# Structure in C

**Structure in c language** is a user defined datatype that allows you to hold different type of elements.

Each element of a structure is called a member.

It works like a template in C++ and class in Java. You can have different type of elements in it.

It is widely used to store student information, employee information, product information, book information etc.

## **Defining structure**

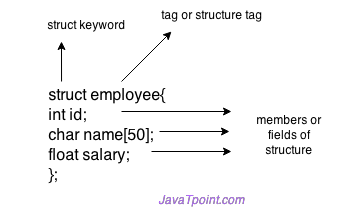
The **struct** keyword is used to define structure. Let's see the syntax to define structure in c.

1. **struct** structure\_name
2. {
3. data\_type member1;
4. data\_type member2;
5. .
6. .
7. data\_type memeberN;
8. };

Let's see the example to define structure for employee in c.

1. **struct** employee
2. {   **int** id;
3. **char** name[50];
4. **float** salary;
5. };

Here, **struct** is the keyword, **employee** is the tag name of structure; **id**, **name** and **salary** are the members or fields of the structure. Let's understand it by the diagram given below:



## **Declaring structure variable**

We can declare variable for the structure, so that we can access the member of structure easily. There are two ways to declare structure variable:

1. By struct keyword within main() function
2. By declaring variable at the time of defining structure.

**1st way:**

Let's see the example to declare structure variable by struct keyword. It should be declared within the main function.

1. **struct** employee
2. {   **int** id;
3. **char** name[50];
4. **float** salary;
5. };

Now write given code inside the main() function.

1. **struct** employee e1, e2;

**2nd way:**

Let's see another way to declare variable at the time of defining structure.

1. **struct** employee
2. {   **int** id;
3. **char** name[50];
4. **float** salary;
5. }e1,e2;

#### Which approach is good

But if no. of variable are not fixed, use 1st approach. It provides you flexibility to declare the structure variable many times.

If no. of variables are fixed, use 2nd approach. It saves your code to declare variable in main() fuction.

## **Accessing members of structure**

There are two ways to access structure members:

1. By . (member or dot operator)
2. By -> (structure pointer operator)

Let's see the code to access the id member of p1 variable by . (member) operator.

1. p1.id

#### C Structure example

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

1. #include <stdio.h>
2. #include <string.h>
3. **struct** employee
4. {   **int** id;
5. **char** name[50];
6. }e1;  //declaring e1 variable for structure
7. **int** main( )
8. {
9. //store first employee information
10. e1.id=101;
11. strcpy(e1.name, "Sonoo Mishra");//copying string into char array
12. //printing first employee information
13. printf( "employee 1 id : %d\n", e1.id);
14. printf( "employee 1 name : %s\n", e1.name);
15. **return** 0;
16. }

Output:

employee 1 id : 101

employee 1 name : Sonoo Mishra

Let's see another example of structure in C language to store many employees information.

1. #include <stdio.h>
2. #include <string.h>
3. **struct** employee
4. {   **int** id;
5. **char** name[50];
6. **float** salary;
7. }e1,e2;  //declaring e1 and e2 variables for structure
8. **int** main( )
9. {
10. //store first employee information
11. e1.id=101;
12. strcpy(e1.name, "Sonoo Mishra");//copying string into char array
13. e1.salary=56000;
15. //store second employee information
16. e2.id=102;
17. strcpy(e2.name, "James Bond");
18. e2.salary=126000;
20. //printing first employee information
21. printf( "employee 1 id : %d\n", e1.id);
22. printf( "employee 1 name : %s\n", e1.name);
23. printf( "employee 1 salary : %f\n", e1.salary);
25. //printing second employee information
26. printf( "employee 2 id : %d\n", e2.id);
27. printf( "employee 2 name : %s\n", e2.name);
28. printf( "employee 2 salary : %f\n", e2.salary);
30. **return** 0;
31. }

Output:

employee 1 id : 101

employee 1 name : Sonoo Mishra

employee 1 salary : 56000.000000

employee 2 id : 102

employee 2 name : James Bond

employee 2 salary : 126000.000000

# 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:

1. By separate structure
2. 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.

1. **struct** Date
2. {
3. **int** dd;
4. **int** mm;
5. **int** yyyy;
6. };
7. **struct** Employee
8. {
9. **int** id;
10. **char** name[20];
11. **struct** Date doj;
12. }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.

1. **struct** Employee
2. {
3. **int** id;
4. **char** name[20];
5. **struct** Date
6. {
7. **int** dd;
8. **int** mm;
9. **int** yyyy;
10. }doj;
11. }emp1;

#### Accessing Nested Structure

We can access the member of nested structure by Outer\_Structure.Nested\_Structure.member as given below:

1. e1.doj.dd
2. e1.doj.mm
3. e1.doj.yyyy

#### C Nested Structure example

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

1. #include <stdio.h>
2. #include <string.h>
3. **struct** Employee
4. {
5. **int** id;
6. **char** name[20];
7. **struct** Date
8. {
9. **int** dd;
10. **int** mm;
11. **int** yyyy;
12. }doj;
13. }e1;
14. **int** main( )
15. {
16. //storing employee information
17. e1.id=101;
18. strcpy(e1.name, "Sonoo Mishra");//copying string into char array
19. e1.doj.dd=10;
20. e1.doj.mm=11;
21. e1.doj.yyyy=2014;
23. //printing first employee information
24. printf( "employee id : %d\n", e1.id);
25. printf( "employee name : %s\n", e1.name);
26. printf( "employee date of joining (dd/mm/yyyy) : %d/%d/%d\n", e1.doj.dd,e1.doj.mm,e1.doj.yyyy);
27. **return** 0;
28. }

Output:

employee id : 101

employee name : Sonoo Mishra

employee date of joining (dd/mm/yyyy) : 10/11/2014