linked list
   b. stack
   c. tree
   d. pointer

3. int a[n] will reserve how many locations in the memory?
   a. n b. n-1
   c. n+1 d. none of the above

4. Which of the following is the correct syntax for the initialisation of one dimensioal array?
   a. num[3]={0 0 0};
   b. num[3]={0,0,0};
   c. num[3]={0;0;0};
   d. num[3]=0

5. Which of the following is the correct syntax for initialisation of Two dimensional array?
   a.table[2][3]={0,0,0,1,1,1};
   b.table[2][3]={
   {0,0,0}
   {1,1,1}
   };
   c.table[2][3]={0,1},{0,1},{0,1};
   d.None of the above

6. Which of the following multi-dimensional array declaration is correct for realizing a 2x3 matrix?
   a. int m[2][3]
   b. int m[3][2]
   c. int m[3],m[2]
   d. None of the above

7. Which of the following is not the name of sorting technique?
   a.Bubble
   b.Selection
   c.Binary
   d.Insertion

8. Which of the following is not the name of a searching technique?
   a.Selection

b.Sequentia
l c.Binary
d.All of the above are searching techniques

9. Which is the last element of character string?
a. Last element of the string
b. Blank space
c. Null character
d. New line character

What will be output if you will execute following c code?

```
#include<stdio.h>
void
main() {
        char arr[11]="The African
        Queen"; printf("%s",arr);

}
```

2. What will be output if you will execute following c code? #include<stdio.h>

```
void
main() {
        char arr[20]="MysticRiver";
        printf("%d",sizeof(arr));
}
```

3. What will be output if you will execute following c code? #include<stdio.h>

```
void
main() {
        int const
        SIZE=5; int
        expr;
        double value[SIZE]={2.0,4.0,6.0,8.0,10.0};
        expr=1|2|3|4;
        printf("%f",value[expr])
; }
```

4. What will be output if you will execute following c code? #include<stdio.h>

```
#define var
3 void
main()  {
        char data[2][3][2]={0,1,2,3,4,5,6,7,8,9,10,11};
        printf("%o",data[0][2][1]);
```

5. What will be output if you will execute following c code?

```
#include<stdio.h>
void main()
{
int arr[][3]={{1,2},{3,4,5},{5}};
printf("%d %d %d",sizeof(arr),arr[0][2],arr[1][2]);
}
```

6. What will be output if you will execute following c code?

```
#include<stdio.h>
void main()
{
int xxx[10]={5};
printf("%d %d",xxx[1],xxx[9]);
}
```

7. What will be output if you will execute following c code?

```
#include<stdio.h>
#define WWW -1
enum {cat,rat};
void main()
{
        int Dhoni[]={2,'b',0x3,01001,'\x1d','\111',rat,WWW};
        int i;
        for(i=0;i<8;i++)
        printf(" %d",Dhoni[i]);
}
```

8. What will be output if you execute the following c code?

```
#include<stdio.h>
void main()
{
        long double a;
        signed char b;
        int arr[sizeof(!a+b)];
        printf("%d",sizeof(arr)
) }
```

SAMPLE C – PROGRAMS

1. Program to find whether a given year is leap year or not.

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

```
main()
{
        int year;

        printf("Enter a year:\n");
        scanf("%d", &year);

        if ( (year % 4) == 0)
                printf("%d is a leap year", year);
        else
                printf("%d is not a leap year\n", year);
}
```

Output:

Enter a year: 2000
2000 is a leap year

RUN2:

Enter a year: 1999
1999 is not a leap year

2. Program to multiply given number by 4 using bitwise operators.

```
# include <stdio.h>

main()
{
        long number, tempnum;
        printf("Enter an integer:\n");
        scanf("%ld", &number);
        tempnum = number;
        number = number << 2;   /*left shift by two bits*/

        printf("%ld x 4 = %ld\n", tempnum, number);
}
```

Output:

Enter an integer: 15
15 x 4 = 60

RUN2:
Enter an integer: 262

262 x 4 = 1048

3. Program to compute the value of X ^ N given X and N as inputs.

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

void main()
{
        long int x, n, xpown;
        long int power(int x, int n);

        printf("Enter the values of X and N: \n");
        scanf("%ld %ld", &x, &n);

        xpown = power (x, n);

        printf("X to the power N = %ld\n");
}

/*Recursive function to computer the X to power N*/

long int power(int x, int n)
{
        if (n==1)
                return(x);
         else if ( n%2 == 0)
                return (pow(power(x,n/2),2));                   /*if n is even*/
        else
                return (x*power(x, n-1));               /* if n is odd*/
}
```

Output:

Enter the values of X and N: 2 5
X to the power N = 32

RUN2:
Enter the values of X and N: 4 4
X to the power N ==256

RUN3:

```
Enter the values of X and N: 5 2
X to the power N = 25
```

RUN4:

```
Enter the values of X and N: 10 5
X to the power N = 100000
```

4. Program to swap the contents of two numbers using bitwise XOR operation. Don't use either the temporary variable or arithmetic operators.

