C Programming

Structures

There are times when you will design a program in C that uses different types of data. In such cases, it is not possible to use an array because arrays only store homogeneous data, i.e., data of the same type.

In such situations, C allows us to use a different data structure to store different types of data. This is called a **Structure**.

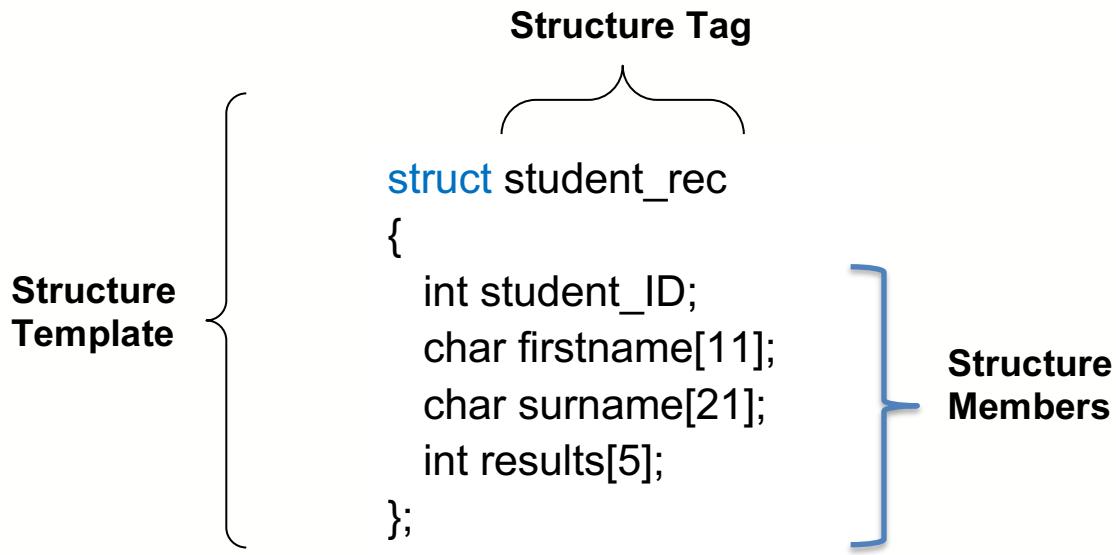
Let's assume we need to design a data structure to store a simple student record. The record consists of a (i) Student ID, (ii) Firstname, (iii) Surname, (iv) Set of results

We can use a Structure to capture the above data. Firstly, let's look at how a Structure is formed in C.

Defining a Structure

The first step when defining a Structure is to declare a *Structure Template*:

```
struct student_rec
{
      int student_ID;
      char firstname[11];
      char surname[21];
      int results[5];
};
```




```c
/*
Introduction to Structures. Creating your first Structure template
*/
#include <stdio.h>

#define SIZE 11

//Structure template(s)
struct student_rec
{
   int student_ID;
   char firstname[11];
   char surname[21];
   int results[5];
};

//Structure template(s)
struct staff_rec
{
   int staff_ID;
   char firstname[11];
```

```
    char surname[21];
};

//Function signature(s)
// ...
// ...

int main()
{
    ...
    ...

    return 0;

} // end main()
```

Declaring a Structure template does not create a variable of the Structure. Therefore, no memory has yet been used in RAM.

We must now create a variable of this Structure in order to use it in the code. Creating a Structure variable is as follows:

```
int main()
{
    // create a student_rec variable(s)
    struct student_rec student1, student2;

    ...
    ...

    return 0;

} // end main()
```

The structure variables **student1** and **student2** both have a student ID, firstname, surname and a set of results.

