

## Unit 9: Structures and Union

- **Structure** is user defined data type.
- Structure is used to store heterogeneous data under unique name.
- Keyword 'struct' is used to declare structure.
- In other word structure is a convenient tool for handling a group of logically related data items of different data types.
- Following are the different instances of structure.

### Syntax 1.

```
struct structure_name{
    <data-type> member1;
    <data-type> member2;
    -----
    -----
    <data-type>member_n;
};
.
```

- Above syntax show structure definition.

### Syntax 2.

```
struct structure_name{
    <data-type> member1;
    <data-type> member2;
    -----
    -----
    <data-type>member_n;
}struct_var_list;
```

- Above syntax shows, structure definition with structure variable declaration.
- Struct\_var\_list is list of structure variables.

### Syntax 3.

```
struct{
    <data-type> member1;
    <data-type> member2;
    -----
    -----
    <data-type>member_n;
}struct_var_list;
```

- Above syntax shows, structure definition with structure variable declaration.
- Struct\_var\_list is list of structure variables as we have structure variable that's why we can omit structure name.

### Example of Syntax 1 :

```
struct student{
    int rollno;
    char nm[10];
    float wt,ht;
};
For this structure we can declare structure
variable as follows:

struct student s1,s2,s3,s4,s5;
```

### Example of Syntax 2 :

```
struct student{
    int rollno;
    char nm[10];
    float wt,ht;
}s1,s2,s3,s4,s5;
```

### Example of Syntax 3 :

```
struct{
    int rollno;
    char nm[10];
    float wt,ht;
}s1,s2,s3,s4,s5;
```

### Declaration of Structure Variable:

- Once user define data type is created we can declare structure variable.
- **Syntax:**  
struct strucuter\_name structure\_variable list;
- Example:  
struct student s1,s2,s3,s4,s5;
- In above example s1,s2,s3,s4 & s5 are structure variable of type student.

### Structure Definition

```
struct student{  
    int rollno;  
    char nm[10];  
    float wt,ht;  
};
```

Structure Tag or Structure Name

Structure Members or Structure Field or Structure Field.

### Structure Variable Declaration

```
struct student s1,s2,s2,s4,s5;
```

Structure Variables Declaration

New User define Data type

### Accessing Members of a Structure

- C provides two operators to access structure members viz dot & arrow.
- Dot(.) is an operator which is used to work with structure variable.
- Arrow (->) is an another operator, to work with structure with pointer.

#### Syntax:-

Structure\_variablename.member;

Exampe:-

```
s1.rollno;
```

Example for pointer to structure:-

```
p->rollno;
```

### Assigning Values to structure members:

- With the help of assignment operator we can assign value to the structure members.

#### Syntax:-

Structure\_varaible.member=value;

- Example:-

```
s1.rollno=50;  
s1.name="Rohit";  
s1.wt=68.00;  
s1.ht=5.7;
```

### Initialization of structure: -

- We can initialize structure as like as array by specifying list of values in curly bracket.
- **Syntax:-**  
Struct structure\_name variable\_1= { list of values separated by comma };
- **Example:-**  
struct student s1={50,"Rohit",68.00,5.7};
- In above example we have initialized the structure elements by separating with comma in curly bracket.
- But we cannot initialize the structure element during structure definition.

### Reading & Displaying structure Member: -

- We can read & display structure member using scanf( ) & printf( ) respectively.
- Example for reading structure member:  
scanf("%d",&s1.rollno);  
scanf("%s",s1.name);  
scanf("%f%f",&s1.wt,&s1.ht);
- Example for displaying structure member:  
printf("%d",s1.rollno);  
printf("%s",s1.name);  
printf("%f%f",&s1.wt,&s1.ht);

### /\* Program to read & display student information \*/

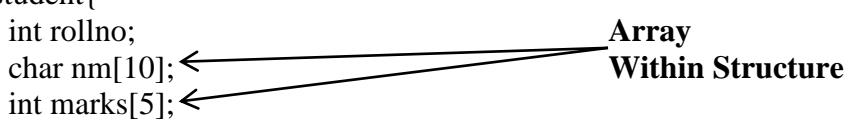
```
#include<stdio.h>
struct student{
    int rollno;
    char nm[10];
    float wt,ht;
};
main( ){
    struct student s;
    printf("Enter Student information\n");
    scanf("%d",s.rollno);
    scanf("%s", &s.nm);
    scanf("%f%f",&s.wt,&s.ht);
    printf("Entered Student Information is\n");
    printf("Roll No =%d\n",s.rollno);
    printf("Name is %s\n",s.nm);
    printf("Weight is %f\n"Height is %f\n",s.wt,s.ht);
}
```

## Array Within structure:-

- Multiple times we need more than one element of structure member for that we can use the array within structure.
- For example, if we have structure members student\_name, & student\_marks etc.

