Structure

Arrays allow to define type of variables that can hold several data items of the same kind. Similarly, structure is another user defined data type available in C that allows to combine data items of different kinds.

Structures are used to represent a record. Suppose you want to keep track of your books in a library. You might want to track the following attributes about each book −

Title

Author

Subject

Book ID

Defining a Structure

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;
   …
};
```

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;
```

Here is the way you would declare the Book structure −

```
struct Books {
   char  title[50];
   char  author[50];
   char  subject[100];
   int   book_id;
} book1,book2,book3;

Or,

struct Books book1,book2,book3;
```

How to assign values to structure members?

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

var_name.memeber_name = value; // book1.book_id = 125;

2) All members assigned in one statement

struct struct_name var_name =

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

Example of Structure in C

```c
#include <stdio.h>
/* Created a structure here. The name of the structure is
 * StudentData.
 */
struct StudentData{
    char stu_name[50];
    int stu_id;
    int stu_age;
};
int main()
{
     /* student is the variable of structure StudentData*/
     struct StudentData student1, student2;

     /*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", student1.stu_name);
     printf("\nStudent Id is: %d", student1.stu_id);
```

```
    printf("\nStudent Age is: %d", student1.stu_age);
    return 0;
}
```

Output:

```
Student Name is: Steve
Student Id is: 1234
Student Age is: 30
```

Example: Book Example

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

struct Books {
  char  title[50];
  char  author[50];
  char  subject[100];
  int   book_id;
};

int main( ) {

  struct Books Book1;        /* Declare Book1 of type Book */
  struct Books Book2;        /* Declare Book2 of type Book */

  /* book 1 specification */
 Book1.title[] = "C Programming";
Book1.author [] = "Nuha Ali";
Book1.subject [] = "C Programming Tutorial"
Book1.book_id = 6495407;

  /* book 2 specification */
  strcpy( Book2.title, "Telecom Billing");
  strcpy( Book2.author, "Zara Ali");
  strcpy( Book2.subject, "Telecom Billing Tutorial");
  Book2.book_id = 6495700;

  /* print Book1 info */
  printf( "Book 1 title : %s\n", Book1.title);
  printf( "Book 1 author : %s\n", Book1.author);
  printf( "Book 1 subject : %s\n", Book1.subject);
  printf( "Book 1 book_id : %d\n", Book1.book_id);
```

```
   /* print Book2 info */
   printf( "Book 2 title : %s\n", Book2.title);
   printf( "Book 2 author : %s\n", Book2.author);
   printf( "Book 2 subject : %s\n", Book2.subject);
   printf( "Book 2 book_id : %d\n", Book2.book_id);

   return 0;
}
```

When the above code is compiled and executed, it produces the following result −

Book 1 title : C Programming
Book 1 author : Nuha Ali
Book 1 subject : C Programming Tutorial
Book 1 book_id : 6495407
Book 2 title : Telecom Billing
Book 2 author : Zara Ali
Book 2 subject : Telecom Billing Tutorial
Book 2 book_id : 6495700

0	1	2	3	5
char name[12]; char ID[32]; int quiz1; int quiz2; int quiz3;	char name[12]; char ID[32]; int quiz1; int quiz2; int quiz3;	char name[12]; char ID[32]; int quiz1; int quiz2; int quiz3;	char name[12]; char ID[32]; int quiz1; int quiz2; int quiz3;	char name[12]; char ID[32]; int quiz1; int quiz2; int quiz3;