UNIT:1 ARRAYS

Topics To be Covered…

1.1 Declaring and initializing One-Dimensional Array and array Operations
- i. Insertion
- ii. Searching
- iii. Merging
- iv. Sorting
- v. Deletion

1.2 Introduction of String as array of characters Declaration and Initialization of String

1.3 Two-Dimensional Array and its Operations
- i. Insertion, Deletion
- ii. Matrix addition operation

1.4 Multi-Dimensional Arrays

1.5 sscanf() and sprintf() Functions

1.6 Drawbacks of Linear Arrays

1.1 Declaring and initializing One-Dimensional Array and array Operations

- An array is a fixed-size sequenced collection of elements of the same data type that shares a common name.
- An array is a collection of variables of same data type known by a same name.
- **Example:**

  Test score of a class of students.

  List of employees in an organization.
- **Types of Arrays**
  - One dimensional array
  - Two dimensional array
  - Multi dimensional array

Declaring Arrays

Syntax:

Data type arrayname [size] ;

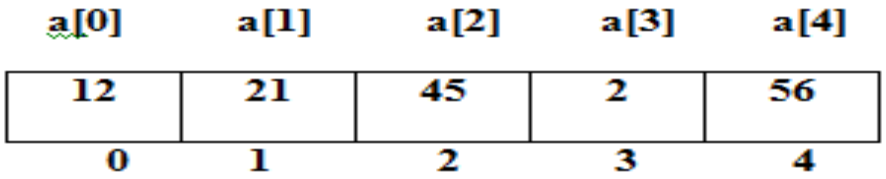
Data type: it defines the type of the array element, like int,float,char,double etc.

Array name: it is the name of variable which represent array.
Size: which is represents within[ ] symbol represent the size of the array.

For example: int a[5];

Here int is a data type.
a is name of variable or array.
5 is the size of the array.

Where a[0] is the first element of the array, while a[4] is the last element of the array.



INITIALIZING ONE-DIMENSIONAL ARRAY

- **Compile Time Initialization**

Syntax:

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

The values in the list are separated by commas.

Example 1:

int num[3]={23,12,32};

Example 2:

int num[5]={54,23,3,28};

INITIALIZING ONE-DIMENSIONAL ARRAY

- **Run Time Initialization**

Example:

```
int num[3];
scanf("%d  %d %d", &num[0], &num[1],
&num[2]);
```

Example:

```
int num[3];
printf("enter three array elements :\n");
for (i=0;i<3;i++)
{
        scanf("%d",&num[i]);
}
```

PROGRAM

Write a program to insert 5 array elements and display it.

```
#include<stdio.h>
#include<conio.h>
void main( )
{
        int a[5],i;
        clrscr( );
        printf("Enter the elements :");
        for(i=0;i<5;i++)
        {
                scanf("%d",&a[i]);
        }
        printf("Elements of array are given below:\n");
        for(i=0;i<5;i++)
        {
                printf("Element no %d = %d \n",i+1,a[i]);
        }
        getch();
}
```

Advantage of an array

- An array is a fixed-size sequenced collection of elements of the same data type that shares a common name.
- We can use one name for similar elements.
- Two-Dimensional array are used to represents matrices.
- Array is used to implement other data structure like link list, stack tree etc.

Characteristics of Array

- Array store elements that have same data type.
- Array store elements in subsequent memory location.
- Array size should be mention in the declaration.
- Array name represent the address of starting elements.

LIST OF OPERATIONS ON ONE DIMENSIONAL ARRAY

- Sorting: We can sort the elements of array in ascending and descending order.
- Merging: We can merge or joint elements of Two One Dimensional array into third one Dimensional array.
- Searching :We can search the any elements from the given array.
- Insertion :We can insert the elements into array at beginning or ending or at specific position.
- Deletion :We can insert the elements into array at beginning or ending or at specific position.

INSERT OPERATION

- We can insert the elements from the array at beginning of array or at ending of array or at specific position of array.

Example:

```
#include<stdio.h>
#include<conio.h>
void main()
{
        int a[6]={1,2,3,4,5};
        int no,i,loc;
        clrscr();
        printf("Elements of array A before insertion : \n");
        for(i=0;i<5;i++)
        {
                printf("%d\n",a[i]);
        }
```

CONT…

```
printf("Enter the element to be inserted into array A : ");
scanf("%d",&no);
printf("Enter the position of the element :");
scanf("%d",&loc);
for(i=4;i>=loc-1;i--)
{
        a[i+1]=a[i];
}
a[loc-1]=no;
printf("Elements of array A after insertion : \n");
for(i=0;i<=5;i++)
{
        printf("%d\n",a[i]);
}
getch();
}
```

DELETE OPERATION

We can delete the elements from the array at beginning of array or at ending of array or at specific position of array.

