7. UNDERSTAND ARRAYS

7.1: DEFINE ARRAY:

An array is **a fixed-size sequenced collection of elements of the same data type**. It is simply **a grouping of like-type data**.

An array is a sequenced **collection of related data items** that **shares a common name**. An array can be used to represent a list of numbers or a list of names.

For e.g.: int salary[50];

is an array called salary to store salaries of 50 employees with single name.

Arrays can be used to:

1. To store list of employees in an organization.
2. Store Table of daily rainfall data.
3. To store list of products and their cost.
4. To store list of customers and their telephone numbers.

Types of arrays:

- ✓ One-dimensional arrays
- ✓ Two-dimensional arrays
- ✓ Multi-dimensional arrays

7.2: Describe declaration and initialization of One –dimensional Array with syntax and sample program

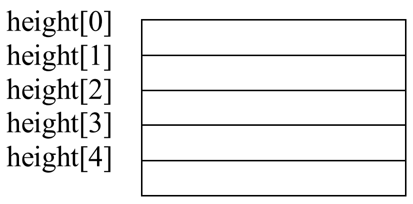
One-dimensional arrays: A list of items can be given **one variable name using only one subscript** and such a variable is called a **single-subscripted** variable or a **one-dimensional array**.

Declaration of one-dimensional arrays:

Data-type array-name[size];

For e.g.: float height[5];

Declares the height to be an array containing 5 real elements and the computer reserves five storage locations as shown below:



Initialization of one-dimensional arrays: An array can be initialized at either of the following stages:

1. **At compile-time**
2. **At run-time**

1. **Compile-time initialization:**

We can initialize the elements of arrays in the same way as the ordinary variables when they are declared. The general form of initialization of array is:

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

The values in the list are separated by commas.

For e.g.:int number[3]={10,20,30};

Note:

- ✓ **If** the number of **values in the list is less than the number of elements** , then only those elements will be initialized and the **remaining elements will be set to zero automatically**.

  e.g.: int number[5]={10,20};

  will initialize first 2 elements to 10 and 20 respectively, and the remaining elements to 0.

  e.g.: cha name[5]={'V'};

  will initialize first element to V and remaining elements to NULL character('\0')
- ✓ The size may be omitted. In such cases, the compiler allocates enough space for all initialized elements

  e.g.: int counter[]={1,2,3};
- ✓ Character arrays may be initialized in a similar manner.

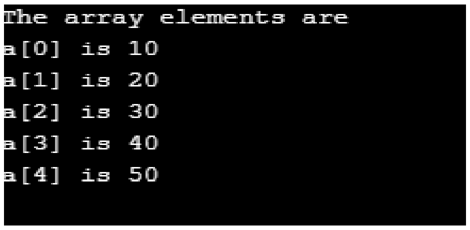
  e.g.: char name[]={'j','o','h','n','\0'};

  or

  char name[]="john";

Example program

```
#include<stdio.h>
main()
{
     int a[5]={10,20,30,40,50};
    printf("The array elements are");
    for(int i=0;i<5;i++)
     printf("a[%d] is %d\n",i,a[i]);
}
```



```
The array elements are
a[0] is 10
a[1] is 20
a[2] is 30
a[3] is 40
a[4] is 50
```

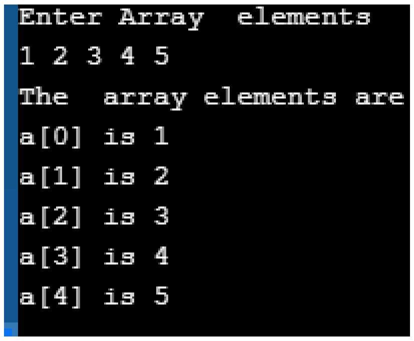
2. **Run –time initialization:** We can also **use scanf** to initialize an array:

int x[3];

scanf ("%d%d%d", &x[o],&x[1],&x[2]);

Example program:

```
#include<stdio.h>
  main()
{
    int a[10],i,n;
   printf("Enter n value\n");
    scanf("%d",&n);
   printf("Enter Array  elements");
   for(i=0;i<n;i++)
    {
    scanf("%d",&a[i]);
    }
  printf("The  array elements are\n");
   for(i=0;i<n;i++)
    printf("a[%d] is %d\n",i,a[i]);
}
```



```
Enter Array  elements
1 2 3 4 5
The  array elements are
a[0] is 1
a[1] is 2
a[2] is 3
a[3] is 4
a[4] is 5
```

