STRUCTURES

Definition, Declaration, accessing structures, initialization, operations on structures, structures containing arrays, structures containing pointers, nested structures, self-referential structures, arrays of structures, structures and functions, structures and pointers.

Array of Structures

- ✓ In array of structures, the variable of structure is array .
- ✓ In our sample program, to store details of 100 students we would be required to use 100 different structure variables from **s1** to **s100**,which is definitely not very convenient. A better approach would be to use an array of structures.

Syntax for declaring structure array

```
struct  struct-name
{
datatype var1;
datatype var2;
datatype varN;
};
struct struct-name structure_variable [ size ];
```

Sample Program:

Define a structure for student which include roll number ,name,age and marks . Write a program to read and display the information of 'n' number of students where n is value supplied by user.

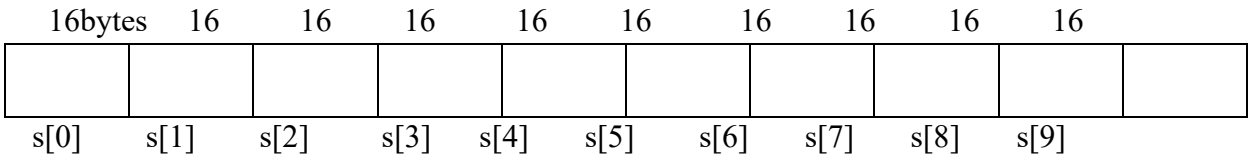
```
#include<stdio.h>
#include<string.h>
  struct student
  {
  int rno;
  char name[10];
  int marks,age;
  };
 void main()
 {
 struct student s[10];//Declares array of 10 student.
 int i,n;
 printf("\n enter number of students: ");
 scanf("%d",&n);
   for(i=0;i<n;i++)
   {
   //reading values for s3 using standard input function.
   printf("\n enter rno,name ,marks,age of student %d: ",i+1);
   scanf("%d%s%d%d",&s[i].rno,s[i].name,&s[i].marks,&s[i].age);
```

```
    }
  printf("\n\n");
  printf("Details of students are :\n");
    for(i=0;i<n;i++)
    {
    printf("\n Details of student %d:\n",i+1); printf("\n
    roll number: %d",s[i].rno); printf("\n name
    :%s",s[i].name);
    printf("\n marks: %d",s[i].marks);
    printf("\n age: %d",s[i].age);
    }
  }
```

Output:

```
enter number of students :3
enter rno,name,marks,age of student 1:
2
Gandhi
89
18

enter rno,name,marks,age of student 2:
5
Raj
76
18

enter rno,name,marks,age of student 3:
6
Ram
86
18

Details of student 1:
roll number: 2
name :Gandhi
marks :89
age: 18

Details of student 2:
roll number: 5
name :Raj
marks :76
age: 18

Details of student 3:
roll number: 6
name :Ram
marks :86
age: 18
```

- ✓ *In the above program the memory allocated for structure variable is 160 bytes consecutively in which first 16 bytes for 1st student 1(s[0]),next 16 bytes for 2nd student 2(s[1]) and so on last 16 bytes for 10 th student (s[9]).*
  - ✓ The following figure shows the memory allocation for array of structures.



  - ✓ Again, in each 16 bytes (2 bytes-roll number,10 bytes-name,2 bytes-marks,2 bytes-age).

Self Referential Structures:

- ✓ Self Referential structures are those structures that have one or more pointers which point to the same type of structure, as their member.
- ✓ In other words, structures pointing to the same type of structures are self-referential in nature.

Syntax:

```
struct node
{
    data_type variable_list;
    struct node* pointer_variable;
};
```

Example:

```
//Program for Self Referential Structures Demo.
#include <stdio.h>

struct node
{
    int data;
    struct node *link;
};
int main()
{
    struct node n1,n2;

    n1.data=10;
    n1.link=NULL;

    n2.data=20;
    n2.link=NULL;

    n1.link=&n2;

    printf("N2 data is : %d\n",n2.data);
    printf("N2 data using N1: %d\n",n1.link->data);

    return 0;
}
```