CS8251- PROGRAMMING IN C

UNIT II

1. **Define array. Explain in detail about array.**

- Array is a collection of similar data elements. All the elements have the same datatype.
- The elements of the array are stored in consecutive memory locations.
- These memory locations are referenced by an index. It is called as subscript
- Subscript is a number which is used to identify an element of the array.

Declaring Array:

Declaring array involve three things.

a. Specifying data type.
b. Name of the array.
c. Size of the array.

Syntax:

```
datatype name[size];
```

Example:

```
int marks[5];
```

Initialization of Array.

Initialization can be done by two ways.

a. Compile Time Initialization
b. Run Time Initialization

Compile Time Initialization.

- Elements of the array can be initialized at the time of declaration

Syntax

```
datatype name[size] = {list of values};
```

Example:

```
int marks[3] = {10,20,30};
        or
int marks[] = {10,20,30};
```

Run Time Initialization

- Array can be initialized at run-time using scanf ().
- This approach is usually used for initializing large array or to initialize array with specified values.

Example:

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

Program: To read and display 'n' numbers using array.

```
#include <stdio.h>
#include <conio.h>
int main()
{
  int i=0, n, arr[20];
  printf("Enter the number of elements");
  scanf("%d", &n);
  printf("Enter the values");
  for(i=0; i<n; i++)
  {
     scanf("%d", &arr[i]);
  }
  printf("The array elements are");
  for(i=0; i<n; i++)
  {
     printf("%d", arr[i]);
  }
}
```

---

2. Explain Two Dimensional Array in detail.

- Two Dimensional array is specified using two subscripts where one subscript specifies the row, other subscript specifies the column.

Syntax:

```
datatype  name[row][column];
```

Example:

```
int marks[5][5];
```

Program:

```
#include <stdio.h>
#include <conio.h>
void main()
{
   int stud[4][2];
   int i;
   for(i=0; i<=3; i++)
   {
      printf("Enter the student roll no and mark");
      scanf("%d %d", &stud[i][0], &stud[i][1]);
   }
   printf("The student mark details are");
   for(i=0; i<=3; i++)
   {
      printf("Student Roll no and marks", stud[i][0],
                                          stud[i][1]);
}}
```

3. Explain the searching techniques using array in detail.

- Searching is the process of finding whether a particular value is available in the array or not.
- The two methods of searching are
  a. Linear Search
  b. Binary Search.

Linear Search

- Linear Search is called as sequential search.
- Searching process compares each and every element of the array one by one in sequence until a match is found.

→ Program:

```
#include <stdio.h>
#include <conio.h>
void main()
{
    int a[20], i, x, n;
    printf("Enter the total number of elements");
    scanf("%d", &n);
    printf("Enter the elements");
    for(i=0; i<n; i++)
    {
        scanf("%d", &a[i]);
    }
    printf("Enter the element to search");
    scanf("%d", &x);
    // Searching process
    for(i=0; i<n; i++)
        if(a[i]==x)
            break;
        if(i<n)
            printf("Element found at index %d", i);
        else
            printf("Element not found");
    getch();
}
```

Binary Search

- Binary Search is a searching technique that works efficiently with sorted arrays.

Steps to be performed in binary search:

1. Find the middle element from the given array.
2. Compare the element to be searched (key) with the middle element.
3. If the key is smaller than the middle element, then continue the searching in the left side of the middle element.
4. If the key is greater than the middle element, then continue the searching in the right side of the middle element.
5. Repeat the above steps until the element is found.

Program:

```
#include <stdio.h>
int main()
{
    int c, first, middle, n, search, array[100];
    printf("Enter number of elements");
    scanf("%d", &n);
    printf("Enter the values");
    for(c=0; c<n; c++)
        scanf("%d", &array[c]);
    printf("Enter value to find");
    scanf("%d", &search);
    first= 0;
    last = n-1;
    middle= (first + last)/2;
    while(first <= last)
    {
      if(array[middle] < search)
          first= middle + 1;
      else if(array[middle] == search)
          printf("%d found at location %d\n", search, middle+1);
          break;
    }
    else
        last= middle - 1; middle = (first + last)/2; }
    if(first > last)
        printf("Not found");
    return 0;
}
```

4. Explain in detail about strings and its functions.

- String is a sequence of characters treated as a single unit. The group of character, digits and symbols enclosed within quotation marks called as strings.
- C programming gives us strings with null-terminated character.
- Null character is used to indicate the end of the string.
- String is a sequence of characters treated as a single unit.

Eg: `char name [ ] = {'R', 'A', 'M'};`

or

`char name [3] = {'R', 'A', 'M'};`

The following table illustrates the string functions.

DESCRIPTION	SYNTAX	CODE

strlen():
• Used to find the length of the string
	var = strlen(strvar);	void main()
		{
		int len;
		char name[10];
		printf("Enter name");
		scanf("%s", name);
		len = strlen(name);
		printf("length is %d", len);
		}
		Output: Enter name: John
		length is 4.

• strcpy()
• Used to copy the value of one string to another string.
	strcpy(destination, source);
	
	Value of source is copied into destination
		void main()
		{
		char str1[10], str2[10];
		printf("Enter any string");
		scanf("%s", str1);
		strcpy(str2, str1);
		printf("The copied string is %s", str2);
		}
		Output: Enter any string:
		welcome
		The copied string is welcome.

strcat()
• Used to concatenate two strings
	strcat(source, destination)
	
	Value of dest is joined along with source and the result is stored in source.
		void main()
		{
		char str1[10], str2[10];
		printf("Enter strings");
		scanf("%s %s", str1, str2);
		strcat(str1, str2);
		printf("The concatenated string is %s", str1);
		}
		Output: Enter strings
		hi
		hello
		The concatenated string is hihello

DESCRIPTION	SYNTAX	CODE

strcmp()
Used to compare two strings.
	strcmp(str1, str2)==0
	This condition must be checked
		void main()
		{
		char str1[10], str2[10];
		printf(" Enter values for Strings");
		scanf("%s %s", str1, str2);
		if (strcmp (str1, str2)==0)
		printf(" Strings are equal");
		else
		printf(" Strings are not equal.

strrev()
Used to reverse a String
	strrev(str var)	void main()
		{
		char str1[10], str2[10];
		printf(" Enter any string");
		scanf("%s", str1);
		str2 = strrev(str1);
		printf(" The reversed string is %s", str2);

strlwr()
Used to convert the upper case into lower case.
	strlwr(strvar).	void main()
		{
		char str1[10]; str2[10];
		printf(" Enter String");
		scanf("%s", str1);
		str2 = strlwr(str1);
		printf(" Converted String is %s", str2);
		}

strupr()
Used to convert the lower case into upper case.
	strupr (strvar)	void main()
		{
		char str1[10], str2[10];
		printf(" Enter String");
		scanf("%s", str1);
		str2 = strupr (str2);
		printf(" Converted String is %s", str2);

5. Explain Sorting concept with an example program

Sorting is any process of arranging items systematically.

Program to sort numbers in ascending order.

```
#include <stdio.h>
void main()
{
  int a[10], i, j, temp, n;
  printf("Enter the total no of. elements");
  scanf("%d", &n);
  printf("Enter the elements");
  for(i=0; i<n; i++)
  { scanf("%d", &a[i]);
  }
  for(i=0; i<n; i++)
  {
    for(j=i+1; j<n; j++)
    {
      if(a[i]> a[j])
      {
        temp = a[i];
        a[i]= a[j];
        a[j]= temp;
      }
    }
  }
  for(i=0; i<n; i++)
  {
    printf("%d\t", a[i]);
  }
  getch();
}
```