7.3 Explain accessing the elements in the ARRAY with sample program:

Accessing 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:

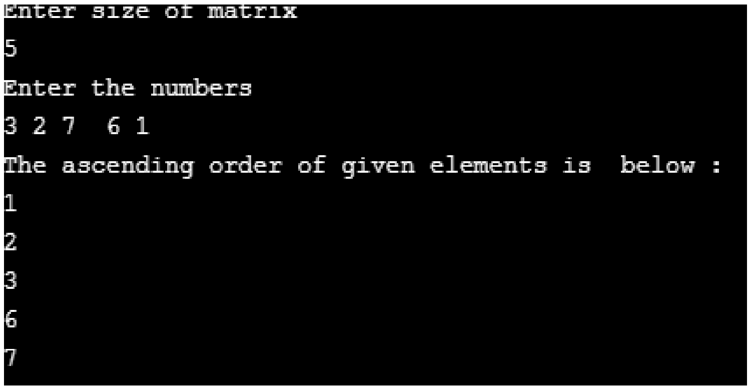
double salary = balance[9];

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

Following is an example which will use all the above mentioned three concepts viz. declaration, assignment and accessing arrays:

7.4: Explain reordering an array in ascending order.

```
#include<stdio.h>
main()
{
   int i, j, temp, n,a[30];
   printf("Enter size of matrix\n");
   scanf("%d", &n);
   printf("Enter the numbers \n");
   for (i = 0; i <n;i++)
     scanf("%d", &a[i]);
   for (i = 0; i <n;i++)
   {
     for (j = 0; j < n-1;j++)
     {
        if (a[j] >a[j+1])
        {
          temp=a[j];
          a[j] = a[j+1];
          a[j+1] =temp;
        }
     }
```



```
Enter size of matrix
5
Enter the numbers
3 2 7  6 1
The ascending order of given elements is  below :
1
2
3
6
7
```

```
 }
 printf("The ascending order of given elements is  below :\n");
    for (i = 0; i <n;i++)
       printf("%d\n",a[i]);
}
```
Explanation: Here we will perform the n-iterations, where the n depends on the size of the array.
In first iteration the first element will compared with remaining elements in the array, if any element found to be small than the first element then they are swapped. Finally the minimum element in the array will be placed at 0th position.

Now in second iteration the second element will compared with remaining elements in the array, if any element found to be small than the second element then they are swapped. Finally the second minimum element in the array will be placed at 1th position.
This process continues for n-elements in the array. Finally we get sorted arrayofelements.

7.5: Explain declaration and initialization of two-dimensional Arrays.

C allows to **store a table of values,** using two-dimensional arrays. The 2-dimensional arrays are declared as follows:

Data-type array_name[row-size][column-size];

The two-dimensional arrays are stored in the memory as shown in below:

	Column 0	column 1	column 2
Row0→	[0][0]	[0][1]	[0][2]
Row 1→	[1][0]	[1][1]	[1][2]
Row 2→	[2][0]	[2][1]	[2][2]

Initializing two-dimensional arrays at compile-time:

Like one-dimensional arrays, two-dimensional array may be initialized by following their declaration with a list of initial values enclosed in braces. The general form of initialization of 2D array is:

Data-type array-name[rowsize][columnsize]={list of values};

For e.g

int table[2][3]={0,0,0,1,1,1};

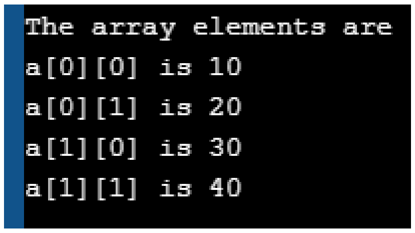
the initialization is done row by row. The equivalent statement is:

int table[2][3]={{0,0,0},{1,1,1}};

we can also initialize a two-dimensional array as **int table[2][3]={**
```
                        {0,0,0},
                        {1,1,1}
                    };
```

Note: we need not specify the size of the first dimension, for e.g :

int table[][3]={0,0,0,1,1,1};

