ARRAYS :

Definition : "Array is a collection of Homogeneous data item called an element of the array, in a continuous memory location with a single name."

→ The data type of the elements may be any valid data type like char, int, or float.

→ The elements of the array shares the same variable name but each element has a different index number known as subscript.

→ Array can be single-dimensional or multidimensional. The number of subscripts determines the dimension of array.

→ A one-dimensional array has one subscript, two dimensional array has two subscripts and so on.

→ The one-dimensional arrays are known as <u>vectors</u> and two dimensional arrays are known as <u>matrices</u>.

One - Dimensional Array :

Declaration of 1-D Array :

The syntax for declaration of a 1-D array is;

```
data-type array-name[size];
```

Here, array-name denotes the name of the array and it can be any valid C identifier, data-type is the data type of the elements of array. The size of the array specifies the number of elements that can be stored in the array. It may be a positive integer constant or constant integer expression.

Example :
```
int age[100];
float sal[15];
char grade[50]; etc.
```

Here, age is an integer type array, which can store 100 elements of integer type. The array sal is a floating type array of size 15, can hold float values and grade is a char type array, which can store 50 elements of char type.

→ The Symbolic Constants can be used to specify the array size. For example;

```
#define SIZE 10
void main()
{
    int mark[SIZE];
    float arr[SIZE];
    ----------
    ----------
}
```

→ The use of symbolic constant to specify the size of array makes it convenient to modify the program if the size of array is to be changed latter, because the size has to be changed only at one place i.e. in the #define directive.

Initialization of 1-D Array :

→ The general form of initializing an array is;

```
data-type array-name[size] = { list of values };
```

→ Example :
```
int num[4] = {101, 102, 103, 104};
float a[3] = {4.15, 5.32, 1.79};
```

```
char name[4] = {'\n', '\t', '\n', '\f'};
```

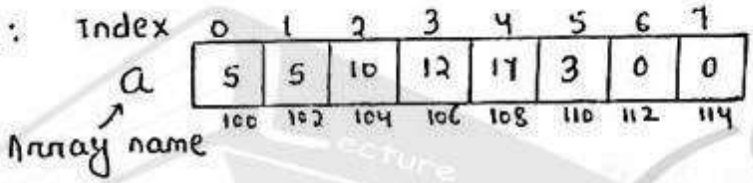
* Array can be initialized by listing the values in {} braces.

If an array will be initialized at the time of the declaration, the it will be of <u>static kind</u>.

* values not initialized are set to zero.

* when an array is initialized, the dimensional value may be omitted. The number of elements in an array will be determined by the compiler.

<u>Example</u>:




```
int a[8] = { 5, 5, 10, 12, 17 }
printf ("%d", a[2]);   output = 10
printf ("%d", a[0]);   output = 5
printf ("%d", a[7]);   output = 0
printf ("%d", a);      output = 100 (address of 1st element)
```

Name of the array holds the starting address of the block where array elements are started.

```
for (i=0 ; i<5 ; i++)
    printf ("%d", a[i]);   output : 5 5 10 12 17
```

Processing 1-D Array :

→ ~~Suppose~~ For processing an array, we generally use a for loop and the loop variable is used at the place of subscript.

→ The initial value of loop variable is taken 0 since array subscripts starts from 0.

→ The loop variable is increased by one each time, so that we can access and process the next element in the array.

→ The total no. of passes in the loop will be equal to the number of elements in the array and in each pass we will process one element.

→ Suppose arr[10] is an array of int type -

(i) Reading values in arr[10]

```
for (i=0 ; i<10 ; i++)
    scanf ("%d", &arr[i]);
```

(ii) Displaying values in arr[10]

```
for (i=0 ; i<10 ; i++)
    printf ("%d", arr[i]);
```

In 1-D array address of the position of the element can be found out by the formula ;

a + i * (size of data-type)

Example : a = 100 int a[4] = {2, 5, 7, 8} ;

int i = 2

address = 100 + 2 * (2)

= 104

	0	1	2	3
a	2	5	7	8
	100	102	104	106

/* write a program to input the values in an array & display it */

```
# include <stdio.h>
# include <conio.h>
Void main()
{
   int arr[5], i;
   Clrscr();
   for(i=0; i<5; i++)
      { printf("Enter the elements for arr[%d]:\n", i);
        scanf("%d", &arr[i]);
      }
   printf("The array elements are:\n");
   for(i=0; i<5; i++)
      { printf("%d\t", arr[i]);
      }
   getch();
}
```

output:

```
Enter the elements for arr[0] : 2
Enter the elements for arr[1] : 5
Enter the elements for arr[2] : 7
Enter the elements for arr[3] : 9
Enter the elements for arr[4] : 10

The array elements are :

2  5  7  9  10
```

/* write a program to find the sum of the elements in an array */

```
# include <stdio.h>
# include <conio.h>
void main()
{
   int arr[5], i, sum = 0;
   clrscr();
   for (i=0; i<5; i++)
       {
          printf ("Enter the elements of an array: \n");
          scanf ("%d", &arr[i]);
             sum = sum + arr[i];
       }
  printf ("Sum of elements in an array is : %d\n", sum);
  getch();
}
```

output : Enter the elements of an array:

5
7
8
10
12

Sum of elements of an array is : 42

/* write a program to count the number of even and odd elements in an array */

```
# include <stdio.h>
# include <conio.h>
# define SIZE 5
void main()
{
   int i, arr[SIZE], even = 0, odd = 0;
```

```
clrscr();
for( i=0 ; i< size ; i++)
    {
     printf(" Enter the elements of an array :\n");
      scanf(" %d", &arr[i]);
    if ( arr[i] % 2 == 0)
          { even++;
          }
        else {
               odd++ ;
             }
        }
 printf(" even no.s = %d  odd no.s = %d\n", even, odd);
 getch();
}
```

output : Enter the elements of an array :