Example:

```
#include<stdio.h>
#include<conio.h>
void main()
{
        int a[5]={10,20,30,40,50};
        int i,position;
        clrscr();
        printf("Before Deletion elements of Arrays are as below:\n");
        for(i=0;i<5;i++)
        {
                printf("%d\n",a[i]);
        }
```

CONT…

```
        printf("Enter the position of element for deletion operation: ");
        scanf("%d",&position);

        for(i=position-1;i<5;i++)
        {
                a[i]=a[i+1];
        }
        a[5]=0;
        printf("After Deletion from Ending Array Elements are as below:\n");
        for(i=0;i<5;i++)
        {
                printf("%d\n",a[i]);
        }
        getch();
}
```

SEARCH OPERATION

```
#include<stdio.h>
#include<conio.h>
void main()
{
        int a[5]={10,20,30,40,50};
        int no,i,t =0;
        clrscr();
        printf("enter no");
        scanf("%d",&no);
        for(i=0;i<5;i++)
        {
                if(a[i]==no)
                {
                printf(Searching element is at location:%d”,i+1);
                        t=1;
                }
  }
```

CONT…

```
        If(t==0)
        {
                printf("Searching element is not found");
        }
        getch();
}
```

MERGE OPERATION

```
#include<stdio.h>
#include<conio.h>
void main()
{
        int a[5],b[5],c[10],i,j,k=0;
        clrscr();
        printf("Enter elements of array A :\n");
        for(i=0;i<5;i++)
        {
                scanf("%d",&a[i]);
        }
        printf("Enter elements of array B :\n");
        for(j=0;j<5;j++)
        {
                scanf("%d",&b[j]);
        }
```

CONT…

```
for(i=0;i<5;i++)
        {
                c[k]=a[i];
                k++;
        }
        for(j=0;j<5;j++)
        {
                c[k]=b[j];
                k++;
        }
        printf("Elements of array C after Merge Operation :\n");
        for(k=0;k<10;k++)
        {
                printf("%d\n",c[k]);
        }
        getch();
}
```

SORT OPERATION

```
#include<stdio.h>
#include<conio.h>
void main( )
{
        int a[5],i,j,temp;
        clrscr( );
        printf("Enter the 5 array elements :\n");
        for(i=0;i<5;i++)
        {
        scanf("%d",&a[i]);
        }
        for(i=0;i<5;i++)
        {
        for(j=i+1;j<5;j++)
        {
```

CONT…

```
                    if(a[i]>a[j])
                    {
                            temp=a[i];
                            a[i]=a[j];
                            a[j]=temp;
                    }
            }
            }
printf("Array Elements in Ascending Order is given below:
\n");
            for(i=0;i<5;i++)
            {
            printf("%d",a[i]);
            printf("\n");
            }
            getch();
}
```

1.2 Introduction of String as array of characters Declaration and Initialization of String

- A string is a sequence of characters that is treated as single data item.
- It is written between double quotations.

Declaring of String Variable

A string variable is always declared as an array of characters.

Syntax:

char string_name[size];

where, size is the number of characters.

Example:

char city[15];

CONT…

Initialization of string variable

Example 1:

char city[9]="NEW YORK";

char city[9]={'N','E','W',' ','Y','O','R','K','\0'};

Example 2:

char name[10]= "**WELL DONE**";

which declares the name as a character array variable that can hold a maximum of 10 characters.

'W'	'E'	'L'	'L'	' '	'D'	'O'	'N'	'E'	'\0'

- When the compiler sees a character string, then it terminate it with an additional null character. So, the element name[9] holds the null character '\0'.
- When declaring character arrays, we must allow one extra element space for the null character('\0').

1.3 Two-dimensional array

- Two dimensional arrays is used to create a matrix.
- In two dimensional arrays two dimensions are available.
- One for row and another for column.