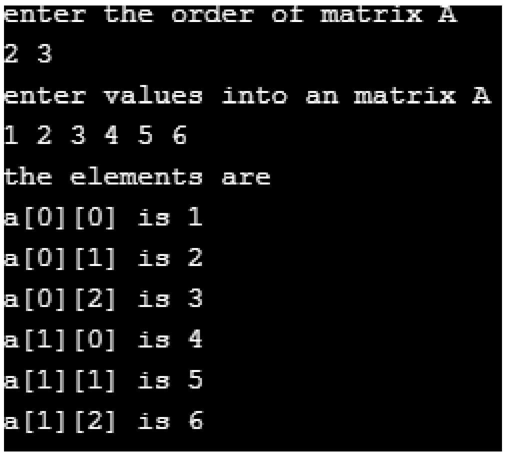
Example program :
```
#include<stdio.h>
main()
{
int a[2][2]={10,20,30,40};
printf("The array elements are\n");
for(int i=0;i<2;i++)
for(int j=0;j<2;j++)
printf("a[%d][%d] is %d\n",i,j,a[i][j]);
}
```

```
The array elements are
a[0][0] is 10
a[0][1] is 20
a[1][0] is 30
a[1][1] is 40
```

Initializing two-dimensional arrays at run-time:

The 2-dimensional arrays can initialized using two looping statements called
Outer loop-for indicating rows
Inner loop-for indicating columns.
```
e.g.: for(i=0;i<2;i++)
 {
for(j=0;j<2;j++)
{
scanf("%d",&x[i][j]);        /*reading values into 2x2 marix*/
}
 }
```
Will initialize the elements of 2x2 matrix at run-time.

Example program :

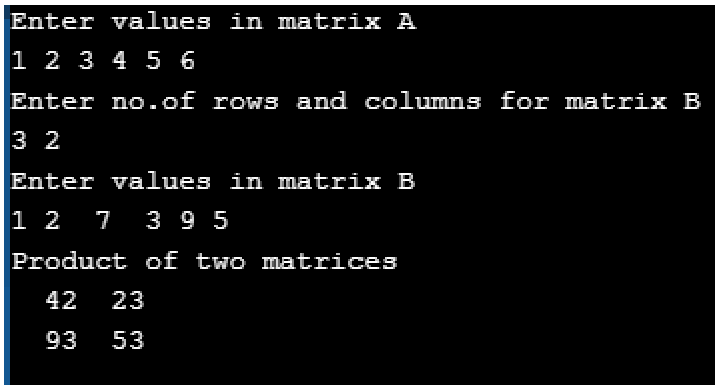
```
#include<stdio.h>
main()
{
int a[10][10],m,n,i,j;
printf("enter the order of matrix A \n");
scanf("%d%d",&m,&n);
printf("enter values into an matrix A \n");
for(i=0;i<m;i++)
  {
for(j=0;j<n;j++)
 {
scanf("%d",&a[i][j]);
}
  }
printf ("the elements are\n");
for(i=0;i<m;i++)
  {
for(j=0;j<n;j++)
 {
printf("a[%d][%d] is %d\n",i,j,a[i][j]);
}
}
}
```



```
enter the order of matrix A
2 3
enter values into an matrix A
1 2 3 4 5 6
the elements are
a[0][0] is 1
a[0][1] is 2
a[0][2] is 3
a[1][0] is 4
a[1][1] is 5
a[1][2] is 6
```

7.6 <u>Multiplication of Matrix:</u>

```
#include<stdio.h>
main()
{
int a[10][10],b[10][10],c[10][10],i,j,k,m,n,p,q;
printf("Enter no.of rows and columns for matrix A\n");
scanf("%d%d",&n,&m);
printf("Enter values in matrix A\n");
for(i=0;i<n;i++)
{
for(j=0;j<m;j++)
scanf("%d",&a[i][j]);
}
printf("Enter no.of rows and columns for matrix B\n");
scanf("%d%d",&p,&q);
printf("Enter values in matrix B\n");
for(i=0;i<p;i++)
{
for(j=0;j<q;j++)
scanf("%d",&b[i][j]);
}
if(m==p)
{
printf("Product of two matrices\n");
for(i=0;i<n;i++)
{
for(j=0;j<q;j++)
{
c[i][j]=0;
```