```
# include <stdio.h>

main()
{
        long i, k;
        printf("Enter two integers: \n");
        scanf("%ld %ld", &i, &k);
        printf("\nBefore swapping i= %ld and k = %ld", i, k);
        i = i^k;
        k = i^k;
        i = i^k;
        printf("\nAfter swapping i= %ld and k = %ld", i, k);
}
```

Output:

```
Enter two integers: 23 34
Before swapping i= 23 and k = 34
After swapping i= 34 and k = 23
```

5. Program to find and output all the roots of a quadratic equation, for non-zero coefficients. In case of errors your program should report suitable error message.

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

main()
{
        float A, B, C, root1, root2;
        float realp, imagp, disc;
        clrscr();
```

```
        printf("Enter the values of A, B and C\n");
        scanf("%f %f %f", &A,&B,&C);

        if( A==0 || B==0 || C==0)
        {
                printf("Error: Roots cannot be determined\n");
                exit(1);
        }
        else
        {
                disc = B*B - 4.0*A*C;
                if(disc < 0)
                {
                        printf("Imaginary Roots\n");
                        realp = -B/(2.0*A) ;
                        imagp = sqrt(abs(disc))/(2.0*A);
                        printf("Root1 = %f  +i %f\n", realp, imagp);
                        printf("Root2 = %f  -i %f\n", realp, imagp);
                }
                else if(disc == 0)
                {
                        printf("Roots are real and equal\n");
                        root1 = -B/(2.0*A);
                        root2 = root1;
                        printf("Root1 = %f  \n",root1);
                        printf("Root2 = %f  \n",root2);
                }
                else if(disc > 0 )
                {
                        printf("Roots are real and distinct\n");
                        root1 =(-B+sqrt(disc))/(2.0*A);
                        root2 =(-B-sqrt(disc))/(2.0*A);
                        printf("Root1 = %f  \n",root1);
                        printf("Root2 = %f  \n",root2);
                }
        }

}
```

Output:

RUN 1

```
Enter the values of A, B and C: 3 2  1
Imaginary Roots
Root1 = -0.333333  +i 0.471405
Root2 = -0.333333  -i 0.471405
```

RUN 2
Enter the values of A, B and C: 1 2 1
Roots are real and equal
Root1 = -1.000000
Root2 = -1.000000

RUN 3
Enter the values of A, B and C: 3 5 2
Roots are real and distinct
Root1 = -0.666667
Root2 = -1.000000

6. Write a C programme to accept a list of data items & find the II largest & II smallest in it & take average of both & search for that value. Display appropriate message on successful search.

```
main ()
{
        int i,j,a,n,counter,ave,number[30];
        printf ("Enter the value of N\n");
        scanf ("%d", &n);
        printf ("Enter the numbers \n");
        for (i=0; i<n; ++i)
        scanf ("%d",&number[i]);
        for (i=0; i<n; ++i)
        {
                for (j=i+1; j<n; ++j)
                {
                        if (number[i] < number[j])
                        {
                                a = number[i];
                                number[i] = number[j];
                                number[j] = a;
                        }
                }
        }
        printf ("The numbers arranged in descending order are given below\n");
        for (i=0; i<n; ++i)
                printf ("%10d\n",number[i]);
        printf ("The 2nd largest number is  = %d\n", number[1]);
        printf ("The 2nd smallest number is = %d\n", number[n-2]);
        ave = (number[1] +number[n-2])/2;
        counter = 0;
        for (i=0; i<n; ++i)
```

```
        {
                if (ave==number[i])
                ++counter;
        }
        if (counter==0)
                printf("The average of 2nd largest & 2nd smallest is not in the array\n");
        else
                printf("The average of 2nd largest & 2nd smallest in array is %d in numbers\n",
                        counter);
 }
```

7. Write a C programme to arrange the given numbers in ascending order.

```
main ()
{
        int i,j,a,n,number[30];
        printf ("Enter the value of N\n");
        scanf ("%d", &n);
        printf ("Enter the numbers \n");
        for (i=0; i<n; ++i)
                scanf ("%d",&number[i]);
        for (i=0; i<n; ++i)
        {
                        for (j=i+1; j<n; ++j)
                        {
                                if (number[i] > number[j])
                                {
                                        a= number[i];
                                        number[i] = number[j];
                                        number[j] = a;
                                }
                        }
        }
        printf("Number in Asscending order:\n");
        for(i=0;i<n;i++)
                printf("\t%d\n",number[i]);
}
```

8. Program to convert the given binary number into decimal.

```
# include <stdio.h>
main()
{
        int   num, bnum, dec = 0, base = 1, rem ;
        printf("Enter the binary number(1s and 0s)\n");
```

```
        scanf("%d", &num);          /*maximum five digits */
        bnum = num;
        while( num > 0)
        {
                        rem = num % 10;
                        if(rem>1)
                        {
                                printf("\nError in input");
                                break;
                        }
                        dec = dec + rem * base;
                        num = num / 10 ;
                        base = base * 2;
        }
        if(num==0)
        {
                printf("The Binary number is = %d\n", bnum);
                printf("Its decimal equivalent is =%d\n", dec);
        }
}
```