2
7
10
15
17

even no.s = 2 odd no.s = 3

```
/* write a program to reverse the elements of an array */
# include <stdio.h>
# include <conio.h>
void main()
{
   int i, j, arr[10] = {1,2,3,5,7,9,12,4,10,51}, temp;
   clrscr();
   for( i=0, j=9 ; i<j ; i++, j--) ;
        {
```

```
    temp = a[i];
    a[i] = a[j];
    a[j] = temp;
  }
  printf("The reverse elements of an array are :\n");
    for(i=0; i<10; i++)
        { printf("%d\t", arr[i]);
        }
  getch();
}
```

Output : The reverse elements of an array are :

51 10 4 12 9 7 5 3 2 1

(or)

```
#include <stdio.h>
#include <conio.h>
void main()
{
   int a[100], n, i;
    clrscr();
   printf("Enter the no. of numbers in an array :\n");
    scanf("%d", &n);
   printf("Enter the numbers in an array :\n");
       for(i=0; i<n; i++)
           scanf("%d", &a[i]);
   printf("the array elements are :\n");
       for(i=0; i<n; i++)
           printf("%d", a[i]);
```

```
    printf(" The array in reverse order is: \n");
      for( i=n-1 ; i>=0 ; i--)
          printf("%d\t", a[i]);
    getch();
}
```

Output: Enter the no. of numbers in an array:
4
Enter the numbers in an array:
5 9 10 12
The array elements are:
5 9 10 12
The array in reverse order is:
12 10 9 5

/* write a program to search an item in an array */
(Linear Search)

```
# include <stdio.h>
# include <conio.h>
void main()
{
   int i, item, arr[50], flag=0, n;
   clrscr();
   printf(" Enter the no. of elements in an array: \n");
   scanf("%d", &n);
   printf(" Enter the elements of an array: \n");
       for(i=0; i<n; i++)
           scanf("%d", &arr[i]);
```

```
printf("Enter the item to be searched :\n");
scanf("%d", &item);
for(i=0; i<n; i++)
    {
      if(item == arr[i])
          {
            printf("%d found at position %d\n", item, i+1)
            flag = 1;
            break;
          }
    }
if(flag == 0)
    printf("Item %d is not found in an array:\n", item);
getch();
}
```

output : Enter the no. of elements in an array : 7

Enter the elements of an array :

5 7 9 4 3 2 12

Enter the item to be searched : 3

3 found at position 5

```
/* write a program to find the maximum and minimum
   element in an array */
#include<stdio.h>
#include<conio.h>
void main()
{
  int i, arr[10] = {2,5,4,1,8,6,9,3,7,12};
  int min, max;
```

```
min = max = arr[0];
for ( i=0; i<09; i++)
        {  if ( arr[i] < min)
              min = arr[i];
           if ( arr[i] > max)
              max = arr[i];
        }
printf (" Minimum = %d  Maximum = %d \n", min, max);
getch();
}
```

Output : Minimum = 1 Maximum = 12

```
/* write a program to sort the elements in an array */
#include <stdio.h>
#include <conio.h>
void main()
{  int a[10], i, n, temp, j=0;
   clrscr();
   printf (" Enter the no. of elements in an array : \n");
      scanf ("%d", &n);
   printf ( " Enter the elements to an array : \n");
        for(i=0; i<n ; i++)
           scanf ("%d", &a[i]);
   for ( i=0;  i<n-1 ; i++)
        {  for ( j= i+1 ; j<n; j++)
              {  if ( a[i] > a[j] )
                    {
                       temp = a[i];
```

```
            a[i] = a[j];
            a[j] = temp;
          }
       }
    }
printf(" The sorted array elements are :\n");
        for(i=0; i<n; i++)
              printf("%d", a[i]);
   getch();
}
```

output : Enter the no. of elements in an array : 9

Enter the elements to an array :

3 5 7 10 2 11 99 57 69

The sorted array elements are :

2 3 5 7 10 11 57 69 99

/* write a program to merge two arrays of no.s into a new array */

```
#include <stdio.h>
#include <conio.h>
void main()
{
   int a, b, j, k, l;
   int arr1[10], arr2[10], arr3[20];
   clrscr();
  printf(" Enter the no. of elements of array 1 :\n");
  scanf("%d", &a);
  printf(" Enter the no of elements of array 2 :\n");
```

```
    scanf ("%d", &b);
printf ( " Enter %d elements of array1 in sorted order :\n");
    for (j=0; j<a; j++)
        scanf (" %d",&arr1[j]);
printf ( " Enter %d elements of array2 in sorted order :\n");
    for (k=0; k<b; k++)
        scanf (" %d", &arr2[k]);
 j=0;
 k=0;
 l=0;
while ( j<a && k<b)
        {  if ( arr1[j] <= arr2[k])
                arr3[l] = arr1[j];
                 j++;
                 l++;
           else
                arr3[l] = arr2[k];
                 k++;
                 l++;
         }
while ( j<a)
        { arr3[l] = arr1[j];
            l++;
            j++;
         }
 while ( k<b)
        {
          arr3[l] = arr2[k];
            l++;
          }  k++;
```

```
    printf (" New merged array in sorted order is : \n");
    for ( j=0; j<l; j++)
        printf ( " %d \t", arr3[j]);
    getch();
}
```

output : Enter the no. of elements of array 1 : 4
Enter the no. of elements of array 2 : 4
Enter 4 elements of array 1 in sorted order :
3
4
5
7
Enter 4 elements of array 2 in sorted order :
9
10
13
20
New merged array in sorted order is :
3 4 5 7 9 10 13 20

