**MIT-WORLD PEACE UNIVERSITY**

**F. Y. B. Tech**

## Trimester: I/II/III  Subject: Programming and Problem Solving

**Name**: Krishnaraj Thadesar  **Division:** 9

**Roll No.:** 109054  **Batch:** I3

**Experiment No.:** 7

**Name of the Experiment:** Write a C program to display student details with Structures.

**Performed on:** 17rd February 2022

**Submitted on:** 24st January 2022

---

**AIM**: Write a C program to accept student details and display their result using structures.

**OBJECTIVE:**

1. To study the concept of Structure in C language

**THEORY:**

When we require using a collection of different data items of different data types we can use a structure. Structure is a method of packing data of different types. A structure is a convenient method of handling a group of related data items of different data types.

*Structure definition:*
General format:

```
structtag_name
{
data type member1;
data type member2;
...
...
};
```

Example:

```
structlib_books
{
char title[20];
char author[15];
int pages;
float price;
};
```

The keyword struct declares a structure to holds the details of four fields namely title, author pages and price. These are members of the structures. Each member may belong to different or same data type. The tag name can be used to define objects that have the tag names structure. The structure we just declared is not a variable by itself but a template for the structure.

We can declare structure variables using the tag name anywhere in the program. For example the statement,

```
structlib_books book1,book2,book3;
```

Declares book1, book2,book3 as variables of type structlib_books each declaration has four elements of the structure lib_books. The complete structure declaration might look like this

```
structlib_books
{
char title[20];
char author[15];
int pages;
float price;
};

structlib_books, book1, book2, book3;
```

Structures do not occupy any memory until it is associated with the structure variable such as book1. The template is terminated with a semicolon. While the entire declaration is considered as a statement, each member is declared independently for its name and type in a separate statement inside the template. The tag name such as lib_books can be used to declare structure variables of its data type later in the program.

We can also combine both template declaration and variables declaration in one statement, the declaration

```
structlib_books
```

```
{
char title[20];
char author[15];
int pages;
float price;
} book1,book2,book3;
```
is valid. The use of tag name is optional.

A structure is usually defines before main along with macro definitions. In such cases the structure assumes global status and all the functions can access the structure.

**Giving values to members:**

As mentioned earlier the members themselves are not variables they should be linked to structure variables in order to make them meaningful members. The link between a member and a variable is established using the member operator '.' Which is known as dot operator or period operator.

For example:

*Book1.price*

**PLATFORM:** *64 Bit Arch Linux with g++/gcc compiler.*

ALGORITHM:

Accept Student_Information

**Algorithm Stud_Info( )**

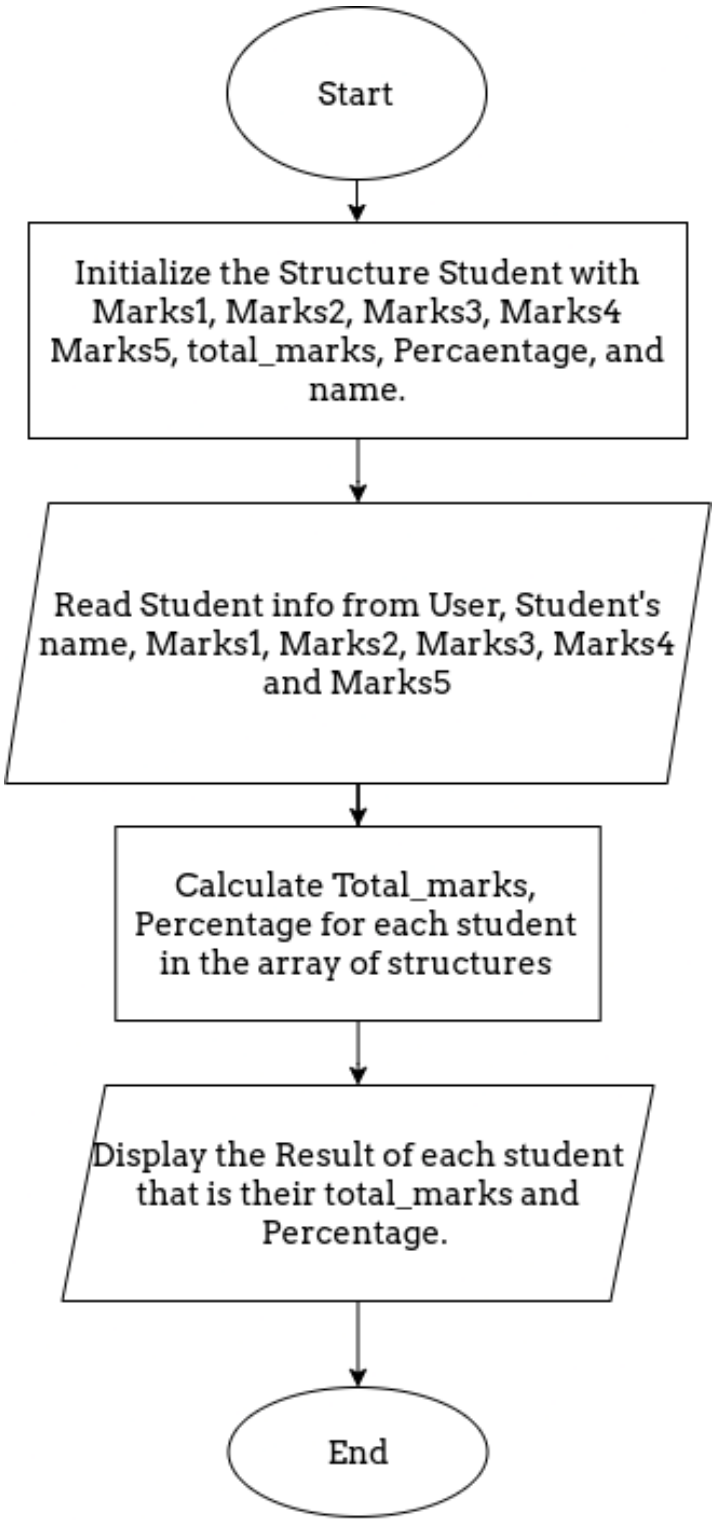
a. Read Student Roll Number,
b. Read Name,
c. Read Branch
d. Read Marks1, Marks2, Marks3, Marks4 and Marks5
e. Calculate Total Marks, Percentage and Class and store it into their respective variables
f. End