9. Program to generate the fibonacci sequence.

```
#include <stdio.h>
main()
{
        int   fib1=0, fib2=1, fib3, limit, count=0;

        printf("Enter the limit to generate the fibonacci sequence\n");
        scanf("%d", &limit);

        printf("Fibonacci sequence is ...\n");
        printf("%d\n",fib1);
        printf("%d\n",fib2);
        count = 2;                      /* fib1 and fib2 are already used */

        while( count < limit)
        {
                fib3 = fib1 + fib2;
                count ++;
                printf("%d\n",fib3);
                fib1 = fib2;
                fib2 = fib3;
        }
}
```

10. Program to reverse the given integer (palindrome).

```
#include <stdio.h>
main()
{
        int   num, rev = 0, found = 0, temp, digit;

        printf("Enter the number\n");
        scanf("%d", &num);

        temp = num;
        while(num > 0)
        {
                digit = num % 10;
                rev = rev * 10 + digit;
                num /= 10;
        }
        printf("Given number =%d\n", temp);
        printf("Its reverse is =%d\n", rev);

        if(temp == rev )
                printf("Number is a palindrome\n");
        else
                printf("Number is not a palindrome\n");
}
```

11. Program to determine the given number is odd.

```
#include <stdio.h>

main()
{
        int numb;
        printf(" Enter the number\n");
        scanf("%d", &numb);

        if((numb%2)!=0)
                printf(" %d , is an odd number\n", numb);

}
```

12. Program to find the largest among three numbers.

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

```
main()
{
        int a, b, c;
        printf(" Enter  the values for A,B,C\n");
        scanf("%d %d %d", &a, &b, &c);
        if( a > b )
        {
                if ( a > c)
                        printf(" A is the Largest\n");
                else
                        printf("C is the largest\n");
        }
        else if ( b > c)
                printf(" B is the Largest\n");
        else
                printf("C is the Largest\n");
}
```

13. Program to find the areas of different geometrical figures using switch statement.

```
#include <stdio.h>
main()
{
        int  fig_code;
        float  side, base, length, bredth, height, area, radius;
        printf("-------------------------\n");
        printf(" 1 --> Circle\n");
        printf(" 2 --> Rectangle\n");
        printf(" 3 --> Triangle\n");
        printf(" 4 --> Square\n");
        printf("-------------------------\n");

        printf("Enter the Figure code\n");
        scanf("%d", &fig_code);

        switch(fig_code)
        {
                case 1:
                        printf(" Enter the radius\n");
                        scanf("%f",&radius);
                        area=3.142*radius*radius;
                        printf("Area of a circle=%f\n", area);
                        break;
                case 2:
                        printf(" Enter the bredth and length\n");
```

```
                    scanf("%f %f",&bredth, &length);
                    area=bredth *length;
                    printf("Area of a Reactangle=%f\n",
                    area);  break;
            case 3:
                    printf(" Enter the base and height\n");
                    scanf("%f %f", &base, &height);
                    area=0.5 *base*height;
                    printf("Area of a Triangle=%f\n", area);
                    break;
            case 4:
                    printf(" Enter the side\n");
                    scanf("%f", &side);
                    area=side * side;
                    printf("Area of a Square=%f\n", area);
                    break;
            default:
                    printf(" Error in figure code\n");
                    break;
        }

}
```

14. Program to find the factorial of a number.

```
#include <stdio.h>
main()
{
        int  i,fact=1,num;
        printf("Enter the number\n");
        scanf("%d",&num);
        if( num <0)
                printf("Factorial is not there for –ve numbers");
        else if(num==0 || num==1)
                fact=1;
        else
        {
                for(i=1;i<=num; i++)
                        fact *= i;
        }
        printf(" Factorial of %d =%5d\n", num,fact);
}
```

15. Program to illustrate for loop without initial and increment/decrement expressions.

```
#include <stdio.h>
main()
{
        int  i=0,limit=5;
        printf(" Values of I\n");
        for(  ; i<limit;   )
        {
                i++;
                printf("%d\n", i);
        }
}
```

16. Program to accept a string and find the sum of all digits in the string.

```
#include <stdio.h>
main()
{
        char string[80];
        int count, nc=0, sum=0;
        printf("Enter the string containing both digits and alphabet\n");
        scanf("%s", string);
        for(count=0; string[count]!='\0'; count++)
        {
                if((string[count]>='0') && (string[count]<='9'))
                {
                        nc += 1;
                        sum += (string[count] - '0');
                }
        }
        printf("NO. of Digits in the string= %d\n",nc);
        printf("Sum of all digits= %d\n",sum);

}
```

17. Program to find the sum of the sine series.

```
#include <stdio.h>
#include <math.h>
#define pi 3.142

main()
{
        int  i,n,k,sign;
        float sum=0,num,den,xdeg,xrad,xsqr,term;
        printf("Enter the angle( in degree): \n");
        scanf("%f",&xdeg);
```

```
        printf("Enter the no. of terms: \n");
        scanf("%d",&n);
        xrad=xdeg * (pi/180.0);    /* Degrees to radians*/
        xsqr= xrad*xrad;
        sign=1; k=2; num=xrad; den=1;

        for(i=1;i<=n; i++)
        {
                term=(num/den)* sign;
                sum += term;
                sign *= -1;
                num *= xsqr;
                den *= k*(k+1);
                k += 2;
        }
        printf("Sum of sine series of %d terms =%8.3f\n",n,sum);
}
```