/* write a program to search an item in an array using Binary search */

```
#include <stdio.h>
#include <conio.h>
void main( )
{
    int arr[20], low = 0, high, mid, item, n;
    clrscr();
    printf(" Enter the no. of elements you want to insert in an
                                                array :\n);
    scanf ( " %d", &n);
```

```
printf("Enter the elements in the array in sorted order:\n");
for(i=0; i<n; i++)
    {
    printf("Enter an element:\n");
     scanf("%d", &a[i]);
    }
printf("Enter the item to be searched:\n");
   scanf("%d", &item);
  high = n-1;
  mid = (low+high)/2;
while(low<=high && a[mid]!=item)
    {
      if(a[mid]<item)
          low = mid+1;
       else
          high = mid-1;
        mid = (low+high)/2;
    }
  if(a[mid]==item)
       printf("Item is found at position %d\n", mid+1);
  else
       printf("Item is not found in an array:\n");
  getch();
}
```

output: Enter the no. of elements you want to insert in an array : 5

Enter the elements in the array in sorted order :

Enter an element : 39

Enter an element : 45

Enter an element : 59

Enter an element : 62

Enter an element : 99

Enter the item to be searched : 59

Item is found at position 3

Traversal :

Definition : Traversal means visiting each elements in an array exactly once.

Algorithm : Traverse (a, n)

```
for i ← 1 to n
Apply process to a[i]
End for
```

Here the process may be :

1) Displaying the contents of an array.
2) Finding max^m & min^m elements in an array
3) Searching an element in an array.

Two-Dimensional Array :

→ A two dimensional array is an array of one dimensional arrays.

→ A single dimensional array can store a list of values. Whereas a two-dimensional array can store a table of values.

Declaration of 2-D Array :

The syntax for the declaration of a 2-D array is ;

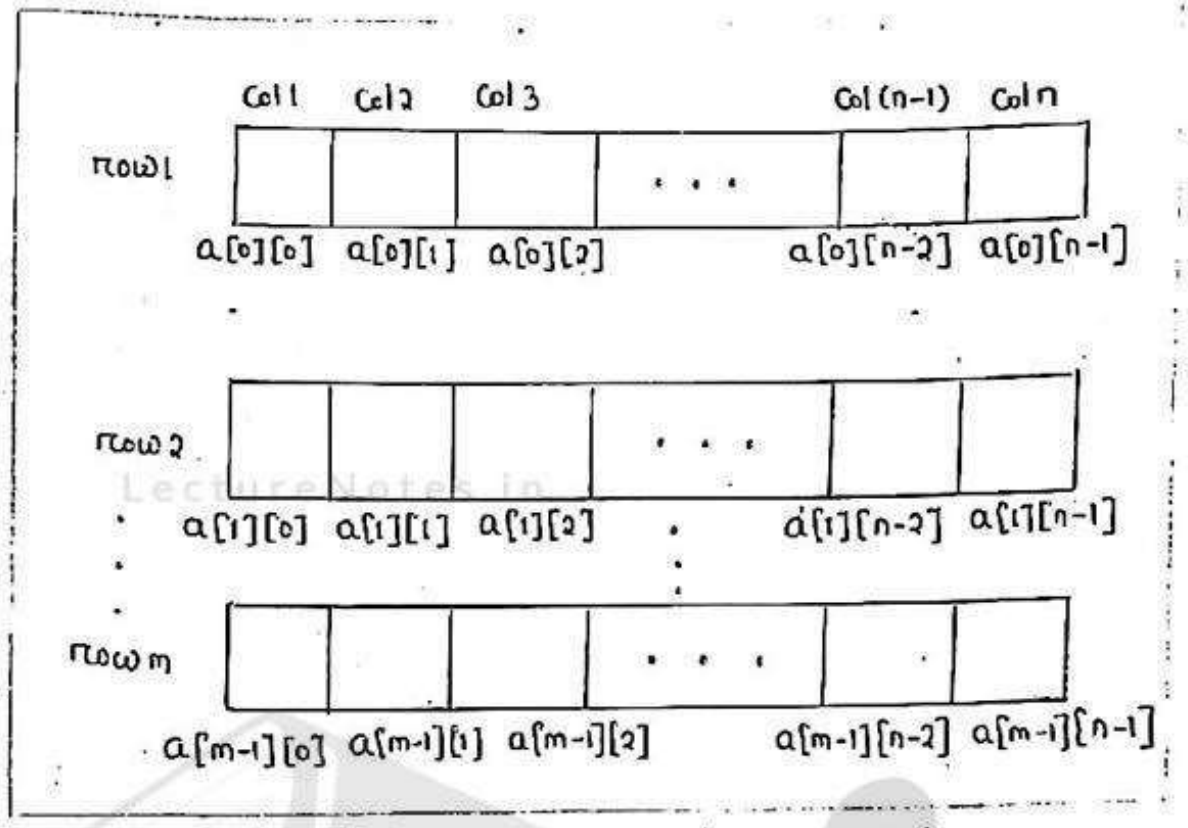
```
data-type array-name [Row-size][Column-size];
```

Here, row-size specifies the number of rows and column-size specifies the number of columns in the array. The total no. of elements in the array are row-size * column-size.

Example : int a[4][5];

Here array is 'a' with 4 rows and 5 columns. The individual elements of this array can be accessed by applying two subscripts, where the first subscript denotes the row number and the second subscript denotes the column number. The starting element of this array is a[0][0] and the last element is a[3][4]. The total no. of elements in this array is 4 * 5 = 20.