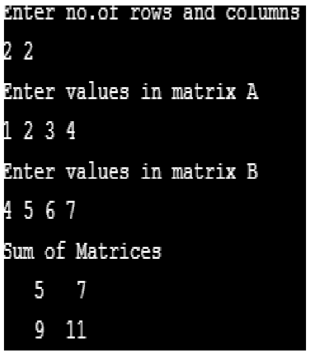


```
Enter values in matrix A
1 2 3 4 5 6
Enter no.of rows and columns for matrix B
3 2
Enter values in matrix B
1 2  7  3 9 5
Product of two matrices
  42  23
  93  53
```

```
for(k=0;k<p;k++)
{
c[i][j]=c[i][j]+a[i][k]*b[k][j];
}
printf("%4d",c[i][j]);
}
printf("\n");
}
}
else
printf("Multiplication not possible\n");
}
```

Addition of two matrices

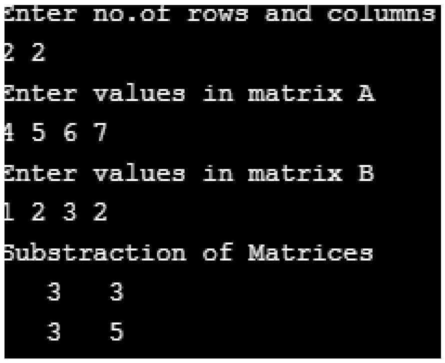
```
#include<stdio.h>
main()
{
int a[10][10],b[10][10],c[10][10],i,j,n,m;
printf("Enter no.of rows and columns\n");
scanf("%d %d",&n,&m);
printf("Enter values in matrix A\n");
for(i=0;i<n;i++)
{
for(j=0;j<m;j++)
{
scanf("%d",&a[i][j]);
}
}
printf("Enter values in matrix B\n");
for(i=0;i<n;i++)
{
for(j=0;j<m;j++)
{
scanf("%d",&b[i][j]);
}
}

for(i=0;i<n;i++)
{
for(j=0;j<m;j++)
{
c[i][j]=a[i][j]+b[i][j];
}
}
printf("Sum of Matrices\n");
for(i=0;i<n;i++)
{
for(j=0;j<m;j++)
{
printf("%4d",c[i][j]);
}
printf("\n");
}
}
```



```
Enter no.of rows and columns
2 2
Enter values in matrix A
1 2 3 4
Enter values in matrix B
4 5 6 7
Sum of Matrices
   5   7
   9  11
```

Substraction of two matrices

```
#include<stdio.h>
main()
{
int a[10][10],b[10][10],c[10][10],i,j,n,m;
printf("Enter no.of rows and columns\n");
scanf("%d %d",&n,&m);
printf("Enter values in matrix A\n");
for(i=0;i<n;i++)
{
for(j=0;j<m;j++)
{
scanf("%d",&a[i][j]);
}
}
printf("Enter values in matrix B\n");
for(i=0;i<n;i++)
{
for(j=0;j<m;j++)
{
scanf("%d",&b[i][j]);
}
}
for(i=0;i<n;i++)
{
for(j=0;j<m;j++)
{
c[i][j]=a[i][j]-b[i][j];
}
}
printf("Subtraction of Matrices\n");
for(i=0;i<n;i++)
{
for(j=0;j<m;j++)
{
printf("%4d",c[i][j]);
}
printf("\n");
}
}
```

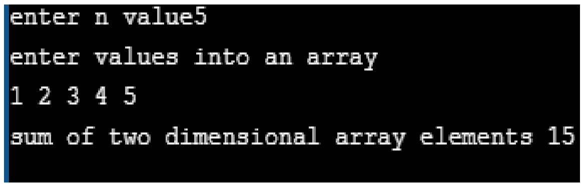


```
Enter no.of rows and columns
2 2
Enter values in matrix A
4 5 6 7
Enter values in matrix B
1 2 3 2
Substraction of Matrices
   3   3
   3   5
```

Extra programs:

/*reading values into 1-dimensional array and print the sum of the values in the array*/