18. Program to find the sum of cos(x) series.

```
#include<stdio.h>
#include<math.h>
main()
{
        float x, sign, cosx, fact;
        int n,x1,i,j;
        printf("Enter the number of the terms in a series\n");
        scanf("%d", &n);
        printf("Enter the value of x(in degrees)\n");
        scanf("%f", &x);
        x1=x;
        x=x*(3.142/180.0); /* Degrees to radians*/
        cosx=1;
        sign=-1;
        for(i=2; i<=n; i=i+2)
        {
                fact=1;
                for(j=1;j<=i;j++)
                {
                        fact=fact*j;
                }
                cosx=cosx+(pow(x,i)/fact)*sign;
                sign=sign*(-1);
        }
        printf("Sum of the cosine series=%f\n", cosx);
```

```
        printf("The value of cos(%d) using library function=%f\n",x1,cos(x));
}
```

19. Program to reverse the given integer.

```
#include <stdio.h>

main()
{
        int  num, rev = 0, found = 0, temp, digit;

        printf("Enter the number\n");
        scanf("%d", &num);

        temp = num;
        while(num > 0)
        {
                digit = num % 10;
                rev = rev * 10 + digit;
                num /= 10;
        }
        printf("Given number =%d\n", temp);
        printf("Its reverse is =%d\n", rev);
}
```

20. Program to accept a decimal number and convert to binary and count the number of 1's in the binary number.

```
#include <stdio.h>
main()
{
        long  num, dnum, bin = 0, base = 1;
        int rem, no_of_1s = 0 ;
        printf("Enter a decimal integer:\n");
        scanf("%ld", &num);                          /*maximum five digits */
        dnum = num;
        while( num > 0)
        {
                rem = num % 2;
                if (rem==1)                          /*To count number of 1s*/
                {
                        no_of_1s++;
                }
                bin = bin + rem * base;
```

```
                num = num / 2 ;
                base = base * 10;
        }
        printf("Input number is = %ld\n", dnum);
        printf("Its Binary equivalent is =%ld\n", bin);
        printf("No. of 1's in the binary number is = %d\n", no_of_1s);
}
```

Output:

```
Enter a decimal integer: 75
Input number is = 75
Its Binary equivalent is =1001011
No. of 1's in the binary number is = 4

RUN2
Enter a decimal integer: 128
Input number is = 128
Its Binary equivalent is=10000000
No. of 1's in the binary number is = 1
```

21. Program to find the number of characters, words and lines.

```
#include<conio.h>
#include<string.h>
#include<stdio.h>
void main()
{
        int count=0,chars,words=0,lines,i;
        char text[1000];
        clrscr();
        puts("Enter text:");
        gets(text);
        while (text[count]!='\0')
                count++;
        chars=count;
        for (i=0;i<=count;i++)
        {
                if ((text[i]==' '&&text[i+1]!=' ')||text[i]=='\0')
                  words++;
        }
        lines=chars/80+1;
        printf("no. of characters: %d\n", chars);
```

```
        printf("no. of words: %d\n", words);
        printf("no. of lines: %d\n", lines);
        getch();
}
```

22. Program to find the GCD and LCM of two integers output the results along with the given integers. Use Euclids' algorithm.

```
#include <stdio.h>
main()
{
        int num1, num2, gcd, lcm, remainder, numerator, denominator;
        clrscr();
        printf("Enter two numbers: \n");
        scanf("%d %d", &num1,&num2);
        if (num1 > num2)
        {
                numerator = num1;
                denominator = num2;
        }
        else
        {
                numerator = num2;
                denominator = num1;
        }
        remainder = numerator % denominator;
        while(remainder !=0)
        {
                numerator   = denominator;
                denominator = remainder;
                remainder   = numerator % denominator;
        }
        gcd = denominator;
        lcm = num1 * num2 / gcd;
        printf("GCD of %d and %d = %d \n", num1,num2,gcd);
        printf("LCM of %d and %d = %d \n", num1,num2,lcm);
}
```

Output:

```
Enter two numbers: 5 15
GCD of 5 and 15 = 5
LCM of 5 and 15 = 15
```

23. Program to find the sum of odd numbers and sum of even numbers from 1 to N. Output the computed sums on two different lines with suitable headings.

```
#include <stdio.h>
main()
{
        int i, N, oddsum = 0, evensum = 0;
        printf("Enter the value of N: \n");
        scanf ("%d", &N);
        for (i=1; i <=N; i++)
        {
                if (i % 2 == 0)
                        evensum = evensum + i;
                else
                        oddsum = oddsum + i;
        }
        printf ("Sum of all odd numbers  = %d\n", oddsum);
        printf ("Sum of all even numbers = %d\n", evensum);
}
```

Output:

