LAB8_Structures

OBJECT ORIENTED PROGRAMMING

BSCS-SPRING-2022

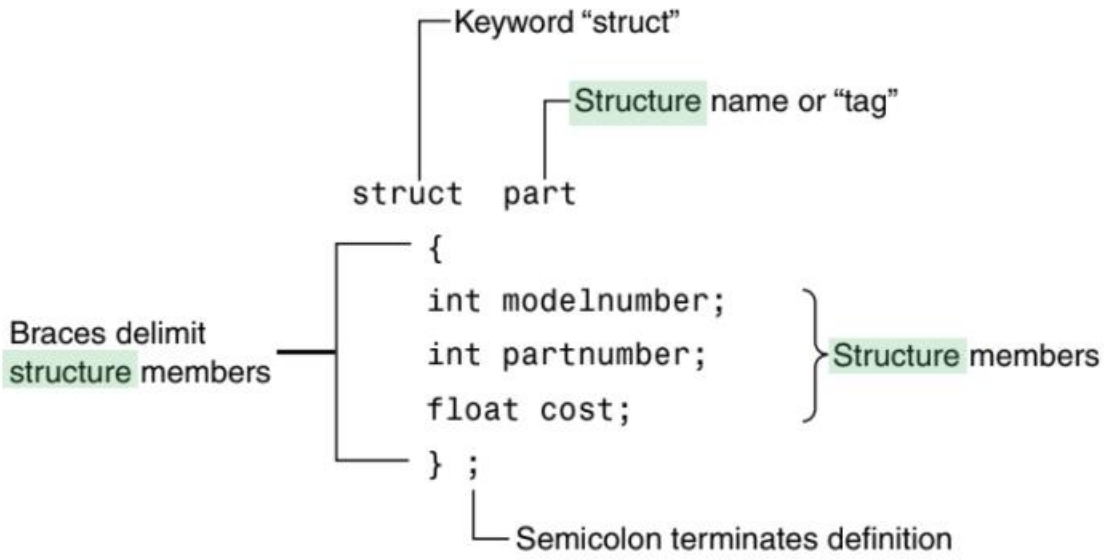
Structures

Structure: C++ arrays allow you to define variables that combine several data items of the same kind, but structure is another user defined data type, which allows you to combine data items of different kinds.

A structure is a collection of simple variables. The variables in a structure can be of different types: Some can be int, some can be float, and so on.

Syntax of the Structure Definition:

The keyword struct introduces the structure definition. Next comes the structure name or tag, which is part. The declarations of the structure members—modelnumber, partnumber, and cost—are enclosed in braces. A semicolon follows the closing brace, terminating the entire structure. Note that this use of the semicolon for structures is unlike the usage for a block of code. As we have seen, blocks of code, which are used in loops, braces, also delimit decisions, and functions. However, they do not use a semicolon following the final brace.



Types of Members Structures

In C++ can contain two types of members:

• **Data Member:** These members are normal C++ variables. We can create a structure with variables of different data types in C++.

• **Member Functions:** These members are normal C++ functions. Along with variables, we can also include functions inside a structure declaration.

Access structure member:

To access any member of a structure, we use the member access operator (.). The member access operator is coded as a period between the structure variable name and the structure member that we wish to access. You would use struct keyword to define variables of structure type. Following is the example to explain usage of structure.

```
#include <iostream>

using namespace std;

struct part //specify a structure

{

int modelnumber; //ID number of widget

int partnumber; //ID number of widget part

float cost; //cost of part

};

int main()

{ //initialize variable

part part1 = { 6244, 373, 217.55F };

part part2; //define variable

//display first variable

cout << "Model " << part1.modelnumber;

cout << ", part " << part1.partnumber;

cout << ", costs $" << part1.cost << endl;
```

```
part2 = part1; //assign first variable to second
//display second variable

cout << "Model " << part2.modelnumber;
cout << ", part " << part2.partnumber;
cout << ", costs $" << part2.cost << endl;
return 0;
}
```

Output:

```
Model 6244, part 373, costs $217.55
Model 6244, part 373, costs $217.55
```

Access modifier:

There are 3 types of access modifiers available in C++:

1. Public
2. Private
3. Protected

For example:

```
struct s1 {
 int a; // public
private:
int b; // private
protected:
int c; // protected
public:
int d; // public again
};
```

Tasks

Problem 1:

Write a structure to store the roll no., name, age (between 11 to 14) and address of students (more than 10). Store the information of the students. 1 - Write a function to print the names of all the students having age 14. 2 - Write another function to print the names of all the students having even roll no. 3 - Write another function to display the details of the student whose roll no is given (i.e. roll no. entered by the user).

Problem 2:

Enter the marks of 5 students in Chemistry, Mathematics and Physics (each out of 100) using a structure named Marks having elements roll no., name, chem_marks, maths_marks and phy_marks and then display the percentage of each student.

Problem 3:

Write a program to take the differences from start and stop time using structures. There should be proper adjustment for seconds, minutes and hours.

```
struct TIME {

int hours;

int minutes;

 int seconds;

struct TIME differenceBetweenTimePeriod(struct TIME stop) {

 }

};
```

Problem 4:

Write a program to add two distances in inch-feet using structure. The function designed for sum should be inside the structure and inch must be added to inch and similar for feet. If inches exceed from 11 and reaches to 12 it will be a feet then there should be increment in feet sum and decrement in inch and leverage inches less than 12.The return type of the function should be struct. The function name should be “add” and the return type must be a struct; the function definition must be inside the structure. (Hint: 1 feet=12 inches)