→ In two-dimensional array, the element declaration (both row and column) done with zero origin subscripting.



(m x n two-dimensional array a)

→ The array int a[3][3] will have elements as ;

a[0][0] a[0][1] a[0][2]

a[1][0] a[1][1] a[1][2]

a[2][0] a[2][1] a[2][2]

→ A 2-D array can be called as a table, as they stores the table of values.

Initialization of 2-D Array :

→ The general form of initializing a 2-D array is ;

```
data-type array-name [row-size][column-size] =
                        { list of values } ;
```

→ Example : int a[3][3] = { 9,9,9, 10,10,10, 11,11,11 } ;

or int a[3][3] = { {9,9,9}, {10,10,10}, {11,11,11} } ;

a[0][0]	9
[0][1]	9
[0][2]	9
[1][0]	10
[1][1]	10
[1][2]	10
[2][0]	11
[2][1]	11
[2][2]	11

$$a = \begin{bmatrix} 9 & 9 & 9 \\ 10 & 10 & 10 \\ 11 & 11 & 11 \end{bmatrix}_{3\times3}$$

→ int a[4][3] = {11, 10, 12, 9, 8, 7, 6, 13, 15, 19, 21, 62};

a[0][0]	11
[0][1]	10
[0][2]	12
[1][0]	9
[1][1]	8
[1][2]	7
[2][0]	6
[2][1]	13
[2][2]	15
[3][0]	19
[3][1]	21
[3][2]	62

$$a = \begin{bmatrix} 11 & 10 & 12 \\ 9 & 8 & 7 \\ 6 & 13 & 15 \\ 19 & 21 & 62 \end{bmatrix}_{4\times3}$$

→ A 2-D array can also be stored like;

int a[3][3] = {9, 10, 52, 45, 6, 5, 3, 13, 15};

	row0			row1			row2		
	col0	col1	col2	col0	col1	col2	col0	col1	col2
a	9	10	52	45	6	5	3	13	15
	100	102	104	106	108	110	112	114	116

In 2-D array the address of the position of the element can be found out by the formula;

$$a + (i \times C + j)(\text{size of data-type})$$

Example: int i=1, j=1 int a[3][2] = {5,7,9,8,2,4};

Column C=2 C = no. of column

a = 100

address = 100 + (1 × 2 + 1)(2)

= 100 + 6

= 106

	row 0		row 1		row 2	
	col 0	col 1	col 0	col 1	col 0	col 1
a	5	7	9	8	2	4
	100	102	104	106	108	110

Processing 2-D Array:

→ For processing 2-D arrays, we use two nested for loops.

→ The outer for loop corresponds to the row and the inner for loop corresponds to the column.

```
int arr[4][5];
```

(i) Reading values in array arr;

```
for( i=0 ; i<4 ; i++)
    for ( j=0 ; j<5 ; j++)
        scanf ( "%d", &arr[i][j]);
```

(ii) Displaying values of array arr;

```
for ( i=0 ; i<4 ;i++ )
    for( j=0 ; j<5 ; j++)
        printf ("%d\t", arr[i][j]);
```

```
/* write a program to input and display the matrix */
#include <stdio.h>
#include <conio.h>
void main()
{
   int a[10][10], i, j, r, c;
   clrscr();
printf("Enter the order of matrix a :\n");
 scanf("%d%d", &r, &c);
printf("Enter %d elements to a matrix a :\n", r*c);
      for(i=0; i<r; i++)
      for(j=0; j<c; j++)
         scanf("%d", &a[i][j]);
printf("Elements of matrix a are :\n");
        for(i=0; i<r; i++)
          {
            for(j=0; j<c; j++)
              {
                printf("%d\t", a[i][j]);
              }
            printf("\n");
          }
      getch();
}
```

output : Enter the order of matrix a : 2
3
Enter 6 elements to a matrix a :
3
6
7
5
4
2

Elements of matrix a are :

3 6 7
5 4 2

```
/* Addition of two matrices */
#include <stdio.h>
#include <conio.h>
void main()
{
   int a[10][10], b[10][10], d[10][10];
   int i, j, m, c, p, q;
   clrscr();
printf("Enter the order of matrix a :\n");
scanf("%d %d", &m, &c);
printf("Enter %d values to matrix a :\n", m*c);
    for(i=0; i<m; i++)
    for(j=0; j<c; j++)
        scanf("%d", &a[i][j]);
printf("Enter the order of matrix b:\n");
 scanf("%d %d", &p, &q);
printf("Enter %d values to matrix b: \n", p*q);
       for(i=0; i<p; i++)
       for(j=0; j<q; j++)
           scanf("%d", &b[i][j]);

printf("Elements of matrix a :\n");
     for(i=0; i<m; i++)
         {
```

```
    for ( j=0 ; j<c ; j++)
      {
        printf (" %d\t", a[i][j]) ;
      }
      printf ( "\n") ;
  }
printf ( " Elements of matrix b : \n") ;
    for ( i=0 ; i<p ; i++)
        {
          for ( j=0 ; j<q ; j++)
            { printf ( " %d\t" , b[i][j]) ;
            } printf ("\n") ;
        }
if ( n==p && c==q)
    {
      for( i=0;  i<n ; i++)
          {
          for ( j=0 ; j<q ; j++)
            d[i][j] = a[i][j] + b[i][j] ;
          }
printf ( " The resultant matrix is : \n") ;
        for ( i=0 ; i<n ; i++)
          {
            for ( j=0 ; j<q ; j++)
              {
              printf ( " %d\t", d[i][j]) ;
            }
          printf ( "\n") ;
    } }
else
    printf ( " Addition is not possible : \n") ;
  getch() ;
}
```