To assign data to any structure variable member, you must use the full-stop/period operator as follows:

structure_variable_name ■ structure_member = abcd;

e.g.,

```
int main()
{
   // create a student_rec variable(s)
   struct student_rec student1, student2;

   // Assign a numeric student ID to student1
   student1.student_ID = 1234;

   return 0;

} // end main()
```

Now let's assign data to all the structure members in student1

```
/*
Introduction to Structures. Creating your first Structure
template
*/
#include <stdio.h>
#include <string.h>

#define SIZE 11



//Structure template(s)
struct student_rec
```

```
{
   int student_ID;
   char firstname[11];
   char surname[21];
   int results[5];
};

//Function signature(s)
// ...
// ...
int main()
{
   // create a student_rec variable(s)
   struct student_rec student1, student2;

   // Assign a numeric student ID to student1
   student1.student_ID = 1234;

   //Assign a string as the first name to student1
   //Unfortunately, you cannot assign a string to a character
   //array directly as follows
   //student1.firstname = "Sarah";

   //You should use the strcpy() function to assign a string to a
   //char array
   strcpy(student1.firstname,"Sarah");

   //Assign a surname to student1
   strcpy(student1.surname,"Jones");

   // Assign a set of results to student1
   // insert code here ..

   printf("\nThe structure member contains\n");
   printf("\nStudent ID is: %d", student1.student_ID);
   printf("\nFirst name is: %s", student1.firstname);
```

```
    printf("\nSurname is: %s", student1.surname);

    for(i = 0; i < 5; i++)
    {
        printf("%d\n", student1.results[i]);

    } // end for

    return 0;

} // end main()
```

Repl 20.1: https://replit.com/@michaelTUDublin/201-Basic-structure

The **Full Code** for the program to assign data to student1 and have data entered for student2 is:

```
/*
Introduction to Structures. Creating your first Structure
template
*/
#include <stdio.h>
#include <string.h>

#define SIZE 5

//Structure template(s)
struct student_rec
{
  int student_ID;
  char firstname[11];
  char surname[21];
  int results[5];
};
```

```c
//Function signature(s)
// ...
// …

int main()
{
   struct student_rec student1, student2;
   int i;


   // Assign a numeric student ID to student1
   student1.student_ID = 1234;

   // Assign a first name to student1
   // Unfortunately, you cannot assign a string to a character
   // array directly as follows
   //student1.firstname = "Sarah";

   //You should use the strcpy() function to assign a string to a
   //char array
   strcpy(student1.firstname,"Sarah");

   // Assign a surname to student 1
   strcpy(student1.surname,"Jones");
   // Assign a set of results to student1
   student1.results[0] = 100;
   student1.results[1] = 89;
   student1.results[2] = 56;
   student1.results[3] = 95;
   student1.results[4] = 91;

   printf("\nStudent 1");
   printf("\nThe structure member contains\n");
   printf("\nStudent 1 student ID is: %d", student1.student_ID);
   printf("\nStudent 1 first name is: %s", student1.firstname);
```

```
    printf("\nStudent 1 surname is: %s", student1.surname);
    printf("\nStudent 1 results are:\n");

    // display the set of 5 results for student1
    for(i = 0; i < SIZE; i++)
    {
           printf("%d\n", student1.results[i]);
    } // end for

    // Enter the data for student2 from standard input (keyboard)
    printf("\nStudent 2");
    printf("\nEnter the student ID for Student 2: ");
    scanf("%d", & student2.student_ID);

    printf("\nEnter the first name for Student 2: ");
    scanf("%s", student2.firstname);

    printf("\nEnter the surname for Student 2: ");
    scanf("%s", student2.surname);

    printf("\nEnter %d results for Student 2\n", SIZE);

    // Enter the 5 results into the results structure member for
Student 2
    for(i = 0; i < SIZE; i++)
    {
        scanf("%d", & student2.results[i]);

    } // end for

    printf("\nStudent 2");
    printf("\nThe structure member contains\n");
    printf("\nStudent 2 student ID is: %d", student2.student_ID);
    printf("\nStudent 2 first name is: %s", student2.firstname);
    printf("\nStudent 2 surname is: %s", student2.surname);
    printf("\nStudent 2 results are:\n");
```

```
    // display the set of 5 results for student1
    for(i = 0; i < 5; i++)
    {
            printf("%d\n", student2.results[i]);
    } // end for

    return 0;

} // end main()
```

Repl 20.2: https://replit.com/@michaelTUDublin/202-Structure-variables