Structure

StructuresStructure Definition, Structure Initialization, Arrays of Structures, Arrays within Structures, Structures within Structures, Passing Structures to Functions.

Structure definition:-

Int a=5; These are premitive data type and can hole only hold

Float b=5.5; only one value at a time

Char name[]="Anil";

Int arr[5]={1,2,3,4,5}; this can hold more than one value but only

Similar data type.

If we want to store and use more than one value and of different data types then we use structure.

Structure is a user defined data type. (Derived)

Using structure we can define data types which holds more than one element of different data types.

Structure:- A structure **is a one data type of user defined that contains a number of data types grouped together. These data may or may not be of same type.**

We can define a structure as follows:-

or Declaration of structure

struct book;

keyword <identity>

{

Data type element 1

Data type element 2

…………………………..

…………………………..

};

Example

Struct employee

{

Int e_number;

Char name[20];

Float salary;

};

accessing structure members in c

1.Array elements are accessed using the Subscript variable,
   Similarly Structure members are accessed using dot {.}operator.

2. (.) is called as "Structure member Operator".

3.Use this Operator in between **"Structure name"** & **"member name"**

Note :- Do not forget to place a semicolon after the declaration of structure.

Example:-1

```
struct employee

{

int eno;

char ename[20];

float esal;

char add[30];

};

#include<stdio.h>

#include<conio.h>

void main()

{

struct employee e;

clrscr();

printf("size of structure is %d \n",sizeof e);

printf("size of structure is %d \n",sizeof(struct employee));

printf("address of structure employee= %u \n",&e);

printf("address of structure employee= %u \n",&e.eno);

printf("size of int in structure is %d \n",sizeof e.eno);

printf("size of char in structure is %d \n",sizeof(e.ename));

printf("size of float in  structure is %d \n",sizeof(e.esal));
printf("size of char in structure is %d \n",sizeof(e.add));
printf("address of Name in  structure Employee= %u
\n",&(e.ename));
printf("address of Salary in  structure Employee= %u
\n",&(e.esal));
printf("address of Address in  structure Employee= %u
\n",&(e.add));
getch();
}
```

Example-2 WAP in c to read and display student information using structure.01

```
#include<stdio.h>
#include<conio.h>
struct stud
{
int a;
char name[20];
};
void main()
{
stud x;
clrscr();
printf("Enter Age and Name");
scanf("%d%s",&x.a,x.name);
printf("Age =%d \t Name =%s",x.a,x.name);
getch();
}
```

Example:3 WAP in c to copy the content of structure and display that.(02)

```
#include<stdio.h>
#include<conio.h>
struct stud
{
int a;
char name[20];
};
void main()
{
stud x,y;
clrscr();
printf("Enter Age and Name");

scanf("%d%s",&x.a,x.name);
printf("Age =%d \t Name =%s \n",x.a,x.name);
y=x;

printf("Age =%d \t Name =%s",y.a,y.name);


getch();

}
```

Initialization of structure:-

Int a=5;

A structure can be initialized in the same way as other data types are initialized. Initializing a structure means assigning some constant to the member of structure. When the user does not explicitly initialized the structure then c automatically does it. For int and float members the values are initialized to 0 and 0.0 and character and string member are initialized to '\0' by default.

The initializers are enclosed in braces and are separated by commas. However care must be taken to ensure that initializers match their corresponding types in the structure definition.

The general syntax to initialize a structure variable is as follows.

```
Struct struct_name
  {
Data type member_name 1
Data type member_name 2
……………………………………..
……………………………………….
}; struct_var={constant 1, constant 2,………. constant n};
```

Example

We can initialize a student structure by writing

```
Struct student

{

Int roll;

Char name[20];

Char course[10];

Float fees;

} stud s={01,”Rahul”,”BCA”,13500};
```

OR

```
Struct student stud 1={01,”Rahul”,”BCA”,13500};
```

Illustration by figure

How the values will be assigned to individual fields of the structure.

1. **Struct student stud 1={01, ”Rahul”, ”BCA”, 13500};**

01	Rahul	BCA	13500
Roll	**name**	**course**	**fees**

2. **Struct student stud 1={07, ”Rahul”};**

07	Rahul	\0	0.0
Roll	**name**	**course**	**fees**

When all the members of a structure are not initialize. It is called partial initialization. In case of partial initialization first few members of structure are initialized and those that are uninitilized are assigned default values.