output:

```
Enter the order of matrix a : 2
2
Enter 4 values to matrix a :
5
6
4
3
Enter the order of matrix b: 2
2
Enter 4 values to matrix b :
9
8
7
5
Elements of matrix a :
5  6
4  3
Elements of matrix b :
9  8
7  5
The resultant matrix is :
14  14
11  8
```

```
/* Matrix Multiplication */
#include <stdio.h>
#include <conio.h>
void main()
{
   int a[10][10], b[10][10], c[10][10];
   int i, j, p, q, m, n, k;
   clrscr();
printf("Enter the order of matrix a: \n");
scanf("%d %d", &m, &n);
printf("Enter the order of matrix b: \n");
scanf("%d %d", &p, &q);
if (n != p)
{
   printf("Matrix multiplication is not possible \n");
}
else {
      printf("Enter the elements of matrix a: \n");
        for(i=0; i<m; i++)
          {
            for(j=0; j<n; j++)
              scanf("%d", &a[i][j]);
          }
      printf("Enter the elements of matrix b: \n");
        for(i=0; i<p; i++)
          {
            for(j=0; j<q; j++)
              scanf("%d", &b[i][j]);
          }
printf("Elements of matrix a: \n");
```

```
for ( i=0; i<m; i++ )
    {
      for ( j=0; j<n; j++)
          {
            printf (" %d\t", a[i][j]);
          }
        printf ("\n");
    }
printf ( " Elements of matrix b: \n");
    for (i=0;  i<p; i++)
        {
          for ( j=0; j<q ; j++)
            {
              printf (" %d\t", b[i][j]);
            }
          printf ("\n");
        }
printf ( " The resultant matrix c is : \n");
    for ( i=0; i<m; i++ )
        { for ( j=0; j<q ; j++)
            {
              c[i][i] = 0;
            for ( k=0; k<p ; k++)
              c[i][i] = (a[i][k] * b[k][j]) + c[i][j];
            }
        }
  for ( i=0; i<m; i++)
        { for (j=0; j<q ; j++)
            {
              printf (" %d", c[i][j]);
            }
            printf ("\n") ;
        }
    getch();
}
```

Output :

```
Enter the order of matrix a : 3
3
Enter the order of matrix b : 3
3
Enter the elements of matrix a :
3
5
7
8
4
2
1
4
6
Enter the elements of matrix b :
2
4
5
3
1
7
9
12
10
Elements of matrix a:
3  5  7
8  4  2
1  4  6
Elements of matrix b:
2  4  5
3  1  7
9  12 10
The resultant matrix c is :
84  101  120
46  60   88
68  80   93
```

```
/* Transposing a matrix */
#include <stdio.h>
#include <conio.h>
void main()
{
   int a[10][10], i, j, r, c, b[10][10];
   clrscr();
 printf("Enter the order of matrix a :\n");
  scanf("%d %d", &r, &c);
 printf("Enter %d values to matrix a :\n", r*c);
     for(i=0; i<r; i++)
     for(j=0; j<c; j++)
          scanf("%d", &a[i][j]);
  printf("Elements of matrix a are :\n");
       for(i=0; i<r; i++)
          {
            for(j=0; j<c; j++)
               {
                 printf("%d\t", a[i][j]);
               }
             printf("\n");
          }
   for(i=0; i<c; i++)
       {
         for(j=0; j<r; j++)
              {
                 b[i][j] = a[j][i];
              }
        }
  printf("After transposing matrix a is :\n");
```

```
    for(i=0; i<c ; i++)
       { for (j=0; j<r; j++)
            { printf ( " %d\t", b[i][j]);
            }
         printf ("\n");
       }
  getch();
}
```


Output :

```
Enter the order of matrix a : 2
3
Enter 6 values to matrix a :
5
4
3
7
8
9
Elements of matrix a are :
    5   4   3
    7   8   9
After transposing matrix a is :
    5   7
    4   8
    3   9
```

```
/* write a program to check whether a matrix is symmetric
   or not */

#include <stdio.h>
#include <conio.h>
void main()
{
   int a[10][10], b[10][10], i, j, m, n
   clrscr();
printf("Enter the order of matrix a: \n");
scanf("%d %d", &m, &n);
printf("Enter %d elements to matrix a: \n", m*n);
      for(i=0; i<m; i++)
      for(j=0; j<n; j++)
            scanf("%d", &a[i][j]);
printf("Elements of matrix a are: \n");
        for(i=0; i<m; i++)
          { for(j=0; j<m; j++)
              {
                printf("%d\t", a[i][j]);
              }
            printf("\n");
          }

printf("After transposing matrix a is: \n");
        for(i=0; i<n; i++)
          {
            for(j=0; j<m; j++)
              {
                b[i][j] = a[j][i];
```

```
for(i=0; i<n; i++)
   { for ( j=0 ; j<m; j++)
        { printf ( " %d\t", b[i][j]) ;
        }
     printf ("\n") ;
}
for ( i=0 ; i<n; i++)
     {
       for ( j=0 ; j<m ; j++)
          {
            if ( a[i][j] = = b[i][j])
              {
                continue ;
              // printf ( " Matrix is symmetric(\n") ;
              }
            else
               printf ( " Not a symmetric matrix") ;
          }
       }
     printf (" Matrix is symmetric(\n");
getch();
}
```

Output :

```
Enter the order of matrix a : 3
3
Enter 9 elements to matrix a :
5
6
7
6
7
5
7
5
3
```

Elements of matrix a are :

```
5  6  7
6  7  5
7  5  3
```

After transposing matrix a is :

```
5  6  7
6  7  5
7  5  3
```

Matrix is symmetric.