RUN1
Enter the value of N: 10
Sum of all odd numbers = 25
Sum of all even numbers = 30

RUN2
Enter the value of N: 50
Sum of all odd numbers = 625
Sum of all even numbers = 650

24. Program to check whether a given number is prime or not and output the given number with suitable message.

```
#include <stdio.h>
#include <stdlib.h>
main()
{
        int num, j, flag;
        clrscr();

        printf("Enter a number: \n");
        scanf("%d", &num);
        if ( num <= 1)
        {
                printf("%d is not a prime numbers\n", num);
```

```
                exit(1);
        }
        flag = 0;
        for (j=2; j<= num/2; j++)
        {
                if( ( num % j ) == 0)
                {
                        flag = 1;
                        break;
                }
        }
        if(flag == 0)
                printf("%d is a prime number\n",num);
        else
                printf("%d is not a prime number\n", num);
}
```

Output:

RUN 1
Enter a number: 34
34 is not a prime number

RUN 2
Enter a number: 29
29 is a prime number

25. Program to generate and print prime numbers in a given range. Also print the number of prime numbers.

```
#include <stdio.h>
#include <math.h>
main()
{
        int M, N, i, j, flag, temp, count = 0;
        clrscr();
        printf("Enter the value of M and N: \n");
        scanf("%d %d", &M,&N);
        if(N < 2)
        {
                printf("There are no primes upto %d\n", N);
                exit(0);
        }
        printf("Prime numbers are\n");
        temp = M;
        if ( M % 2 == 0)
```

```
        {
                M++;
        }
        for (i=M; i<=N; i=i+2)
        {
                flag = 0;
                for (j=2; j<=i/2; j++)
                {
                        if( (i%j) == 0)
                        {
                                flag = 1;
                                break;
                        }
                }
                if(flag == 0)
                {
                        printf("%d\n",i);
                        count++;
                }
        }
        printf("Number of primes between %d  and %d = %d\n",temp,N,count);
}
```

Output:

```
Enter the value of M and N:  15 45
Prime numbers are
17
19
23
29
31
37
41
43
Number of primes between  15 and 45 =  8
```

26. Write to accept a 1-dimensional array of N elements & split into 2 halves & sort 1st half in ascending order & 2nd into descending order.

```
#include<stdio.h>
main ()
{
        int i,j,a,n,b,number[30];
        printf ("Enter the value of N\n");
```

```
        scanf ("%d", &n);
        b = n/2;
        printf ("Enter the numbers \n");
        for (i=0; i<n; ++i)
                scanf ("%d",&number[i]);
        for (i=0; i<b; ++i)
        {
                for (j=i+1; j<b; ++j)
                {
                        if (number[i] > number[j])
                        {
                                a = number[i];
                                number[i] = number[j];
                                number[j] = a;
                        }
                }
        }
        for (i=b; i<n; ++i)
        {
                        for (j=i+1; j<n; ++j)
                        {
                                if (number[i] < number[j])
                                {
                                        a = number[i];
                                        number[i] = number[j];
                                        number[j] = a;
                                }
                        }
         }
        printf (" The 1st half numbers\n");
        printf (" arranged in asc\n");
        for (i=0; i<b; ++i)
                printf ("%d ",number[i]);
        printf("\nThe 2nd half Numbers\n");
        printf("order arranged in desc.order\n");
        for(i=b;i<n;i++)
                printf("%d ",number[i]);
}
```

27. Program to delete the desired element from the list.

```
# include <stdio.h>
main()
{
        int  vectx[10];
```

```
        int  i, n, found = 0, pos, element;
        printf("Enter how many elements\n");
        scanf("%d", &n);
        printf("Enter the elements\n");
        for(i=0; i<n; i++)
        {
                scanf("%d", &vectx[i]);
        }
        printf("Input array elements are\n");
        for(i=0; i<n; i++)
        {
                printf("%d\n", vectx[i]);
        }
         printf("Enter the element to be deleted\n");
         scanf("%d",&element);
        for(i=0; i<n; i++)
        {
                if ( vectx[i] == element)
                {
                        found = 1;
                        pos = i;
                        break;
                }
        }
        if (found == 1)
        {
                for(i=pos; i< n-1; i++)
                {
                        vectx[i] = vectx[i+1];
                }
                printf("The resultant vector is \n");
                for(i=0; i<n-1; i++)
                {
                        printf("%d\n",vectx[i]);
                }
        }
        else
                printf("Element %d is not found in the vector\n", element);
}
```

28. Write a "C" program to Interchange the main diagonal elements with the scondary diagonal elements.

```
#include<stdio.h>
main ()
```

```
{
        int i,j,m,n,a;
        static int ma[10][10];
        printf("Enetr the order of the matix \n");
        scanf("%dx%d",&m,&n);
        if(m==n)
        {
                printf("Enter the co-efficients of the matrix\n");
                for(i=0;i<m;++i)
                {
                        for(j=0;j<n;++j)
                        {
                                scanf("%d",&ma[i][j]);
                        }
                }
                printf("The given matrix is \n");
                for(i=0;i<m;++i)
                {
                        for(j=0;j<n;++j)
                        {
                                printf(" %d",ma[i][j]);
                        }
                        printf("\n");
                }
                for(i=0;i<m;++i)
                {
                        a = ma[i][i];
                        ma[i][i]   = ma[i][m-i-1];
                        ma[i][m-i-1] = a;
                }
                printf("THe matrix after changing the \n");
                printf("main diagonal & secondary diagonal\n");
                for(i=0;i<m;++i)
                {
                        for(j=0;j<n;++j)
                        {
                                printf(" %d",ma[i][j]);
                        }
                        printf("\n");
                }
        }
        else
                printf("The given order is not square matrix\n");
}
```

