## How to create a structure in C Programming

We use **struct** keyword to create a **structure in C**. The struct keyword is a short form of **structured data type.**

struct struct\_name {

DataType member1\_name;

DataType member2\_name;

DataType member3\_name;

…

};

Here struct\_name can be anything of your choice. Members data type can be same or different. Once we have declared the structure we can use the struct name as a data type like int, float etc.

First we will see the syntax of creating struct variable, accessing struct members etc and then we will see a complete example.

### How to declare variable of a structure?

struct  struct\_name  var\_name;

or

struct struct\_name {

DataType member1\_name;

DataType member2\_name;

DataType member3\_name;

…

} var\_name;

### How to access data members of a structure using a struct variable?

var\_name.member1\_name;

var\_name.member2\_name;

…

### How to assign values to structure members?

There are three ways to do this.  
1) Using Dot(.) operator

var\_name.memeber\_name = value;

2) All members assigned in one statement

struct struct\_name var\_name =

{value for memeber1, value for memeber2 …so on for all the members}

3) **Designated initializers** – We will discuss this later at the end of this post.

### Example of Structure in C

#include <stdio.h>

/\* Created a structure here. The name of the structure is

\* StudentData.

\*/

struct StudentData{

char \*stu\_name;

int stu\_id;

int stu\_age;

};

int main()

{

/\* student is the variable of structure StudentData\*/

struct StudentData student;

/\*Assigning the values of each struct member here\*/

student.stu\_name = "Steve";

student.stu\_id = 1234;

student.stu\_age = 30;

/\* Displaying the values of struct members \*/

printf("Student Name is: %s", student.stu\_name);

printf("\nStudent Id is: %d", student.stu\_id);

printf("\nStudent Age is: %d", student.stu\_age);

return 0;

}

Output:

Student Name is: Steve

Student Id is: 1234

Student Age is: 30

## Nested Structure in C: Struct inside another struct

You can use a structure inside another structure, which is fairly possible. As I explained above that once you declared a structure, the **struct struct\_name** acts as a new data type so you can include it in another struct just like the data type of other data members. Sounds confusing? Don’t worry. The following example will clear your doubt.

### Example of Nested Structure in C Programming

Lets say we have two structure like this:  
**Structure 1: stu\_address**

struct stu\_address

{

int street;

char \*state;

char \*city;

char \*country;

}

**Structure 2: stu\_data**

struct stu\_data

{

int stu\_id;

int stu\_age;

char \*stu\_name;

struct stu\_address stuAddress;

}

As you can see here that I have nested a structure inside another structure.

#### Assignment for struct inside struct (Nested struct)

Lets take the example of the two structure that we seen above to understand the logic

struct  stu\_data  mydata;

mydata.stu\_id = 1001;

mydata.stu\_age = 30;

mydata.stuAddress.state = "UP"; //Nested struct assignment

..

#### How to access nested structure members?

Using chain of “.” operator.  
Suppose you want to display the city alone from nested struct –

printf("%s",  mydata.stuAddress.city);

## Use of typedef in Structure

typedef makes the code short and improves readability. In the above discussion we have seen that while using structs every time we have to use the lengthy syntax, which makes the code confusing, lengthy, complex and less readable. The simple solution to this issue is use of typedef. It is like an alias of struct.

**Code without typedef**

struct home\_address {

int local\_street;

char \*town;

char \*my\_city;

char \*my\_country;

};

...

struct home\_address var;

var.town = "Agra";

**Code using tyepdef**

typedef struct home\_address{

int local\_street;

char \*town;

char \*my\_city;

char \*my\_country;

}addr;

..

..

addr var1;

var.town = "Agra";

Instead of using the struct home\_address every time you need to declare struct variable, you can simply use addr, the typedef that we have defined.

## Designated initializers to set values of Structure members

We have already learned two ways to set the values of a struct member, there is another way to do the same using designated initializers. This is useful when we are doing assignment of only few members of the structure. In the following example the structure variable s2 has only one member assignment.

#include <stdio.h>

struct numbers

{

int num1, num2;

};

int main()

{

// Assignment using using designated initialization

struct numbers s1 = {.num2 = 22, .num1 = 11};

struct numbers s2 = {.num2 = 30};

printf ("num1: %d, num2: %d\n", s1.num1, s1.num2);

printf ("num1: %d", s2.num2);

return 0;

}

Output:

num1: 11, num2: 22

num1: 30