Syntax of Two dimensional arrays:

data-type array-name[row-size][column-size];

Example:

int a[3][4];

CONT…

Initialization of Two dimensional arrays:

In Two Dimensional array, all the elements are stored row wise.

```
int a[2][3]={{1},{2,2}};
```

Another way to define 2-D array is

```
int A[3][3]={
                {11,12,13},
                {14,15,16},
                {17,18,19}

            }
```

CONT…

Example:

```
int a[3][3],i,j;
clrscr();
printf("Enter element into 3 X 3 matrix A :");
        for(i=0;i<3;i++)
        {
        for(j=0;j<3;j++)
        {
                scanf("%d",&a[i][j]);
        }
        }
```

Matrix addition operation

	0	1	2	3
0	0	1	0	2
1	-1	2	4	3
2	0	-1	3	1

+

	0	1	2	3
0	3	-1	3	1
1	1	4	2	0
2	2	1	1	3

=

	0	1	2	3
0	3	0	3	3
1	0	6	6	3
2	2	0	4	4

Program to add Two Matrices of size 3 x 3

```
#include<stdio.h>
#include<conio.h>
void main()
{
int a[3][3],b[3][3],c[3][3],i,j;
clrscr();
printf("Enter element into 3 X 3 matrix A :");
for(i=0;i<3;i++)
{
        for(j=0;j<3;j++)
        {
                scanf("%d",&a[i][j]);
        }
}
```

CONT…

```
printf("Enter element into 3 X 3 matrix B :");
        for(i=0;i<3;i++)
        {
                for(j=0;j<3;j++)
                {
                        scanf("%d",&b[i][j]);
                }
                }
        for(i=0;i<3;i++)
                {
                for(j=0;j<3;j++)
                {
                        c[i][j]=a[i][j]+b[i][j];
                }
                }
```

CONT…

```
printf("Elements of Matrix C is given below :");
        for(i=0;i<3;i++)
        {
        for(j=0;j<3;j++)
        {
                printf(" %3d",c[i][j]);
        }
        printf("\n");
        }
        getch();
        }
```

1.4 Multi-Dimensional Arrays

- A multi-dimensional array is an array that has more than one dimension.
- It is an array of arrays; an array that has multiple levels.

Syntax:

Data_type array_name[size1][size2]....[sizeN];

data_type: Type of data to be stored in the array. Here data_type is valid C/C++ data type
array_name: Name of the array
size1, size2,... ,sizeN: Sizes of the dimensions

Example: int a[5][10][20];
array **int a[5][10][20]** can store total (5*10*20) = 1000 elements.

CONT…

Example: int a[3][3][3];

	Column 0	Column 1	Column 2
Row 0	two_d[0][0]	two_d[0][1]	two_d[0][2]
Row 1	two_d[1][0]	two_d[1][1]	two_d[1][2]
Row 2	two_d[2][0]	two_d[2][1]	two_d[2][2]

1.5 SSCANF() AND SPRINTF() FUNCTIONS

- **sscanf() function** is used to extract strings from the given string.

Syntax:

sscanf(characterArray, "Conversion specifier", variables);

- This will extract the data from the character array according to the conversion specifier and store into the respective variables.

- sscanf() will read subsequent characters until a whitespace is found (whitespace characters are blank, newline and tab).

CONT…

Program of sscanf( ) function.

```
#include<stdio.h>
#include<conio.h>
void main( )
{
char name[50]={“SACHIN RAMESH TENDULKAR”};
char fname[10],mname[20],lname[10];
sscanf(name,”%s %s %s”,fname,mname,lname);
printf(“First Name = %s“,fname);
printf(“Middle Name = %s“,mname);
printf(“Last Name = %s“,lname);
getch();
}
```

CONT…

sprintf() function is exactly opposite to sscanf() function. Sprint() function writes the formatted text to a character array.

Syntax:

sprintf (CharacterArray,"Conversion Specifier", variables);

CONT…

Program for sprintf( ) function.

```
#include<stdio.h>
#include<conio.h>
void main( )
{
char name[50];
char fname[10]= {“SACHIN”};
char mname[20]={“RAMESH”};
char lname[10]={“TENDULKAR”};
sprintf(name,”%s %s %s”,fname,mname,lname);
printf(“Full Name = %s”,name);
getch();
}
```

1.6 Drawbacks of Linear array

- The size of array mast is constant or known at compile time.
- Array cannot be copied or compared, because they are pointers.
- Array has static structures.
- The operation of Array is more complex and time consuming like insertion, deletion, searching and sorting.
- Array index type must be integer.