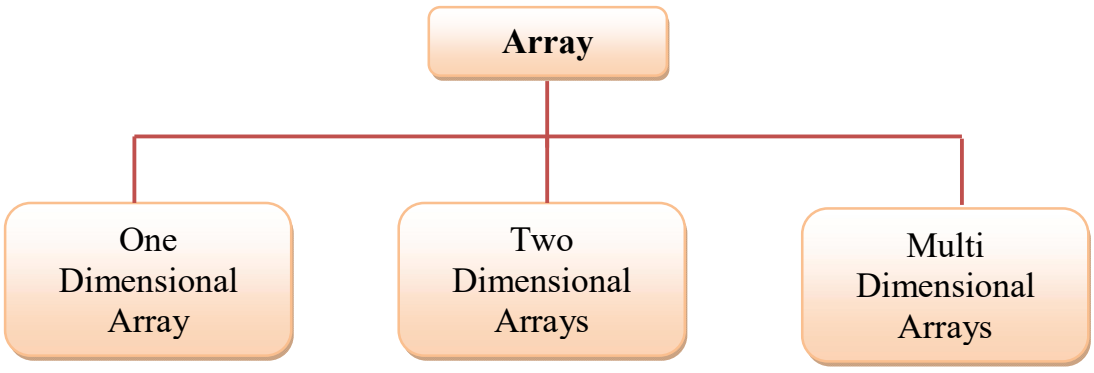
Chapter No: 4
Array & String

Chapter No: 4	Array & String
1	What is array? Types of array?
2	How can we define single dimensional array?
3	Explain two dimensional arrays with example.
4	What is string? How can we enter string in C program?
5	Explain different string function with example.

Q-1. What is array? Explain types of array.
Answer:

Simple variable can store only one value at a time, but if you want to store more than one value into single variable then C language provide derived data type array.
Array is collection of elements of the same data type.
Array is one kind of data structure in which we can store multiple values at time and it is known by a single name.
Each element into array can distinguish with help of index.

There are three types of array in C.



For example if you want to store rollno of single student then simple variable can be used, but if you want to store rollno of 100 students then there are two option.

1) One may use 100 variables to store 100 values
2) One may use array with size of 100

Obviously you can say that second option is far better as we just need to remember only one name of array rather than the 100 name of variables.

Q-2. Explain one dimensional array with example.
Answer:

An array that represent list of items with help of only one index (subscript) then it is known as one dimensional array.

Syntax:

<Data type> <array name> [<size>];

Here, Data type represent the type of data which we want to store into array.
Array name is the name of array which we want to create.
Size defines the size of array or total number of value that we want to store into array.

Array can be initialized in two different ways:

1) Compile time initialization
2) Run time initialization

1) Compile Time Initialization:

In this type of initialization elements of array can be initialized at the time of array declaration.
The values are assigned to each array elements enclosed within braces and separated by comma.

Syntax:

<Data type> <array name> [<size>] = {value1, value2 … value3};

a[0]	a[1]		a[n]
Value 1	**Value 2**	**….**	**Value n**

Example:

```
#include<stdio.h>
#include<conio.h>
void main()
{
        int a[5] = {1,2,3,4,5};
        int i;
        clrscr();
        for(i=0;i<5;i++)
        {
                printf("%d", a[i]);
        }
        getch();
}
```

Note:
When we are initializing the values at the time of declaration, then it is not required to specify the size of array.

a[0]	a[1]	a[2]	a[3]	a[4]
1	**2**	**3**	**4**	**5**

2) Run Time Initialization:

hen user enters the value of array at runtime then it is known as run time initialization.

Using scanf() we can enter the value of array at runtime.

Size of array should be either a numeric constant or symbolic constant.

Example:

```
#include<stdio.h>
#include<conio.h>
void main()
{
        int a[5];
        int i;
        clrscr();
```

```
        for(i=0;i<5;i++)
        {
                printf("enter value");
                scanf("%d",&a[i]);
        }
        for(i=0;i<5;i++)
        {
                printf("%d",a[i]);
        }
        getch();
}
```

Q-3. Explain two dimensional arrays with example.
Answer:

Array which represents the list of elements using two indexes (Subscript) is known as two dimensional arrays.
In two dimensional arrays, the data is stored into rows and columns format.

Syntax:

<Data type> <array name> [<row size>][<column size>];

Here, data type represents the type of data that user want to store into array.
Array name represent the name of array that user want to create.
Row size represents the size of row in array and column size represents the size of column in array.

Example:

```
#include<stdio.h>
#include<conio.h>
void main()
{
        int a[3][3],i,j;
        for(i=0;i<3;i++)
        {
                for(j=0;j<3;j++)
                {
                        scanf("%d",&a[i][j]);
```

```
            }
        }
        for(i=0;i<3;i++)
        {
            for(j=0;j<3;j++)
                printf("%d",a[i][j]);
        }
}
```

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

Consider following example for two dimensional arrays.

