Content 26

Structures In C

  Variables and arrays can handle a great variety of situations. But quite often we have to deal with the collection of dissimilar data types. For dealing with dissimilar data types, C provides a data type called **‘structure’**. A structure gathers together, different information that belongs to different data types.

#### What is a Structure in C?

***Structures***are usually used when we want to store dissimilar data together.For example, we want to store data about a book. Book has its title, author name, number of pages and price. All of the book attributes belong to different data types. One way to store the data is to construct individual arrays, and another method is to use a structure variable. Structure elements are always stored in contiguous memory locations.

**The general form of a structure declaration statement is given below:**

struct <structure name> {

structure element 1;

structure element 2;

structure element 3;

} struct\_variable;

//Or

struct book b1, b2, b3 ;

Before the final semicolon, we specify the structure variables but it is optional. We can also specify the structure variable in main body. The keyword **struct** should be used to define variables of structure type.

**Following is the example of declaring struct in C:**

struct Books{

char title[20];

char author\_name[100];

float price;

int pages;} book1;

#### How to Access the Structure Elements?

As we use subscript to access individual elements of an array. But in the case of structures, to access any element, we use the**operator (.)**. This dot operator is coded between the structure variable name and the structure member that we wish to access.

**Note** that before the dot operator there must always be a structure variable and after the dot operator there must always be a structure element.

#include <stdio.h>

#include <string.h>

struct book

{

    char title[15];

    char author\_of\_book[10];

    float price;

    int pages\_of\_book;

} b1, b2;

int main()

{ printf("Displaying b1 book details;\n");

    struct book b1 = {"C Programming", "Charlie", 350.58, 180};

    printf("Title: %s\n", b1.title);

    printf("Title: %s\n", b1.author\_of\_book);

    printf("Title: %.2f\n", b1.price);

    printf("Title: %d\n\n", b1.pages\_of\_book);

    printf("Displaying b2 book details;\n");

    struct book b2 = {"C Programming2", "James", 150.78, 192};

    printf("Title: %s\n", b2.title);

    printf("Title: %s\n", b2.author\_of\_book);

    printf("Title: %.2f\n", b2.price);

    printf("Title: %d\n", b2.pages\_of\_book);

    return 0;

}

**Output:**

Displaying b1 book details;

Title: C Programming

Title: Charlie

Title: 350.58

Title: 180

Displaying b2 book details;

Title: C Programming2

Title: James

Title: 150.78

Title: 192

**Code1:**

#include <stdio.h>

#include <string.h>

struct distance

{

    int feet;

    float inches;

} d1, d2, sum;

int main()

{

    printf("Enter 1st Distance\n");

    printf("Enter feet Distance d1:  ");

    scanf("%d", &d1.feet);

    printf("Enter inches Distance d1:  ");

    scanf("%f", &d1.inches);

    printf("\nEnter 2nd Distance \n");

    printf("Enter feet Distance d2:  ");

    scanf("%d", &d2.feet);

    printf("Enter feet Distance d2:  ");

    scanf("%f", &d2.inches);

    //adding feet

    sum.feet = d1.feet + d2.feet;

    printf("\nSum of feet is: %d\n", sum.feet);

    //adding inches

    sum.inches = d1.inches + d2.inches;

    printf("Sun of inches is: %.2f", sum.inches);

    //converrting to feet if inch is greater than 12

    while (sum.inches >= 12)

    {

        sum.inches -= 12;   //sum.inches = sum.inches - 12

        sum.feet++;

    }

    printf("\n\nSum of distances = %d\'-%.1f\"", sum.feet, sum.inches);

    return 0;

}

**Output:**

Enter 1st Distance

Enter feet Distance d1: 12

Enter inches Distance d1: 14

Enter 2nd Distance

Enter feet Distance d2: 4

Enter feet Distance d2: 47

Sum of feet is: 16

Sun of inches is: 61.00

Sum of distances = 21'-1.0"