29 Program to insert an element at an appropriate position in an array.

```
#include <stdio.h>
#include <conio.h>
main()
{
        int  x[10];
        int  i, j, n, m, temp, key, pos;
        clrscr();
        printf("Enter how many elements\n");
        scanf("%d", &n);
        printf("Enter the elements\n");
        for(i=0; i<n; i++)
        {
                scanf("%d", &x[i]);
        }
        printf("Input array elements are\n");
        for(i=0; i<n; i++)
        {
                printf("%d\n", x[i]);
        }
        for(i=0; i< n; i++)
        {
                for(j=i+1;j<n; j++)
                {
                        if (x[i] > x[j])
                        {
                                temp = x[i];
                                x[i] = x[j];
                                x[j] = temp;
                        }
                }
        }
        printf("Sorted list is:\n");
        for(i=0; i<n; i++)
        {
                printf("%d\n", x[i]);
        }
        printf("Enter the element to be inserted\n");
        scanf("%d",&key);
        for(i=0; i<n; i++)
        {
                if ( key < x[i] )
                {
                        pos = i;
                        break;
```

```
                }
        }
        m= n - pos + 1 ;
        for(i=0; i<= m ; i++)
        {
                x[n-i+2] = x[n-i+1] ;
        }
        x[pos] = key;

        printf("Final list is:\n");
        for(i=0; i<n+1; i++)
        {
                printf("%d\n", x[i]);
        }
}
```

30. Program to compute mean, varience and standard deviation.

```
main()
{
                float x[10];
                int  i, n;
                float avrg, var, SD, sum=0, sum1=0;

                printf("Enter how many elements\n");
                scanf("%d", &n);
                printf("Enter %d numbers:",n);
                for(i=0; i<n; i++)
                {
                        scanf("%f", &x[i]);
                }
                                /* Compute the sum of all elements */
                for(i=0; i<n; i++)
                        sum = sum + x[i];

                avrg = sum /(float) n;
                                /* Compute varaience and standard deviation  */
                for(i=0; i<n; i++)
                {
                        sum1 = sum1 + pow((x[i] - avrg),2);
                }
                var = sum1 / (float) n;
                SD = sqrt(var);
                printf("Average of all elements =%.2f\n", avrg);
                printf("Variance of all elements =%.2f\n", avrg);
```

```
        printf("Standard Deviation of all elements =%.2f\n", avrg);
}
```

6.1 INTRODUCTION TO STRINGS

A string in the C language is simply an array of characters. Strings must have a NULL or \0 character after the last character to show where the string ends. A string can be declared as a character array or with a string pointer. The string can be declared as follow :

Syntax:

```
char string_nm[size];
```

Example:

```
char name[50];
```

When compiler assigns string to character array then it automatically supplies **null character ('\0')** at the end of string. Thus, size of string = original length of string + 1.

```
char name[7];
name =
```

'T'	'E'	'C'	'H'	'N'	'O'	'\o'
1	2	3	4	5	6	7

Take a look at this example:

```
#include<stdio.h>
int main()
{
        char str[20];
        str[0] = 'H';
        str[1] = 'E';
        str[2] = 'L';
        str[3] = 'L';
        str[4] = 'O';
        str[5] = '\n';
        str[6] = '\0';
        printf("%s",
        str);  return 0;
}
```

Note: %s is used to print a string.

String pointers are declared as a pointer to a char. When there is a value assigned to the string pointer the NULL is put at the end automatically.

Ex:

```
#include<stdio.h>
 int main(){
char *ptr_str;
ptr_str = "HELLO";
printf("%s\n", ptr_str);  return 0;}
```

6.2 READING AND WRITING STRINGS

6.2.1 Read Strings

To read a string, we can use scanf() function with format specifier %s.

```
char name[50];
   scanf("%s",name);
```

The above format allows accepting only string which does not have any blank space, tab, new

6.2.2 Write Strings

To write a string, we can use printf() function with format specifier %s.

```
char name[50];
scanf("%s",name); printf("%s",name);
```

6.2.3 Array of strings

To create an array of strings, a two dimensional character array is used with the size of the left- Index determining the number of strings and the size of the right Index specifying the maximum length of each string.

For example, to declare an array of 30 strings each having a max length of 80 characters.

```
char str_array[30][80];
```

6.3 STRING HANDLING FUNCTIONS

'string.h' is a header file which includes the declarations, functions, constants of string handling utilities. These string functions are widely used today by many programmers to deal with string operations.