Display Student_Information

**Algorithm Display_Stud_Info ( )**

g. Display Student Roll Number,
h. Display Name,
i. Display Branch
j. Display Marks1, Marks2, Marks3, Marks4 and Marks5
k. Display Total Marks, Percentage and Class
l. End

Flowchart:


Start
Initialize the Structure Student with Marks1, Marks2, Marks3, Marks4 Marks5, total_marks, Percaentage, and name.
Read Student info from User, Student's name, Marks1, Marks2, Marks3, Marks4 and Marks5
Calculate Total_marks, Percentage for each student in the array of structures
Display the Result of each student that is their total_marks and Percentage.
End

CODE:

```c
#include <stdio.h>

struct Student
{
    char name[100];
    int english;
    int maths;
    int science;
    int total, average, percentage;
};

int main()
{
    int number_of_students = 0;
    struct Student students[100];
    printf("Enter the number of students you want to enter the marks for (Less than 100): ");
    scanf("%d", &number_of_students);
    for (int i = 0; i < number_of_students; i++)
    {
        printf("\n Enter the name of the Student: ");
        scanf("%s", &students[i].name);
        printf("Enter the marks obtained out of 100 in Maths: ");
        scanf("%d", &students[i].maths);
        printf("Enter the marks obtained out of 100 in Science: ");
        scanf("%d", &students[i].science);
        printf("Enter the marks obtained out of 100 in English: ");
        scanf("%d", &students[i].english);
        students[i].total = students[i].english + students[i].maths + students[i].science;
        students[i].percentage = students[i].total / 3;
        printf("The Total Marks obtained by the student %s are: %d", students[i].name, students[i].total);
        printf("\nThe Percentage obtained by the student %s is: %d", students[i].name, students[i].percentage);
    }

    return 0;
}
```

## OUTPUT

```
Enter the number of students you want to enter the marks for (Less than 100): 2

 Enter the name of the Student: Tony
Enter the marks obtained out of 100 in Maths: 99
Enter the marks obtained out of 100 in Science: 99
Enter the marks obtained out of 100 in English: 99
The Total Marks obtained by the student Tony are: 297
The Percentage obtained by the student Tony is: 99
 Enter the name of the Student: Peter
Enter the marks obtained out of 100 in Maths: 98
Enter the marks obtained out of 100 in Science: 90
Enter the marks obtained out of 100 in English: 78
The Total Marks obtained by the student Peter are: 266
The Percentage obtained by the student Peter is: 88
```

## CONCLUSION:

The concept of Structure in C language was understood.

## FAQs:

*Q1. What is a structure?*

When we require using a collection of different data items of different data types we can use a structure. Structure is a method of packing data of different types. A structure is a convenient method of handling a group of related data items of different data types.

*Q2. How a member of structure is accessed?*

The link between a member and a variable is established using the member operator '.' Which is known as dot operator or period operator. This is how a member of a structure can be accessed, or changed.

For example:
*Book1.price*

*Q3. Explain similarities between Structure and Union?*

*Both are user-defined data types used to store data of different types as a single unit.*

*1.Their members can be objects of any type, including other structures and unions or arrays. A member can also consist of a bit field.*

*2.Both structures and unions support only assignment = and sizeof operators. The two structures or unions in the assignment must have the same members and member types.*

*3.A structure or a union can be passed by value to functions and returned by value by functions. The argument must have the same type as the function parameter. A structure or union is passed by value just like a scalar variable as a corresponding parameter.*

4.*'.' operator is used for accessing members.*