Sructure

It is the collection of dissimilar data types or heterogenous data types grouped together. It means the data types may or may not be of same type.

Structure declaration-

```
struct tagname

{

Data type member1;

Data type member2;

Data type member3;

………

………

Data type member n;

};
```

OR

```
struct

{

Data type member1;

Data type member2;
```

Data type member3;

………

………

Data type member n;

};

OR

struct tagname

{

struct element 1;

struct element 2;

struct element 3;

………

………

struct element n;

};

Structure variable declaration;

struct student

{

    int age;

char name[20];

char branch[20];

}; struct student s;

Initialization of structure variable-

Like primary variables structure variables can also be initialized when they are declared. Structure templates can be defined locally or globally. If it is local it can be used within that function. If it is global it can be used by all other functions of the program.

We cant initialize structure members while defining the structure

```
struct student
{
        int age=20;
char name[20]=”sona”;
}s1;
```

The above is **invalid.**

A structure can be initialized as

```
struct student
{
        int age,roll;
char name[20];
} struct student s1={16,101,”sona”};
  struct student s2={17,102,”rupa”};
```

If initialiser is less than no.of structure variable, automatically rest values are taken as zero.

Accessing structure elements-

Dot operator is used to access the structure elements. Its associativety is from left to right.

structure variable ;

s1.name[];

s1.roll;

s1.age;

Elements of structure are stored in contiguous memory locations. Value of structure variable can be assigned to another structure variable of same type using assignment operator.

Example:

```
#include<stdio.h>

#include<conio.h>

void main()

{

int roll, age;

char branch;

} s1,s2;

printf("\n enter roll, age, branch=");

scanf("%d %d %c", &s1.roll, &s1.age, &s1.branch);

s2.roll=s1.roll;

printf(" students details=\n");

printf("%d %d %c", s1.roll, s1.age, s1.branch);

printf("%d", s2.roll);
```

```
}
```

Unary, relational, arithmetic, bitwise operators are not allowed within structure variables.

Lecture Note:24

Size of structure-

Size of structure can be found out using sizeof() operator with structure variable name or tag name with keyword.

```
sizeof(struct student); or

sizeof(s1);

sizeof(s2);
```

Size of structure is different in different machines. So size of whole structure may not be equal to sum of size of its members.

Array of structures

When database of any element is used in huge amount, we prefer Array of structures.

Example: suppose we want to maintain data base of 200 students, Array of structures is used.

```
#include<stdio.h>

#include<string.h>

struct student

{
```

```
char name[30];
char branch[25];
int roll;
};
void main()
{
struct student s[200];
int i;
s[i].roll=i+1;
printf("\nEnter information of students:");
for(i=0;i<200;i++)
{
printf("\nEnter the roll no:%d\n",s[i].roll);
printf("\nEnter the name:");
scanf("%s",s[i].name);
printf("\nEnter the branch:");
scanf("%s",s[i].branch);
printf("\n");
}
printf("\nDisplaying information of students:\n\n");
for(i=0;i<200;i++)
{
printf("\n\nInformation for roll no%d:\n",i+1);
printf("\nName:");
puts(s[i].name);
printf("\nBranch:");
puts(s[i].branch);
}
}
```

In Array of structures each element of array is of structure type as in above example.

Array within structures

```
struct student
{
char name[30];
int roll,age,marks[5];
}; struct student s[200];
```

We can also initialize using same syntax as in array.

Nested structure

When a structure is within another structure, it is called Nested structure. A structure variable can be a member of another structure and it is represented as

```
struct student
```

```
{
element 1;
element 2;
………
………
struct student1
{
member 1;
member 2;
}variable 1;
……….
……….
element n;
}variable 2;
```

It is possible to define structure outside & declare its variable inside other structure.

```
struct date
{
int date,month;
};
struct student
{
 char nm[20];
int roll;
struct date d;
}; struct student s1;
   struct student s2,s3;
```

Nested structure may also be initialized at the time of declaration like in above example.

```
struct student s={“name”,200, {date, month}};
                      {“ram”,201, {12,11}};
```

Nesting of structure within itself is not valid. Nesting of structure can be extended to any level.

```
struct time
{
int hr,min;
};
struct day
{
int date,month;
struct time t1;
};
struct student
```

```
{
char nm[20];
struct day d;
}stud1, stud2, stud3;
```

Lecture Note: 25

Passing structure elements to function

We can pass each element of the structure through function but passing individual element is difficult when number of structure element increases. To overcome this, we use to pass the whole structure through function instead of passing individual element.

```
#include<stdio.h>
#include<string.h>
void main()
{
struct student
{
char name[30];
char branch[25];
int roll;
}struct student s;
printf("\n enter name=");
gets(s.name);
printf("\nEnter roll:");
scanf("%d",&s.roll);
printf("\nEnter branch:");
gets(s.branch);
display(name,roll,branch);
}
display(char name, int roll, char branch)
{
printf("\n name=%s,\n roll=%d, \n branch=%s", s.name, s.roll. s.branch);
}
```

Passing entire structure to function

```
#include<stdio.h>
#include<string.h>
struct student
{
char name[30];
int age,roll;
};
display(struct student);                    //passing entire structure
void main()
```

```
{
  struct student s1={"sona",16,101 };
  struct student s2={"rupa",17,102 };
display(s1);
display(s2);
}
display(struct student s)
{
printf("\n name=%s, \n age=%d ,\n roll=%d", s.name, s.age, s.roll);
}
```

```
Output: name=sona
          roll=16
```

Lecture Note: 26

UNION

Union is derived data type contains collection of different data type or dissimilar elements. All definition declaration of union variable and accessing member is similar to structure, but instead of keyword struct the keyword union is used, the main difference between union and structure is