**/\* Program to read & display student information with five subject marks\*/**

```
#include<stdio.h>
struct student{
    int rollno;
    char nm[10];
    int marks[5];
};
main( ){
    struct student s;
    printf("Enter Student information\n");
    scanf("%d",&s.rollno);
    scanf("%s",&s.nm);
    printf("Enter four subject marks\n");
    for(i=0;i<5;i++)
        scanf("%d",&s.m[i]);
    printf("Entered Student Information is\n");
    printf("Roll No =%d\n",s.rollno);
    printf("Name is %s\n",s.nm);
    printf("Marks of five subjects are");
    for(i=0;i<5;i++)
        printf("%d\t",s.m[i]);
}
```



## Array of Structure: -

- Multiple times we need more than one variables of structure for that we can use the array of structure.
- For example, if we have structure of student. We want the list of students of entire class but the one variable of structure can represent only one student. So to represent entire class variable we have to use array of student structure.

### **Syntax:-**

```
struct structure_name arrayname[size];
```

### **Example:-**

```
struct student s[100];
```

**/\* Program to show how array of structure should be initialized \*/**

```
#include<stdio.h>
struct student{
    int rollno;
    char nm[10];
    float wt,ht;
};
main( ){
    int i;
```



```

struct student s[5]={ {5,"Parag",74,5.6},
                      {6,"Ashish",68,5.7},
                      {7,"Kunal",60,5.5},
                      {8,"Aniruddha",5.8},
                      {9,"Pravin",5.6}
};

printf("Initialized information is \n");
for(i=0;i<5;i++){
    printf("Roll No =%d\n",s[i].rollo);
    printf("Name is %s\n",s[i].nm);
    printf("Weight is %f\n", s[i].wt);
    printf("Height is %f\n",s[i].ht);
}
}

```

**/\*Write a structure cricket with following information**

**Player Name, Team Name & Batting average**

**Write a program to read & display information of 14 players of a team \*/**

```

#include<stdio,h>
struct cricket{
    char pname[10];    //player name.
    char tname[10];    //team name.
    float bavg;        //batting average.
};
main( ){
    int i;
    struct cricket c[14];
    for(i=0;i<14;i++){
        printf("Enter information of player %d\n",i+1);
        scanf("%s",&c[i].pname);
        scanf("%s",&c[i].tname);
        scanf("%f",&c[i].bavg);
    }
    for(i=0;i<5;i++){
        printf("Player %d Details is \n",i+1);
        printf("Name is=%d\n",c[i].pname);
        printf("Team Name is %s\n",c[i].tname);
        printf("Batting Avg is %f\n",c[i].bavg);
    }
}

```

## Passing Structure Member to Function

- Like normal variable we can also pass structure member to function.

**/\* Programming Example to pass structure member to function \*/**

```
#include<stdio.h>
struct student{
    int rollno;
    char nm[10];
    int marks[5];
};
int total_mrk(int [ ]);
main( ){
    struct student s;
    int total,avg;
    printf("Enter Student information\n");
    scanf("%d",&s.rollno);
    scanf("%s",&s.nm);
    printf("Enter four subject marks\n");
    for(i=0;i<5;i++)
        scanf("%d",&s.m[i]);
    printf("Entered Student Information is\n");
    printf("Roll No =%d\n",s.rollno);
    printf("Name is %s\n",s.nm);
    printf("Marks of five subjects are");
    for(i=0;i<5;i++)
        printf("%d\t",s.m[i]);
    total=total_mrk(s.m);
    avg=total/5;
    printf("Total marks %d\n",total);
    printf("Average is %d",avg);
}

int total_mrk(int m[5]){
    int i,tot=0;
    for(i=0;i<5;i++)
        tot+=m[i];
    return tot;
}
```

**Array Within Structure**

**Function Call...**

## Union

- Union is a collection of data items of different data types.
- It can hold data of only one member at a time though it has members of different data types(it is scope of union).
- If a union has two members of different data types, they are allocated the same memory(shared memory).
- Syntax of structure & union is same only difference is instead of 'struct' keyword, 'union' keyword is used

