Nested Structure in C

Nested structure in c language can have another structure as a member. There are two ways to define nested structure in c language:
By separate structure
By Embedded structure

---

1) Separate structure

We can create 2 structures, but dependent structure should be used inside the main structure as a member. Let's see the code of nested structure.

```
struct Date
{
  int dd;  // emp1.doj.dd;
  int mm;
  int yyyy;
};
struct Employee
{
  int id;  // emp1.id
  char name[20];
  struct Date doj;
}emp1;
```

As you can see, doj (date of joining) is the variable of type Date. Here doj is used as a member in Employee structure. In this way, we can use Date structure in many structures.

---

2) Embedded structure

We can define structure within the structure also. It requires less code than previous way. But it can't be used in many structures.

```
struct Employee
{
  int id;
  char name[20];
  struct Date
   {
     int dd;
     int mm;
     int yyyy;
   }doj;
}e1;
```

Accessing Nested Structure

We can access the member of nested structure by Outer_Structure.Nested_Structure.member as given below:

```
e1.doj.dd
e1.doj.mm
e1.doj.yyyy
```

C Nested Structure example

Let's see a simple example of nested structure in C language.

```
#include <stdio.h>
#include <string.h>

struct Employee
{
  int id;
  char name[20];
  struct Date
   {
     int dd;
     int mm;
     int yyyy;
   }doj;
}e1;
int main( )
{
  //storing employee information
  e1.id=101;
  strcpy(e1.name, "MCA STUDENT");//copying string into char array
  e1.doj.dd=10;
  e1.doj.mm=11;
  e1.doj.yyyy=2014;

  //printing first employee information
  printf( "employee id : %d\n", e1.id);
  printf( "employee name : %s\n", e1.name);
  printf( "employee date of joining (dd/mm/yyyy) : %d/%d/%d\n", e1.doj.dd,e1.doj.mm,e1.doj.y
yyy);
  return 0;
}
```

Output:

```
employee id : 101
employee name : MCA STUDENT
employee date of joining (dd/mm/yyyy) : 10/11/2014
```

Array of Structures in C

There can be array of structures in C programming to store many information of different data types. The array of structures is also known as collection of structures.

Let's see an example of structure with array that stores information of 5 students and prints it.

```
#include<stdio.h>
#include <string.h>
struct student{
int rollno;
char name[10];
};
int main(){
int i;
struct student st[5];
printf("Enter Records of 5 students");
for(i=0;i<5;i++){
printf("\nEnter Rollno:");
scanf("%d",&st[i].rollno);
printf("\nEnter Name:");
scanf("%s",&st[i].name);
}
printf("\nStudent Information List:");
for(i=0;i<5;i++){
printf("\nRollno:%d, Name:%s",st[i].rollno,st[i].name);
}
  return 0;
}
```

Output:

```
Enter Records of 5 students
Enter Rollno:1
Enter Name:Sonoo
Enter Rollno:2
Enter Name:Ratan
Enter Rollno:3
Enter Name:Vimal
Enter Rollno:4
Enter Name:James
Enter Rollno:5
Enter Name:Sarfraz

Student Information List:
Rollno:1, Name:Sonoo
Rollno:2, Name:Ratan
Rollno:3, Name:Vimal
Rollno:4, Name:James
Rollno:5, Name:Sarfraz
```

C – Passing struct to function

A structure can be passed to any function from main function or from any sub function.
Structure definition will be available within the function only.
It won't be available to other functions unless it is passed to those functions by value or by address(reference).

Else, we have to declare structure variable as global variable. That means, structure variable should be declared outside the main function. So, this structure will be visible to all the functions in a C program.

PASSING STRUCTURE TO FUNCTION IN C:

It can be done in below 3 ways.

Passing structure to a function by value

Passing structure to a function by address(reference)

No need to pass a structure – Declare structure variable as global

EXAMPLE PROGRAM – PASSING STRUCTURE TO FUNCTION IN C BY VALUE:

In this program, the whole structure is passed to another function by value. It means the whole structure is passed to another function with all members and their values. So, this structure can be accessed from called function. This concept is very useful while writing very big programs in C.

```
#include <stdio.h>
#include <string.h>

struct student
{
            int id;
            char name[20];
            float percentage;
};

void func(struct student record);

int main()
{
            struct student record;

            record.id=1;
            strcpy(record.name, "Raju");
            record.percentage = 86.5;

            func(record);
            return 0;
}

void func(struct student record)
{
            printf(" Id is: %d \n", record.id);
            printf(" Name is: %s \n", record.name);
            printf(" Percentage is: %f \n", record.percentage);
}
```

OUTPUT:

```
Id is: 1
Name is: Raju
Percentage is: 86.500000
```

EXAMPLE PROGRAM – PASSING STRUCTURE TO FUNCTION IN C BY ADDRESS:

In this program, the whole structure is passed to another function by address. It means only the address of the structure is passed to another function. The whole structure is not passed to another function with all members and their values. So, this structure can be accessed from called function by its address.

```c
#include <stdio.h>
#include <string.h>

struct student
{
        int id;
        char name[20];
        float percentage;
};

void func(struct student *record);

int main()
{
        struct student record;

        record.id=1;
        strcpy(record.name, "Raju");
        record.percentage = 86.5;

        func(&record);
        return 0;
}

void func(struct student *record)
{
        printf(" Id is: %d \n", record->id);
        printf(" Name is: %s \n", record->name);
        printf(" Percentage is: %f \n", record->percentage);
}
```

OUTPUT:

```
Id is: 1
Name is: Raju
Percentage is: 86.500000
```

EXAMPLE PROGRAM TO DECLARE A STRUCTURE VARIABLE AS GLOBAL IN C:

Structure variables also can be declared as global variables as we declare other variables in C. So, When a structure variable is declared as global, then it is visible to all the functions in a program. In this scenario, we don’t need to pass the structure to any function separately.

```
#include <stdio.h>
#include <string.h>

struct student
{
            int id;
            char name[20];
            float percentage;
};
struct student record; // Global declaration of structure

void structure_demo();

int main()
{
            record.id=1;
            strcpy(record.name, "Raju");
            record.percentage = 86.5;

            structure_demo();
            return 0;
}

void structure_demo()
{
            printf(" Id is: %d \n", record.id);
            printf(" Name is: %s \n", record.name);
            printf(" Percentage is: %f \n", record.percentage);
}
```

OUTPUT:

```
Id is: 1
Name is: Raju
Percentage is: 86.500000
```