Comparisonfunctions

memcmp	Compare Memory Blocks
strcmp	String Compare
Strcoll	String Compare Using Locale-Specific Collating Sequence
strncmp	Bounded String Compare
strxfrm	Transform Locale-Specific String

Concatentationfunctions

strcat	String Concatenation
Strncat	Bounded String Concatenation

Copyingfunctions

memcpy	Copy Memory Block
memmove	Copy Memory Block
strcpy	String Copy
strncpy	Bounded String Copy

Searchfunctions

memchr	Search Memory Block for Character
strchr	Search String for Character
Strcspn	Search String for Intial Span of Characters Not in Set
strpbrk	Search String for One of a Set of Characters
strrchr	Search String in Reverse for Character
strspn	Search String for Initial Span of Characters in Set
strstr	Search String for Substring
strtok	Search String for Token

Miscellaneousfunctions

memset	Initialize Memory Block
strerror	Convert Error Number to String
strlen	String Length

```
#include <stdio.h>
#include <conio.h>
#include <string.h>
void main()
 {
        char str[50];
        clrscr();
printf("\n\t Enter your name : ");
gets(str);
printf("\nLower case of string: %s",strlwr(str));
printf("\nUpper case of string: %s",strupr(str));
printf("\nReverse of string: %s",strrev(str));
printf("\nLength of String: %d",strlen(str));      getch();
 }
```

Output
:

```
Enter your name : akanksha
Lower case of string: akanksha
Upper case of string: AKANKSHA
Reverse of string: AHSKNAKA
Length of String: 8
```

String Manipulation Functions

1. char *strcpy (char *dest, char *src) - Copy src string string into dest string.
2. char *strncpy(char *string1, char *string2, int n) - Copy first n characters of string2 to stringl .
3. int strcmp(char *string1, char *string2) - Compare string1 and string2 to determine alphabetic order.
4. int strncmp(char *string1, char *string2, int n) - Compare first n characters of two strings.
5. int strlen(char *string) - Determine the length of a string.
6. char *strcat(char *dest, const char *src); - Concatenate string src to the string dest.
7. char *strncat(char *dest, const char *src, int n); - Concatenate n chracters from string src to the string dest.
8. char *strchr(char *string, int c)- Find first occurrence of character c in string.
9. char *strrchr(char *string, int c) - Find last occurrence of character c in string.
10. char *strstr(char *string2, char string*1) - Find first occurrence of string string1 in string2.
11. char *strtok(char *s, const char *delim)- Parse the string s into tokens using delim as delimiter.

String functions examples

1) int strlen(char array): This function accepts string as parameter and return integer i.e the length of String passed to it.

Example

```
#include <stdio.h>
#include <string.h>
void main(void)
 {
```

```
  char
  string[]="hello"; int
  len;
  len=strlen(string);
  printf("length of %s is %d\t",string,len);
}
```
Output::length of hello is 5.

2) strcpy (Destination string,source string): This function accepts 2 strings as parameter,1st one is destination string and 2nd is source string. This function copies source string to destination string.

Example

```
#include <stdio.h>
#include <string.h>
void main(void)
{
  char
  src[]="hello",dest[15];
  strcpy(dest,src);
printf("%s is copied to dest string\t",dest);
}
```

3) strcat (Destination string,source string): This function accepts two,strings source string is appended to the destination string.

Example

```
#include <stdio.h>
#include <string.h>
void main(void)
{
  char
  src[]="sea",dest[]="programming";
  strcat(dest,src);
printf("concatenated string is %s",dest);
}
```

4) strrev (string): This function accepts single string as parameter and reverse that string. **Example**

```
#include <stdio.h>
#include <string.h>
void main(void)
{
  char
  string[]="sea";
  strrev(string);
printf("reverse string is %s",string);
}
```

5)int strcmp (string 1, string2): This function compares two strings passed as parameters and returns either +ve number,0,-ve number.+ve value indicates string1 > string2. 0 indicates string1 and string2 are equal -ve value indicates string1 < string2.

Example

```
#include <stdio.h>
#include <string.h>
void main(void)
{
   Char
   string1[]="sea",string2[]="programming";
   int cmp;
cmp=strcmp(string1,string2);
   if(cmp>0)
   printf("%s > %s",string1,string2);
     else
   {
   if(cmp<0
   )
   printf("%s < %s",string1,string2);
   else
printf("%s = %s",string1,string2);
  }
}Output: sea > programming.
```

.this is because alphabetically p comes first then s. so the string compare function returns difference between ASCII of s and p which would be +ve.

6) strcmpi (string 1, string2): This function is similar to strcmp().The onlyy difference is that it ignores the case . Example SparK and spark both are same.

Example

```
#include <stdio.h>
#include <string.h>
void main(void)
{
   char
   string1[]="spark",string2[]="SPArk";  int
   cmp;
   cmp=strcmpi(string1,string2)
   ;  if(cmp>0)
      printf("%s >
   %s",string1,string2);  else
   {
   if(cmp<0
   )
      printf("%s <
   %s",string1,string2);  else
 printf("%s =
  %s",string1,string2);  }
```

7) strlwr (string)::This function accepts single string that can be in any case(lower or upper).It converts the string in lower case.