/* write a program to find out the sum of all the diagonal elements in a matrix */

```
#include <stdio.h>
# include <conio.h>
void main()
{
   int  a[10][10], i, j, r, c, s;
   clrscr();
   printf ("Enter the order of matrix a : \n");
   scanf ("%d %d", &r, &c);
   printf ("Enter the elements of matrix a : \n");
        for( i=0; i<r ; i++)
             for ( j=0; j<c ; j++)
                  scanf ("%d", &a[i][j]);
   printf ("The elements of matrix a is : \n");
```

```
    for( i=0; i<r ; i++)
      {
        for (j=0 ; j<c ; j++)
          {
            printf ("%d\t", a[i][j]);
          }
        printf ("\n");
      }
  s = 0;
if ( r == c)
   {
     for ( i=0; i<r ; i++ )
          {
            for ( j=0; j< c ; j++ )
            {
          if ( i+j == 0 || i+j == 2 || i+j == 4 )
                                || i+j = 6 || i+j = 8 )
              s = s + a[i][j] ;
          }
      }
  printf (" Sum of diagonal = %d\n", s);
  }
  else
      printf (" Addition of diagonal not possible \n");

  getch();
}
```

Output : Enter the orden of matrix a : 3

3

Enter the elements of matrix a :

3

7

9
2
5
8
10
12
50

The elements of matrix a is :

3	7	9
2	5	8
10	12	50

Sum of diagonal = 77

Problems :

/* write a program to search an element using Binary search */

```
#include <stdio.h>
#include <conio.h>
void main()
{
    int a[90], low=0, high, mid, n, item;
    clrscr();
    printf(" Enter total no. of element you want to insert in an array");
    scanf("%d", &n);
    high= n-1;
    mid = (low + high)/2;
 printf(" Enter the elements in sorted order :\n");
  for( i=0; i<n; i++)
   scanf("%d", &a[i]);
 printf(" Enter the item to be searched :\n");
 Scanf("%d", & item);
while ( low <=high && a[mid] != item)
      {
        if ( a[mid] < item)
            low = mid+1;
       else
            high = mid-1;
        mid = (low + high)/2;
    }
if ( a[mid] == item)
      printf(" Item is found at position %d\n", mid+1);
else
      printf(" Item is not found in an array");
}     getch();
```

O/P : Enter total no. of elements you want to insert in an array
7
Enter the elements in sorted order :
3 9 12 15 29 44 59
Enter the item to be searched : 44
Item is found at position 6.

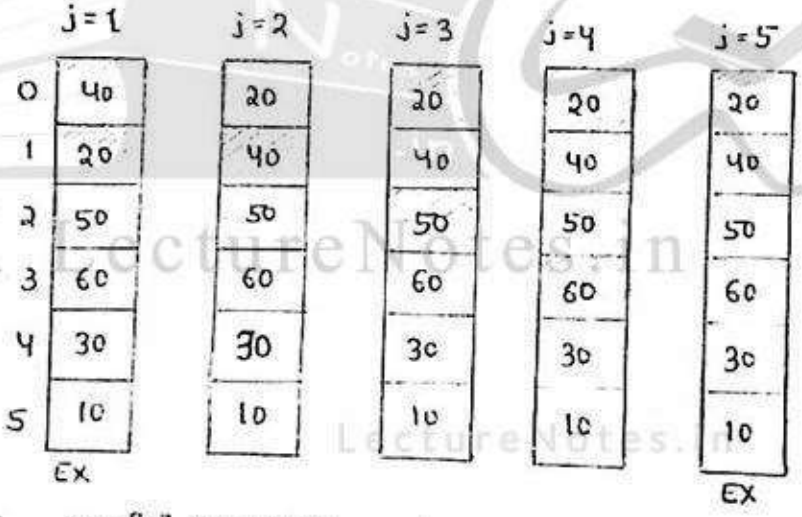
Selection Sort :

Let us take a list of elements in unsorted order and sort them by applying selection sort.

Elements of the array are :-

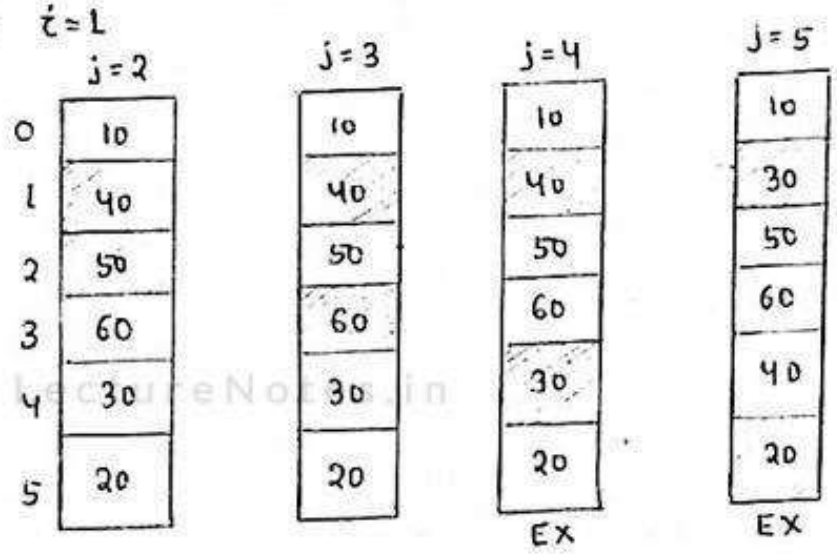
40 20 50 60 30 10

Pass 1 : $i = 0$



In pass 1 arr[0] > arr[1], Exchange
arr[0] > arr[2] No
arr[0] > arr[3] No
arr[0] > arr[4] No
arr[0] > arr[5], Exchange

