Structures

A structure is a collection of variables under a single name. These variables can be of different types, and each has a name which is used to select it from the structure.

- Array is a collection of same type of elements but in many real life applications we may need to group different types of logically related data.
- For eg, if we want to create a record of a person that contains name, age and height of that person, then we can't use array because all the three data elements are of different types.
- Record{ name(string),age(int), height(int)}
- To store these related fields of different data types we can use a structure, which is cable of storing heterogenous data.
- Data of different types can be grouped together under a single name using structures.
- Structure which allows you to wrap one or more variables with different data types.
- A structure can contain any valid data types like int, char, float even arrays or even other structures.
- Each variable in structure is called a structure member.

Defining a Structure

- To define a structure, you use struct keyword.

```
struct tagname{
        datatype member1;
        datatype member2;
        ….
        datatype memberN;
                };
```

- Here struct is a keyword which tells the compiler that a structure is being defined.
- Member1,member2…memberN are members of structure and are declared inside curly braces.
- There should be semicolon at the end of curly braces.

example

```
struct student{
        char name[20];
        int rollno;
        float marks;
        };
```

- Here student is a structure and there are three members i.e name ,rollno and marks.

Declaring structure variables

- By defining a structure we have only created a format, the actual use of structure will be when we declare variables based on this format.

```
struct student stu1,stu2;
```

- Here stu1 and stu2 are structure variables that are declared using the structure tag student.
- Declaring a structure variable reserves space in memory.
- Each structure variable declared to be of type struct student has three members i.e name ,rollno, marks.
- The compiler will reserve space to each variable sufficient to hold all the members.
- For eg: each variable of type struct student will occupy 26(20+2+4)bytes.

Initialization of structure variables

```
struct student{
                    char name[20];
                    int rollno;
                    float marks;
                    };

struct student stu1={“mary”,12,78};
```

Accessing members of structures

- Name of stu1 is given by-stu1.name
- Rollno of stu1 is given by-stu1.rollno
- Marks of stu1 is given by-stu1.marks
- Name of stu2 is given by-stu2.name
- Rollno of stu2 is given by-stu2.rollno
- Marks of stu2 is given by-stu2.marks

WAP to display the values of structure members

```
#include<stdio.h>
#include<conio.h>
struct student{
        char name[20];
        int rollno;
        float marks;
        };
void main()
{
struct student stu1={“Amit”,23,76};
struct student stu2;
clrscr();
printf(“enter the value of stu2”);
scanf(“%s%d%f”,stu2.name,&stu2.rollno,&stu2.marks);
printf(“stu1=%s%d%f”,stu1.name,stu1.rollno,stu1.marks);
printf(“stu2=%s%d%f”,stu2.name,stu2.rollno,stu2.marks);
getch();
}
```

Array of structure

- We know that array is a collection of elements of same type.
- We can declare array of structures where each element of array is of structure type.
- Array of structure can be declared as –
- Struct student stu[10];
- Here stu is an array of 10 elements, each of which is a structure of type struct student ,means each element of stu has 3 members, which are name, rollno and marks.
- These structures can be accessed through subscript notation.
- To access the individual members of these structures we'll use the dot operator.

stu0.name stu0.rollno stu0.marks

stu1.name stu1.rollno stu1.marks

……

stu9.name stu9.rollno stu9.marks

WAP to display the values of structure using arrays

```
#include<stdio.h>
#include<conio.h>
struct student{
        char name[20];
        int rollno;
        float marks;
        };
void main()
{
struct student stu[10];
int i;
for(i=0;i<10;i++)
{
    printf("enter the value");
    scanf("%s%d%f",stu[i].name,&stu[i].rollno,&stu[i].marks);
}
for(i=0;i<10;i++)
    printf("%s%d%f",stu[i].name,stu[i].rollno,stu[i].marks);
getch();
}
```

```
#include<stdio.h>
#include<conio.h>
struct student{
        char name[20];
        int rollno;
        float marks[4];
        };
void main()
{
struct student stu[10];
int i;
for(i=0;i<10;i++)
{
    printf("enter the value");
    scanf("%s%d",stu[i].name,&stu[i].rollno);
    for(j=0;j<4;j++)
    scanf("%f",&stu[i].marks[j]);
}
for(i=0;i<10;i++)
{
    printf("%s%d",stu[i].name,stu[i].rollno);
    for(j=0;j<4;j++)
    printf("%f",stu[i].marks[j]);
```