STRUCTURES

Definition, Declaration, accessing structures, initialization, operations on structures, structures containing arrays, structures containing pointers, nested structures, self-referential structures, arrays of structures, structures and functions, structures and pointers.

Structure within a Structure

- ✓ Nested structure in C is nothing but structure within structure. One structure can be declared inside other structure as we declare structure members inside a structure.
- ✓ The structure variables can be a normal structure variable or a pointer variable to access the data.
- ✓ This program explains how to use structure within structure in C using normal variable. "student_college_detail' structure is declared inside "student_detail" structure in this program. Both structure variables are normal structure variables.
- ✓ Please note that members of "student_college_detail" structure are accessed by 2 dot(.) operator and members of "student_detail" structure are accessed by single dot(.) operator.

Sample Program 1: Structure within Structure / Nested Structures

```
#include <stdio.h>
#include <string.h>
  struct student_college_detail
  {
    int college_id;
    char college_name[50];
  };
  struct student_detail
  {
    int id;
    char name[20];
    float percentage;
    // structure within structure
    struct student_college_detail clg_data;
  }stu_data;

  int main()
  {
      struct student_detail stu_data = {1, "Raju",90.5, 71145,"GITAM
      University"};
      printf(" Id is: %d \n", stu_data.id);
      printf(" Name is: %s \n", stu_data.name);
      printf(" Percentage is:%f\n\n",stu_data.percentage);
      printf(" College Id is: %d \n", stu_data.clg_data.college_id);
      printf(" College Name is: %s \n",stu_data.clg_data.college_name);
      return 0;
  }
```

OUTPUT:

```
Id is: 1
Name is: Raju
Percentage is: 90.500000
College Id is: 71145
College Name is: GITAM University
```

Sample Program 2 Nested structure

```
#include<stdio.h>
struct Address
         {
           char HouseNo[25];
           char City[25];
           char PinCode[25];

         };
struct Employee
       {
          int Id;
          char Name[25];
          float Salary;
          struct Address Add;
       };
void main()
{
   int i;
   struct Employee E;
   printf("\n\tEnter Employee Id : ");
   scanf("%d",&E.Id);
   printf("\n\tEnter Employee Name : ");
   scanf("%s",&E.Name);
   printf("\n\tEnter Employee Salary : ");
   scanf("%f",&E.Salary);
   printf("\n\tEnter Employee House No : ");
   scanf("%s",&E.Add.HouseNo);
   printf("\n\tEnter Employee City : ");
   scanf("%s",&E.Add.City);
   printf("\n\tEnter Employee House No : ");
   scanf("%s",&E.Add.PinCode);
   printf("\nDetails of Employees");
   printf("\n\tEmployee Id : %d",E.Id);
   printf("\n\tEmployee Name : %s",E.Name);
   printf("\n\tEmployee Salary : %f",E.Salary);
   printf("\n\tEmployeeHouseNo : %s",E.Add.HouseNo);
   printf("\n\tEmployee City : %s",E.Add.City);
   printf("\n\tEmployeeHouseNo : %s",E.Add.PinCode);
 }
```

Output :

```
Enter Employee Id : 101
Enter Employee Name : Suresh
Enter Employee Salary : 45000
Enter Employee House No : 4598/D
```

```
Enter Employee City : Delhi
Enter Employee Pin Code : 110056

Details of Employees
   Employee Id : 101
   Employee Name : Suresh
   Employee Salary : 45000
   Employee House No : 4598/D
    Employee City : Delhi
   Employee Pin Code : 110056
```

Structures Containing Pointers:

- ✓ A **pointer could be a member of structure**, but you should be careful before creating the pointer as a member of structure in C.
- ✓ Generally, we take a pointer as a member when we don't know the length of the data which need to store.

Example:

```
struct Employee
     {
        int Id;
        char *Name;
        float Salary;
        struct Address Add;
     };
```

- ✓ The **structure pointer** points to the address of a memory block where the Structure is being stored.
- ✓ Like a pointer that tells the address of another variable of any data type (int, char, float) in memory. And here, we use a structure pointer which tells the address of a structure in memory by pointing pointer variable **ptr** to the structure variable.

Syntax:

struct structure_name *ptr = &structure_variable;

Access Structure member using pointer:

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

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

Sample Program (Accessing using * operator):

```
#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 = &sub; /* 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;
}
```

Sample Program (Accessing using -> operator):

```
#include <stdio.h>

// create Employee structure
struct Employee
{
    // define the member of the structure
    char name[30];
    int id;
    int age;
    char gender[30];
    char city[40];
};

// define the variables of the Structure with pointers
struct Employee emp1, emp2, *ptr1, *ptr2;

int main()
{
    // store the address of the emp1 and emp2 structure variable
    ptr1 = &emp1;
    ptr2 = &emp2;

    printf (" Enter the name of the Employee (emp1): ");
    scanf (" %s", &ptr1->name);

    printf (" Enter the id of the Employee (emp1): ");
    scanf (" %d", &ptr1->id);
    printf (" Enter the age of the Employee (emp1): ");
```

```
    scanf (" %d", &ptr1->age);
    printf (" Enter the gender of the Employee (emp1): ");
    scanf (" %s", &ptr1->gender);
    printf (" Enter the city of the Employee (emp1): ");
    scanf (" %s", &ptr1->city);

    printf (" \n Second Employee: \n");
    printf (" Enter the name of the Employee (emp2): ");
    scanf (" %s", &ptr2->name);

    printf (" Enter the id of the Employee (emp2): ");
    scanf (" %d", &ptr2->id);
    printf (" Enter the age of the Employee (emp2): ");
    scanf (" %d", &ptr2->age);
    printf (" Enter the gender of the Employee (emp2): ");
    scanf (" %s", &ptr2->gender);
    printf (" Enter the city of the Employee (emp2): ");
    scanf (" %s", &ptr2->city);

    printf ("\n Display the Details of the Employee using Structure Pointer");
    printf ("\n Details of the Employee (emp1) \n");
    printf(" Name: %s\n", ptr1->name);
    printf(" Id: %d\n", ptr1->id);
    printf(" Age: %d\n", ptr1->age);
    printf(" Gender: %s\n", ptr1->gender);
    printf(" City: %s\n", ptr1->city);

    printf ("\n Details of the Employee (emp2) \n");
    printf(" Name: %s\n", ptr2->name);
    printf(" Id: %d\n", ptr2->id);
    printf(" Age: %d\n", ptr2->age);
    printf(" Gender: %s\n", ptr2->gender);
    printf(" City: %s\n", ptr2->city);
    return 0;
}
```