Struct1

```
struct employee

{

int eno;

char ename[20];

float esal;

char add[30];

};

#include<stdio.h>

#include<conio.h>

void main()

{

struct employee e={1002,"Radha",400.50,"Ranchi College,Ranchi"};

clrscr();

printf("Employee Details \n");
printf("%d\n",e.eno);

printf("%s \n",e.ename);

printf("%f \n",e.esal);

printf("%s",e.add);

getch();

}
```

We can also write this program in this way(all printf statement can write in one statement).

```
struct employee

{

int eno;

char ename[20];

float esal;

char add[30];

};

#include<stdio.h>

#include<conio.h>

void main()

{

struct employee e={1002,"Radha",400.50,"Ranchi College,Ranchi"};

clrscr();

printf("%d\n %s \n %f \n %s",e.eno,e.ename,e.esal,e.add);
getch();
}
```

Array of structure

Array is a collection of similar data types which themselves are a collection of dissimilar data types. We can use array inside structure.

Struct4

```
#include<stdio.h>
#include<conio.h>
struct stud
{
int a;
char n[10];
};

void main()
{
struct stud x [2];
clrscr();
int i;
for(i=0;i<2;i++)
{
printf("Enter Age and Name\n");
scanf("%d\n %s",&x[i].a,&x[i].n);
}
printf("Age and Name \n");
for(i=0;i<2;i++)
{
printf("Age=%d \nName= %s \n",x[i].a,x[i].n);
}
getch();
}
```

Array within structure

Struct6

```
#include<stdio.h>
#include<conio.h>
struct student
{
int roll;
char name[20];
int marks[3];
};
void main()
{
```

```
clrscr();
struct student s;
int i;
printf("Enter Student Roll:-");
scanf("%d",&s.roll);
printf("Enter Student Name:-");
scanf("%s",s.name);
printf("Enter 3 sub  Marks:-");
for(i=0;i<3;i++)
{
scanf("%d",&s.marks[i]);
}
printf("Roll = %d \n",s.roll);
printf("Name = %s \n",s.name);
printf("Marks of student:-");
for(i=0;i<3;i++)
{
        printf("%d \t",s.marks[i]);
}
getch();
}
```

```
};
```

In the above example , we have created an array marks[] inside structure representing 3 marks of a single student. Marks [] is now a member of structure student and to access marks[] we have used dot (.) operator along with object s.

Structure within structure (Nested structure)

One structure can be nested within another structure. A structure may contain another structure as its member. A structure that contains another as its member is called a nested structure.

```
#include<stdio.h>

#include<conio.h>

void main()

{

clrscr();

struct dob

{

int day;

int month;

int year;

};

struct student
```

```
{
int roll;
char name[20];
float fees;
struct dob date;
};
struct student stud;
printf("Enter Roll No");
scanf("%d",&stud.roll);
printf("Enter Name");
scanf("%s",stud.name);
printf("Enter fees");
scanf("%f",&stud.fees);
printf("Enter the DOB");
scanf("%d%d%d",&stud.date.day,&stud.date.month,&stud.d
ate.year);
printf("Student Details");
printf("\n Student Roll no= %d",stud.roll);
printf("\n Student Name= %s",stud.name);
printf("\n Student Fees= %f",stud.fees);
printf("\n Student
DOB=%d/%d/%d",stud.date.day,stud.date.month,stud.date.y
ear);
getch();
}
```

Passing Structures to Functions

Like an ordinary variable , a structure variables can also be passed to a function. We may either pass individual structure elements or the entire structure at one time.

Example of passing individual structure element

Struct9

```
#include<stdio.h>
#include<conio.h>
struct student
{
char name[20];
int roll;
int marks;
};
void display(char name[],int roll,int marks);  // function prototype
void main()
{
clrscr();
struct student stu={"Sudhir",20,80};    // structure initilization
display(stu.name,stu.roll,stu.marks);    // function Call
getch();
}
void display(char name[],int roll,int marks)  // function definition
{
printf("Name=%s \n",name);
printf("Roll=%d \n",roll);
printf("Marks=%d \n",marks);
}
```

Struct8

<u>Passing an entire structure to a function:-</u>

It can be immediately realised that to pass individual elements would become more tedious as the member of structure elements go on increasing. A better way would be to pass the entire structure variable at a time.

Example of passing an entire structure to a function.

```
#include<stdio.h>

#include<conio.h>

struct employee

{

int code;

char name[20];

float sal;

};

void display(struct employee);

void main()

{

clrscr();
struct employee e;

printf("Enter Employee code");

scanf("%d",&e.code);

printf("Enter Employee Name");

scanf("%s",e.name);

printf("Enter Employee Salary");

scanf("%f",&e.sal);

display(e);

getch();

}

void display(struct employee x)

{

printf("\n Code= %d",x.code);

printf("\n Name= %s",x.name);

printf("\n Salary= %f",x.sal);

}
```