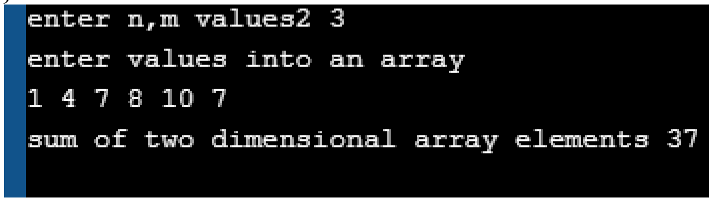
```
#include<stdio.h>
main()
{
int x[10],n,i,sum=0;
printf("enter n value");
            scanf("%d",&n);
printf("enter values into an array \n");
for(i=0;i<n;i++)
 {
scanf("%d",&x[i]);
 }
for(i=0;i<n;i++)
{
   sum=sum+x[i];
}
printf("sum of two dimensional array elements
%d",sum);
}
```



```
enter n value5
enter values into an array
1 2 3 4 5
sum of two dimensional array elements 15
```

/*reading values into 2-dimensional array and print the sum of the values in the array*/

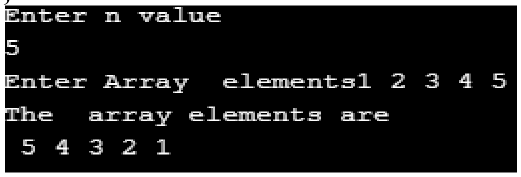
```
#include<stdio.h>
main()
{
int x[10][10],n,m,i,j,sum=0;
printf("enter n,m values");
scanf("%d%d",&n,&m);
printf("enter values into an array \n");
for(i=0;i<n;i++)
 {
for(j=0;j<m;j++)
{
scanf("%d",&x[i][j]);
 }
 }

for(i=0;i<n;i++)
{
for(j=0;j<m;j++)
{
sum=sum+x[i][j];
}
}
printf("sum of two dimensional array elements %d",sum);
}
```



```
enter n,m values2 3
enter values into an array
1 4 7 8 10 7
sum of two dimensional array elements 37
```

/*Write a c program to reverse given array elements*/

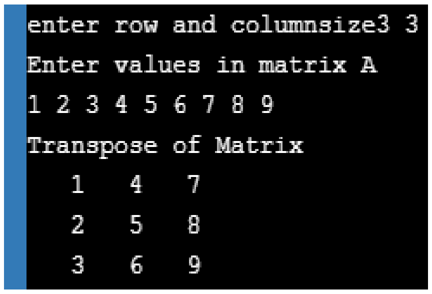
```
#include<stdio.h>
main()
{
int a[10],i,n;
printf("Enter n value\n");
scanf("%d",&n);
printf("Enter Array  elements");
for(i=0;i<n;i++)
{
scanf("%d",&a[i]);
}
printf("The  array elements are\n");
for(i=n-1;i>-1;i--)
printf("%2d",a[i]);
}
```



```
Enter n value
5
Enter Array  elements1 2 3 4 5
The  array elements are
 5 4 3 2 1
```

Transpose of a given matrix

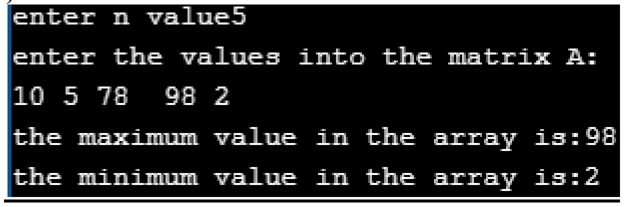
```
main(){
int a[10][10],i,j,n,m;
printf("enter row and columnsize" );
scanf("%d %d",&n,&m);
printf("Enter values in matrix A\n");
for(i=0;i<n;i++){
for(j=0;j<m;j++){
scanf("%d",&a[i][j]);
}
}
printf("Transpose of Matrix\n");
for(i=0;i<m;i++)
{
for(j=0;j<n;j++)
{
printf("%4d",a[j][i]);
}
printf("\n");
}
}
```



```
enter row and columnsize3 3
Enter values in matrix A
1 2 3 4 5 6 7 8 9
Transpose of Matrix
   1   4   7
   2   5   8
   3   6   9
```

/* to find smallest(minimum) and largest(maximum) values in an one-dimensional array*/