Example:

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

Total number of elements in two dimensional array = (row size * column size).

Q-4. What is string? How we can enter string?
Answer:

String is collection characters. C language does not have string as a data type so user needs to create array of characters for storing string value.

Syntax:

char <string name> [<size>];

For declaration of string data type must be character. Here string name represent the name of string that user want to create. Size define the total number of characters that user want to store as string.

Compiler automatically supplies a null character ('\0') at the end of each string. We can assign string at two different levels:

1) Compile Time
2) Run Time

1) Compile Time:

Compile time initialization is done during declaration of string.

Example:

```
char name[4] = {"vvp"};
char name[4] = {'v', 'v', 'p', '\0'};
```

Compile add null value at the end of string. So size of string is greater than one to the original size of string value that user want to store.

```
char name[] = "Vvp Engineering College";
char name[3] = "Good" ; //error
char name[5];
name = "Good"; //error
```

2) Run Time:

String can be entered at run time in three different ways:

1) Using scanf()
2) Using getchar()
3) Using gets()

1) Using scanf():

Using %s format specification we can use scanf() to get string value. But using scanf() we can't get the string with white space character.

Syntax:

scanf("%s", <string name>);

Note:
In case of character array, the ampersand (&) is not required before variable name.

2) Using getchar():

getchar() is used to get the character value from user. It reads only one character at a time. So to read multiple characters or series of characters need to use loop.

```
char ch[5];
ch[0] = getchar();
```

3) Using gets():

This function is used to read the entire string from user.
Using this function we can enter string with white space characters.
It reads the value from user until new line character is not entered.
gets() can get only one string at time, scanf() can get multiple string at time.

Example:

```
#include<stdio.h>
#include<conio.h>
void main()
{
        char str[20],ch;
        printf("\nenter the value through gets");
        gets(str);
        printf("\nEntered value through get is %s",str);
        printf("\nenter the string through scanf");
        scanf("%s",str);
        printf("\nEntered value through scanf is : %s",str);
}
```

Q-5. Explain different string function with example.
Answer:

Consider following different string functions of C language.

1) strlen()
2) strcat()
3) strcpy()
4) strcmp()
5) strlwr()
6) strupr()
7) strstr()

1) strlen():

This function is used to identify the length of specified string.
Length means total number of character that a string contains.

Syntax:

```
n = strlen(string);
```

Here n is integer variable, which can store the total number of characters.

Argument may be string constant or any string variable name.

2) strcat():

This function is used to concate or joins the two strings.

Syntax:

```
strcat(string1, string 2);
```

Here string 1 and string 2 both are characters array. When this function is executed, string 2 is appended at the end of string 1. The null value of string 1 is removed and first character of string 2 is starts to append from that position.

3) strcmp():

This function is used to compare two strings.

Syntax:

```
strcmp(string1, string 2);
```

It both strings are same then function returns 0, if string 1 is greater than string 2 then function returns greater than 0 and if string 1 is smaller than string 2 then function returns less than 0 value.
Comparison is based on the ASCII value of each character.

For example:

strcmp("their", "there");

This function returns -9, because first 3 characters of both strings are same but difference of 4th character of both string is -9.

4) **strcpy():**

This function is used to copy one string into another string.

Syntax:

strcpy(string1, string 2);

It just assigns the value of string 2 to the string 1. If string 1 contains some value then its value is overwrite by string 2.
It's recommended that, size of string 1 is enough larger to get the value of string 2.

5) **strlwr():**

This function is used to convert string into lower case or small letters.

Syntax:

strlwr(string);

If specified string is in already lower case or small letters then it remains as it is.

6) **strupr():**

This function is used to convert string into lower case or small letters.

Syntax:

strlwr(string);

If specified string is in already lower case or small letters then it remains as it is.

7) strstr():

This function is used to find the position of sub string.

Syntax:

strstr(string1,string2);

This function search string 1 to see whether the string s2 is contained in string 1.

If string is found then it returns the position of the first occurrence of the substring. Otherwise it returns the null.

Example:

```
#include<stdio.h>
#include<conio.h>
void main()
{
        char s1[20], s2[20]; int i;
        printf("Enter first string");
        gets(s1);
        printf("'Enter second string");
        scanf("%s",s2);
        i=strlen(s1);
        printf("\nLength of string %d",i);
        printf("\nString joint %s",strcat(s1,s2));
        printf("\nString copy %s",strcpy(s1,s2));
        i=strcmp(s1,s2);
        if(i>0)
                printf("\nString 1 is greater than string 2");
        else if(i<0)
                printf("\nString 1 is less than string 2");
        else
                printf("\n Both are equal");
        printf("\nString lower %s",strlwr(s1));
        printf("\nString upper %s",strupr(s2));
}
```