pass 2 : i = 1



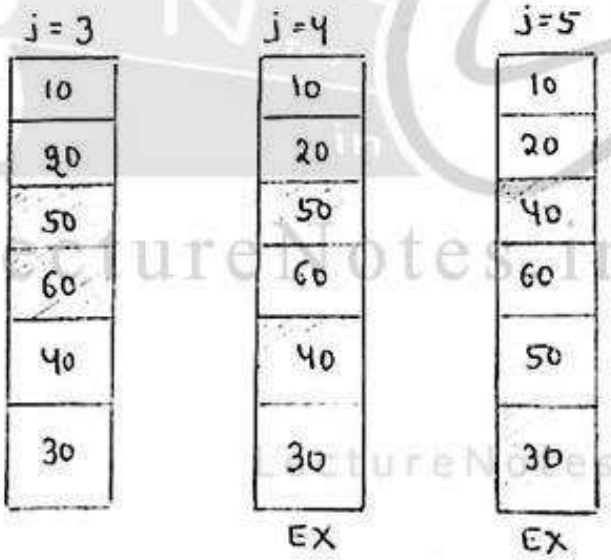
on pass 2 arr[1] > arr[2] No

arr[1] > arr[3] No

arr[1] > arr[4], Exchange

arr[1] > arr[5], Exchange

pass 3 : i = 2

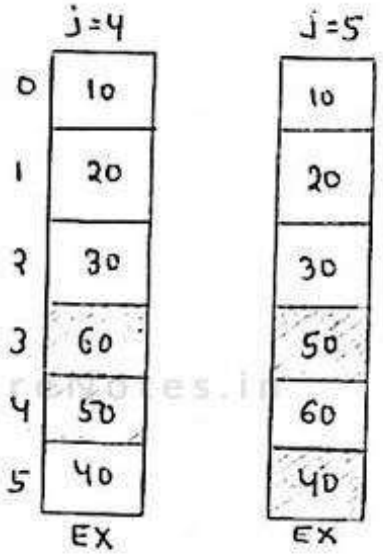


on pass 3 arr[2] > arr[3] No

arr[2] > arr[4], Exchange

arr[2] > arr[5], Exchange

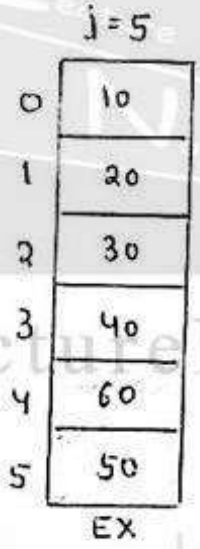
pass 4 : i=3



In pass 4 arr[3] > arr[4], Exchange

arr[3] > arr[5], Exchange

pass 5 : i=4



In pass 5 arr[4] > arr[5], Exchange

Finally after pass 5 we get the sorted array.

10 20 30 40 50 60

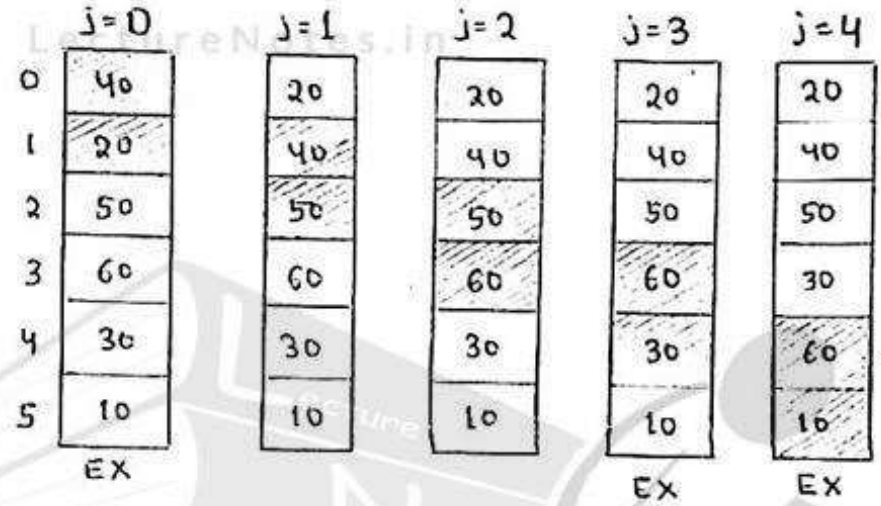
```
/* program of sorting using selection sort */
#include <stdio.h>
#include <conio.h>
#define SIZE 10
void main()
{
    int arr[SIZE];
    int i, j, temp;
    clrscr();
printf("Enter elements of the array :\n");
    for(i=0; i<SIZE; i++)
        scanf("%d", &arr[i]);
 for(i=0; i<SIZE-1; i++)
      for(j=i+1; j<SIZE; j++)
        {  if(arr[i] > arr[j])
             {
                temp = arr[i];
                arr[i] = arr[j];
                arr[j] = temp;
             }
        }
   printf("The sorted array is :\n");
      for(i=0; i<SIZE; i++)
          printf("%d", arr[i]);
       printf("\n");
  getch();
}
```

Bubble Sort: let us take a list of element in unsorted order and sort them by using bubble sort.