Example

```
#include <stdio.h>
#include <string.h>
void main(void)
{
   char string1[]="Hello";
```

```
   strlwr(string1);
   printf("%s is in lower
case",string1);  }
```
Output: hello is in lower case.

8) strupr (string)::This function accepts single string that can be in any case(lower or upper).It converts the string in upper case.

Example

```
#include <stdio.h>
#include <string.h>
void main(void)
{
   char string1[]="hello";
   strupr(string1);
   printf("%s is in upper case",string1);

}
```
Output: HELLO is in upper case.

9)char* strstr (main string,substring): This function accepts two strings i.e main string and substring.

It searches for the first occurrence substring in main string and returns the character pointer to the first char.

Example

```
#include <stdio.h>
#include <string.h>
void main()
{
      char str1[]="programmingsea",str2[]="ming",*ptr;
      ptr=strstr(str1, str2);
      printf("substring is: %s",ptr);
      getch();
}
```
Output : substring is mingsea

C program to count the number of vowels and consonants in a string

Here is the program to count number of vowels and consonants in c.

```
#include<stdio.h>
#include<conio.h>
#include<string.h
>  void main()
{
   char str[50];
   int vovels=0,consonants=0,len,i;
   printf("Enter the String:\n");
   gets(str);
   len=strlen(str);
for(i=0;i<len;i++)  {
if((str[i]>64&&str[i]<91)||(str[i]>96&&str[i]<123)) //Firstly checking the character is alphabet or not
{
```

```
if(str[i]=='a'||str[i]=='A'||str[i]=='e'||str[i]=='E'||str[i]=='i'||str[i]=='I'||str[i]=='o'||str[i]=='O'||str[i]=='u'|
|str[i]=='U')          //checking if it is vowel or not
   vovels++;
     else
consonants++;
  }
    }
printf("Number of vovels = %d and consonants = %d ",vowels, consonants);
getch();
 }
```

Output:

```
Enter the String:
Welcome to C
Number of vovels = 4 and consonants =
6
```

C program to count number of words,spaces,characters,digits and special symbols

```
#include<stdio.h>
#include<conio.h>
void main()
{
       int i=0,chr=0,sp=0,words=1,ss=0,digits=0;
      char line[100],temp=32;
      //clrscr();
printf("\nEnter the line:\n");
gets(line);

      while(line[i]!='\0')                          //to check for string termination
      {
      if((line[i]>64&&line[i]<91)||(line[i]>96&&line[i]<123))          // ascii range of
 characters
            chr++;
            else
           {if(line[i]==32)                                    //ascii value of  space is 32
                 {
                       sp++;
                       if(temp!=32)
                       words++;
                 }
                else
                {
                       if(line[i]>47&&line[i]<58)                    //ascii range of digits
                          digits++;
                        else
                        ss++;
                }
            }
           temp=line[i];
           i++;
     }
    printf("\nNumber of characters = %d words = %d spaces %d special symbols = %d digits =
```

%d",chr,words,sp,ss,digits);
}
Output:
Enter the line:
Hello3847 Wor#4@ld

Number of characters = 10 words = 2 spaces 1 special symbols = 2 digits = 5

ONE WORD ANSWERS

Q1. What is a string?
Ans: Sstring is a collection of characters.

Q2) Which is the last character of a string ?
Ans: Null character is the last char of a string

Q3).
Ans: Write the general form of the declaration of a simple string? char string_name[size];

Q4)
Ans: Other than the scanf function,what is the other function that can be used to read the string? gets() is the other function that can be used to read the string.

Q5)
Ans: What is the funtcion we use to write the string on the screen? printf() function by using format code %s.

Q6)
Ans: What is getchar()?
it reads a charactor from the keyboard.

Q7)
Ans: What is meant by concatenation?
The process of combining one string with another is known as string concatneation.

Q8)
Ans: What is meant by strcmp?
The process of comparing the one string into another is known as strcmp.

Q9)
Ans: Which header file is used for string operations? string.h .

Q10)
Ans: What is meant by strncmp?
This is a three parametre function that compares only 'n' characters of both the given strings.
the general format of the function

MULTIPLE CHOICE QUESTIONS

1. Which of the following is used to represent the end of a string?
   a. blank space
   b. null character
   c. newline character
   d. last element of the string

2. Which of the following is used to display a string on I/O console?
   a. %s b. %d c. %c d. %f

3. Which of the following is true for getchar?

a. read a string of characters
b. read a character
c. read the characters until \n is encountered
d. none of the above

4. What is the value of x in the following character arithmetic expression? X='A'-2;
a. 63
b. 64
c. 65
d. 66

5. Which function is used to determine the length of a string?
a. strcmp
b. strcpy
c. strlen
d. strcat

6. What value will strlen functions return for the string {'r','a','m','\0'} a. 3
b. 4
c. 5
d. None of the above

7. Which of the following is the correct syntax for copying a string S1 into S2?
a. strcmp(S2,S1);
b. strcpy(S1,S2);
c. strcmp(S1,S2);
d. strcpy(S2,S1);