Access Structure member using pointer:

There are two ways to access the member of the structure using Structure pointer:

1. Using (*) asterisk or indirection operator and dot (.) operator.
2. Using arrow (->) operator or membership operator.

Program to access the structure member using structure pointer and the dot operator

Let's consider an example to create a Subject structure and access its members using a structure pointer that points to the address of the Subject variable in C.

Pointer.c

```
#include <stdio.h>

// create a structure Subject using the struct keyword
struct Subject
{
   // declare the member of the Course structure
   char sub_name[30];
   int sub_id;
   char sub_duration[50];
   char sub_type[50];
};

int main()
{
   struct Subject sub; // declare the Subject variable
   struct Subject *ptr; // create a pointer variable (*ptr)
   ptr = ⊂ /* ptr variable pointing to the address of the structure variable sub */

   strcpy (sub.sub_name, " Computer Science");
   sub.sub_id = 1201;
   strcpy (sub.sub_duration, "6 Months");
   strcpy (sub.sub_type, " Multiple Choice Question");

   // print the details of the Subject;
   printf (" Subject Name: %s\t ", (*ptr).sub_name);
         printf (" \n Subject Id: %d\t ", (*ptr).sub_id);
      printf (" \n Duration of the Subject: %s\t ", (*ptr).sub_duration);
        printf (" \n Type of the Subject: %s\t ", (*ptr).sub_type);

   return 0;

}
```

Output:

```
Subject Name:  Computer Science
 Subject Id: 1201
 Duration of the Subject: 6 Months
 Type of the Subject:  Multiple Choice Question
```

Nested Structure in C

C provides us the feature of nesting one structure within another structure by using which, complex data types are created.

```
#include<stdio.h>
struct address
{
   char city[20];
   int pin;
   char phone[14];
};
struct employee
{
   char name[20];
   struct address add;
};
void main ()
{
   struct employee emp;
   printf("Enter employee information?\n");
   scanf("%s %s %d %s",emp.name,emp.add.city, &emp.add.pin, emp.add.phone);
   printf("Printing the employee information....\n");
   printf("name: %s\nCity: %s\nPincode: %d\nPhone: %s",emp.name,emp.add.city,emp.add.pin,emp.add.phone);
}
```

The structure can be nested in the following ways.

1. By separate structure
2. By Embedded structure

1) Separate structure

Here, we create two structures, but the dependent structure should be used inside the main structure as a member. Consider the following example.

```
struct Date
{
  int dd;
  int mm;
  int yyyy;
};
struct Employee
{
  int 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

The embedded structure enables us to declare the structure inside the structure. Hence, it requires less line of codes but it can not be used in multiple data structures. Consider the following example.

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

- **C Nested Structure example**

Let's see a simple example of the 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, "Sonoo Jaiswal");//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.yyyy);
  return 0;
}
```

- **Passing structure to function**

Just like other variables, a structure can also be passed to a function. We may pass the structure members into the function or pass the structure variable at once. Consider the following example to pass the structure variable employee to a function display() which is used to display the details of an employee.

```
#include<stdio.h>
struct address
{
  char city[20];
  int pin;
  char phone[14];
};
struct employee
{
  char name[20];
  struct address add;
};
void display(struct employee);
void main ()
{
  struct employee emp;
  printf("Enter employee information?\n");
  scanf("%s %s %d %s",emp.name,emp.add.city, &emp.add.pin, emp.add.phone);
  display(emp);
}
void display(struct employee emp)
{
 printf("Printing the details....\n");
```

```
  printf("%s %s %d %s",emp.name,emp.add.city,emp.add.pin,emp.add.phone);

}
```