Elements of the array are -

40 20 50 60 30 10

pass 1: $i = 0$



In pass 1 arr[0] > arr[1], exchange

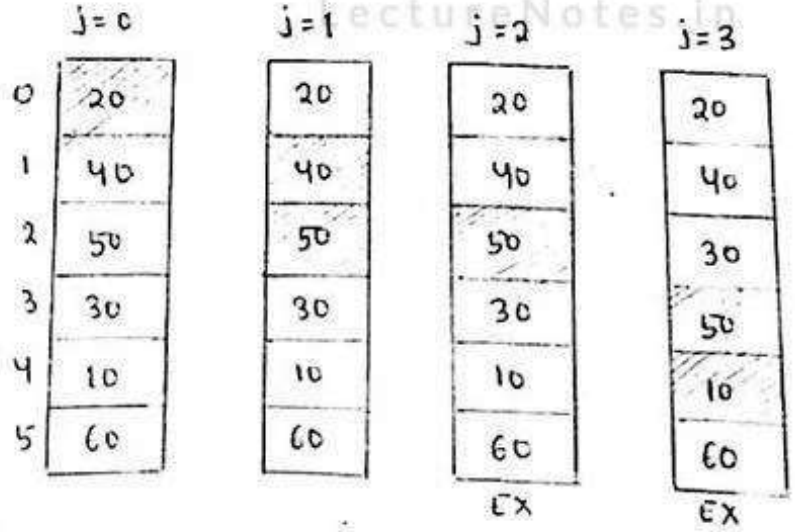
arr[1] > arr[2] No

arr[2] > arr[3] No

arr[3] > arr[4], Exchange

arr[4] > arr[5], Exchange

pass 2: $i = 1$



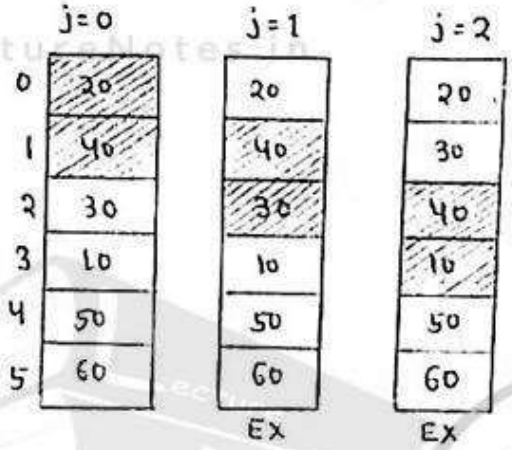
In pass 2. arr[0] > arr[1] No

arr[1] > arr[2] No

arr[2] > arr[3], Exchange

arr[3] > arr[4], Exchange

pass 3: i=2

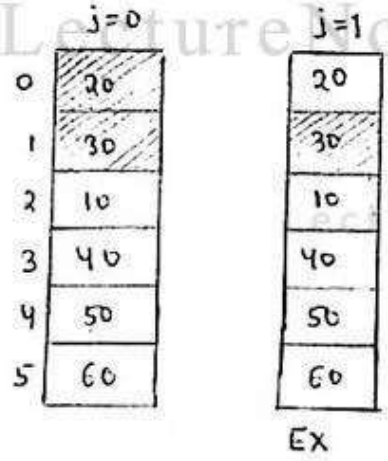


In pass 3 arr[0] > arr[1] No

arr[1] > arr[2], Exchange

arr[2] > arr[3], Exchange

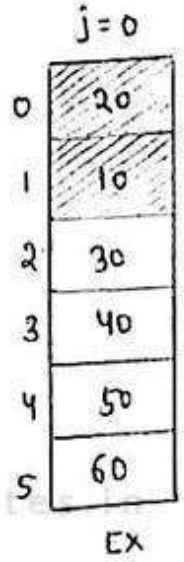
pass 4: i=3



In pass 4. arr[0] > arr[1] No

arr[1] > arr[2], Exchange

pass 5 : i=4



In pass 5 arr[0] > arr[1], Exchange

Finally after pass 5 we got the sorted array.

10 20 30 40 50 60

/* program of sorting using bubble sort */

```
#include <stdio.h>
# include <conio.h>
# define SIZE 20
void main()
{
   int arr[SIZE];
   int i, j, temp;
   clrscr();
  printf (" Enter elements of the array :\n");
    for ( i=0 ; i<SIZE ; i++ )
       scanf ("%d", &arr[i]) ;

    for ( i=0 ; i < SIZE-1 ; i++ )
       {
          for ( j=0 ;  j< SIZE -1-i  ; j++ )
                {
```

```
        if ( arr[j] > arr[j+1])
            {
              temp = arr[j];
              arr[j] = arr[j+1];
              arr[j+1] = temp;
            }
        }
    }
  printf ( " The sorted array is : \n" ) ;
     for (i=0 ; i<SIZE ; i++)
        printf ( "%d" , arr[i]) ;
     printf ( "\n") ;
  getch() ;
}
```

/* write a program to print the pascal triangle */

```
#include <stdio.h>
#include <conio.h>
#define SIZE 15
void main()
{   int a[SIZE][SIZE];
    int i, j, n;
    clrscr();
   printf ( " Enter n :") ;
   scanf ("%d", &n);
   for( i=0 ; i<=n ; i++ )
    {
        for( j=0 ; j<=i ; j++ )
            if ( j==0 || i==j )
                a[i][j] = 1;
```

```
else
    a[i][j] = a[i-1][j-1] + a[i-1][j];
}
 for(i=0; i<n; i++)
    {
      for (j=0; j<=i; j++)
            {
            printf ("%d", a[i][j]);
            }
            printf ("\n");
    }
 getch();
}
```

output :

```
Enter n : 7
1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
1 5 10 10 5 1
1 6 15 20 15 6 1
```