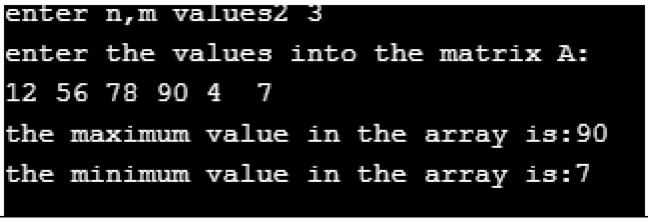
```
 #include<stdio.h>
main()
{
          int a[10],n,i,min,max;
          printf("enter n value");
          scanf("%d",&n);
          printf("enter the values into the matrix A:\n");
          for(i=0;i<n;i++)
          {
                    scanf("%d",&a[i]);
          }
          min=a[0];
          max=a[0];
          for(i=1;i<n;i++)
          {
                    if(a[i]>max)
                    max=a[i];
             else
                    min=a[i];
}
printf("the maximum value in the array is:%d\n",max);
printf("the minimum value in the array is:%d",min);
}
```



```
enter n value5
enter the values into the matrix A:
10 5 78  98 2
the maximum value in the array is:98
the minimum value in the array is:2
```

/* to find smallest(minimum) and largest(maximum) values in an two-dimensional array*/

```
#include<stdio.h>
main()
{
          int a[10][10],n,m,i,j,min,max;
          printf("enter n,m values");
          scanf("%d%d",&n,&m);
          printf("enter the values into the matrix A:\n");
          for(i=0;i<n;i++)
          {
             for(j=0;j<m;j++)
          {
                    scanf("%d",&a[i][j]);
          }
          }
          min=a[0][0];
          max=a[0][0];
          for(i=0;i<n;i++)
          {
             for(j=0;j<m;j++)
          {
                    if(a[i][j]>max)
                    max=a[i][j];
             else
                    min=a[i][j];
          }
          }
printf("the maximum value in the array is:%d\n",max);
printf("the minimum value in the array is:%d",min);
}
```

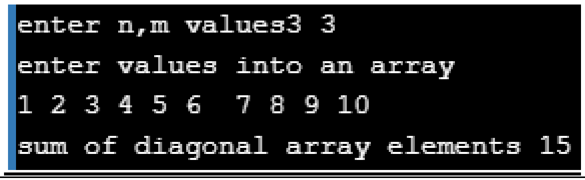


```
enter n,m values2 3
enter the values into the matrix A:
12 56 78 90 4  7
the maximum value in the array is:90
the minimum value in the array is:7
```

```
/* Write a c program to print of sum of diagonal elements in 2D array */
#include<stdio.h>
main()
{
int x[10][10],n,m,i,j,sum=0;
printf("enter n,m values");
scanf("%d%d",&n,&m);
printf("enter values into an array \n");
for(i=0;i<n;i++)
 {
for(j=0;j<m;j++)
{
scanf("%d",&x[i][j]);
 }
 }
for(i=0;i<n;i++)
{
for(j=0;j<m;j++)
```

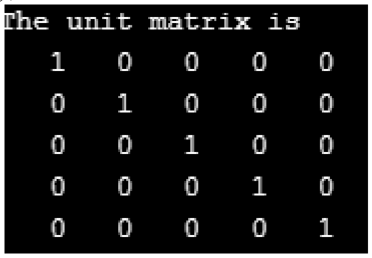
Write a c program to generate a unit matrix with 5 rows and 5 columns

```
#include<stdio.h>
main()
{
int x[10][10],n=5,m=5,i,j;
for(i=0;i<n;i++)
 {
for(j=0;j<m;j++)
{
   if(i==j)
    {
   x[i][j]=1;
    }
   else
    {
   x[i][j]=0;
    }
{
   if(i==j)
sum=sum+x[i][j];
}
}
printf("sum of diagonal array elements %d",sum);
}
```



```
enter n,m values3 3
enter values into an array
1 2 3 4 5 6  7 8 9 10
sum of diagonal array elements 15
```

```
 }
 }
printf("The unit matrix is\n");
for(i=0;i<n;i++)
{
for(j=0;j<m;j++)
{
 printf("%4d",x[i][j]);
}
printf("\n");
}
}
```



```
The unit matrix is
   1   0   0   0   0
   0   1   0   0   0
   0   0   1   0   0
   0   0   0   1   0
   0   0   0   0   1
```