### **/\* Programming Example to work with union\*/**

```
#include<stdio.h>
union Test{
    char c;
    int i;
    float f;
};
main(){
    union Test u;
    u.c= 'd';
    printf("%c\n",u.c);
    u.i=10;
    printf("%d\n",u.i);
    u.f=9.81;
    printf("%f",u.f);
}
```

## Difference between Structure & Union.

Structure	Union
A <b>structure</b> is a collection of items of different types; and each data item will have its own memory location.	Where as only one item within the <b>union</b> can be used at any time, because the memory allocated for each item inside the union is in a <i>shared</i> memory location i.e., only one memory location will be shared by the data items of union.
Size of structure is the sum of size of each structure member	Size of union will be the size of the biggest member (variable) of union.
<p>Example:</p> <pre>struct student{     int rollno;     char name[10];     float wt,ht; }s;</pre> <p>Allocated Memory space for 's' will be as follows:</p> <div> <div>rollno</div> <div></div> <div>name</div> <div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div> <div>wt</div> <div></div> <div>ht</div> <div></div> </div> <p>for 's' 16 bytes are allocated.</p>	<p>Example:</p> <pre>union student{     int rollno;     char name[10];     float wt,ht; }u;</pre> <p>Allocated Memory space for 'u' will be as follows:</p> <div> <div>rollno/name/wt/ht</div> <div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div> </div> <p>for 'u' 10 bytes are allocated &amp; that will be shared by all member.</p>



## Difference between Array & Structure.

Array	Structure																				
All data in a array should be of same data type (Homogeneous).	But in structures data can be of different data types (Heterogeneous).																				
Individual entries in an array are called elements.	But in structure individual entries are called members.																				
Array Derived Data Type	Structure is User defined data type.																				
<div>Example:- int arr[10];  Allocated memory space for 'arr' will be as follows:  arr</div> <table><tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr></table>											<div>Example: struct student{     int rollno;     char name[10];     float wt,ht; }; Allocated Memory space for 's' will be as follows:</div> <div><div>rollno</div><div></div><div>name</div><table><tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr></table><div>wt</div><div></div><div>ht</div><div></div><div>for 's' 16 bytes are allocated.</div></div>										

## Pointer to structure...

### C program to show use of pointer to structure....

```
#include<stdio.h>
main(){
    struct date{
        int day;
        char month[10];
        int year;
    };
    struct date dob,*p;
    p=&dob;
    printf("Enter your DOB\n");
    scanf("%d %s %d",&p->day,&p->month,&p->year);
    printf("Ur DOB is: %d-%s-%d",p->day,p->month,p->year);
}
```

Pointer to Structure.

Accessing Structure member using arrow operator.

### Typedef and Structure declaration for bank customer.

#### Structure definition and declaration...

```
struct bank{
    int accno;
    char cust_nm[10];
    char acc_type[10];
    float acc_bal;
};
struct bank cust1, cust2;
```

#### Typedef for bank .....

```
typedef struct bank{
    int accno;
    char cust_nm[10];
    char acc_type[10];
    float acc_bal;
}bank;
bank cust1, cust2;
```

**Refer following program for:-**

- 1. Array With in Structure.**
- 2. Array of Structure.**
- 3. Initializing Array of Structure.**
- 4. Nested Structure.**
- 5. Write a program in C to understand how structure members are sent to a function.**

```
#include<stdio.h>
main( ){
    struct date{
        int dy;
        char mn[10];
        int yr;
    };
    struct student{
        int roll;
        char name[10];
        int mrk[5];
        struct date dob;
    };

    struct student s[5]= {
        {1,"Dhananjay",{67,88,56,37,78},{10,"Dec",1998}},
        {2,"Rushi",{66,87,46,57,88},{25,"Aug",1999}},
        {3,"Kunal",{57,68,86,57,98},{5,"Jun",1998}},
        {4,"Ram",{47,28,76,87,78},{15,"May",2005}},
        {5,"Opal",{47,68,56,57,98},{8,"Nov",2009}},
    }

    int i,avg;
    int sum(int [ ]);
    for(i=0;i<5;i++){
        printf("Roll No: %d",s[i].roll);
        printf("Name: %s",s[i].name);
        avg = sum(s[i].mrk) / 5;
        printf("Percentage : %d",avg);
        printf("DOB: %d-%s-%d ",s[i].dob.dy,s[i].dob.mn,s[i].dob.yr);
    }
}

int sum(int m[5]){
    int s=0,i;
    for(i=0;i<5;i++)
        s+=m[i];
    return (s);
}
```