

## 4] Car Class

Write a class named Car that has the following member variables:

- yearModel. An int that holds the car's year model.
- make. A string that holds the make of the car.
- speed. An int that holds the car's current speed.

In addition, the class should have the following constructor and other member functions.

- Constructor. The constructor should accept the car's year model and make as arguments. These values should be assigned to the object's yearModel and make member variables. The constructor should also assign 0 to the speed member variables.
- Accessors. Appropriate accessor functions to get the values stored in an object's yearModel, make, and speed member variables.
- Accelerate. The accelerate function should add 5 to the speed member variable each time it is called.
- Brake. The brake function should subtract 5 from the speed member variable each time it is called.

Demonstrate the class in a program that creates a Car object, and then calls the accelerate function five times. After each call to the accelerate function, get the current speed of the car and display it. Then, call the brake function five times. After each call to the brake function, get the current speed of the car and display it.

**The source code is in three separate files**

Source Code:

**File 1 (Named Car.h):**

```
#pragma once
//Car.h
//DO NOT MODIFY THIS SECTION
#ifndef CAR_H
#define CAR_H
#include <string>
class Car
{
private:
    int year;
    std::string make;
    int speed;
public:
    Car(int y, std::string m);
    int getYear();
    //ADD YOUR CODE FROM HERE
    //Add prototypes for accessors and accelerate and brake methods
    std::string getMake() const;
    int getSpeed() const;
```

```

        int accelerate();
        int brake();
};
#endif

```

## File 2 (Named Car.cpp):

```

#include "Car.h"
using namespace std;

//Give definitions for constructor, accessors, and accelerator and brake methods
Car::Car(int y, string m)
{
    Car::year = y;
    Car::make = m;
    Car::speed = 0;
}

int Car::getYear()
{
    return Car::year;
}

string Car::getMake() const
{
    return Car::make;
}

int Car::getSpeed() const
{
    return Car::speed;
}

int Car::accelerate()
{
    return Car::speed += 5;
}

int Car::brake()
{
    return Car::speed -= 5;
}

```

## File 3 (Named lab8\_1.cpp):

```

#include <iostream>
#include "Car.h"
using namespace std;

int main()
{
    //1. Create an object of Car class
    Car c1(1965, "Buick");

    //2. Print c1 data
    cout << "Data from this car: " << c1.getYear() << ", " << c1.getMake() <<
endl;
}

```

```

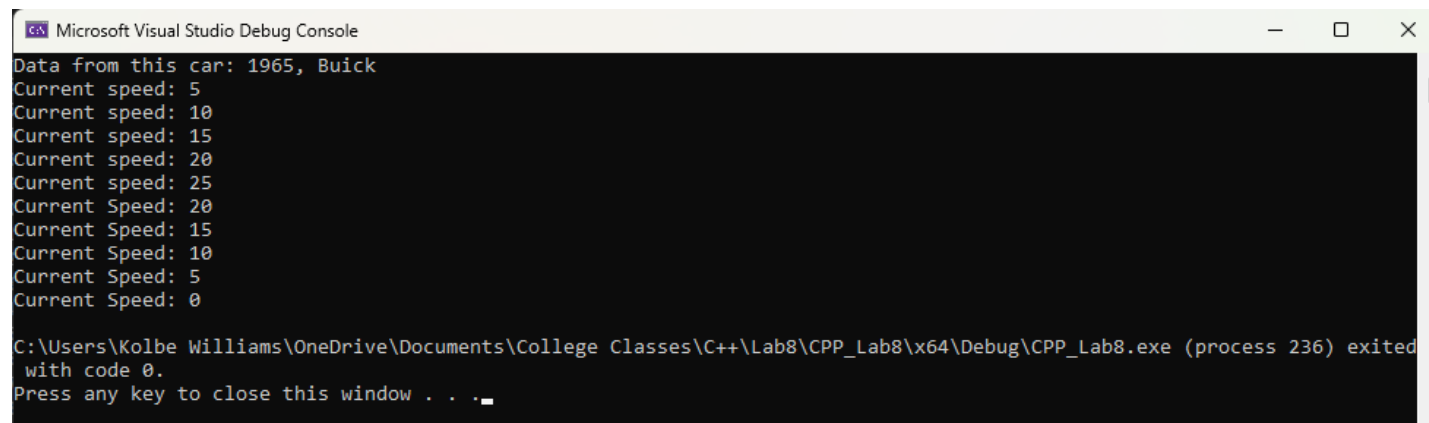
//3. Use a for loop to call accelerate method five times and print the speed
for (int i = 0; i < 5; i++)
{
    c1.accelerate();
    cout << "Current speed: " << c1.getSpeed() << endl;
}

//4. Use a for loop to call brake method five times and print the speed
for (int i = 0; i < 5; i++)
{
    c1.brake();
    cout << "Current Speed: " << c1.getSpeed() << endl;
}

return 0;
}

```

### Output:



The screenshot shows the Microsoft Visual Studio Debug Console window. The title bar reads "Microsoft Visual Studio Debug Console". The output text is as follows:

```

Data from this car: 1965, Buick
Current speed: 5
Current speed: 10
Current speed: 15
Current speed: 20
Current speed: 25
Current Speed: 20
Current Speed: 15
Current Speed: 10
Current Speed: 5
Current Speed: 0

C:\Users\Kolbe Williams\OneDrive\Documents\College Classes\C++\Lab8\CPP_Lab8\x64\Debug\CPP_Lab8.exe (process 236) exited
with code 0.
Press any key to close this window . . .

```

## 5] Coin Toss Simulator

Write a class named Coin. The Coin class should have the following member variable:

- A string named sideUp. The sideUp member variable will hold either “heads” or “tails” indicating the side of the coin that is facing up.

The Coin class should have the following member functions:

- A default constructor that randomly determines the side of the coin that is facing up (“heads” or “tails”) and initializes the sideUp member variable accordingly.
- A void member function named toss that simulates the tossing of the coin. When the toss member function is called, it randomly determines the side of the coin that is facing up (“heads” or “tails”) and sets the sideUp member variable accordingly.
- A member function named getSideUp that returns the value of the sideUp member variable.

Write a program that demonstrates the Coin class. The program should create an instance of the class and display the side that is initially facing up. Then, use a loop to toss the coin 20 times. Each time the coin is tossed, display the side that is facing up. The program should keep count of the number of times heads are facing up and the number of times tails are facing up, and display those values after the loop finishes.

## The Source Code is in Three Separate Files

Source Code:

### File 1 (Named Coin.h):

```
#pragma once
//Coin.h
//DO NOT MODIFY THIS SECTION
#ifndef COIN_H
#define COIN_H
#include <iostream>
class Coin {
private:
    std::string sideup;
public:
    Coin() //constructor
    {
        toss();
    }
    std::string getSideUp() //accessor
    {
        return sideup;
    }
    void toss();
};
#endif // COIN_H
```

## File 2 (Named Coin.cpp):

```
#include "Coin.h"
#include <iostream>
#include <ctime>
using namespace std;

void Coin::toss()
{
    //1. Generate a random number that is either 0 or 1
    int num = rand() % 2;

    //2. Set sideup member variable to either heads or tails depending on the
    random number
    if (num == 0)
        Coin::sideup = "heads";
    else
        Coin::sideup = "tails";
}
```

## File 3 (Named lab8\_2.cpp):

```
#include <iostream>
#include "Coin.h"
using namespace std;

int main()
{
    srand(time(nullptr));

    //1. Create an object of the Coin class
    Coin c1;

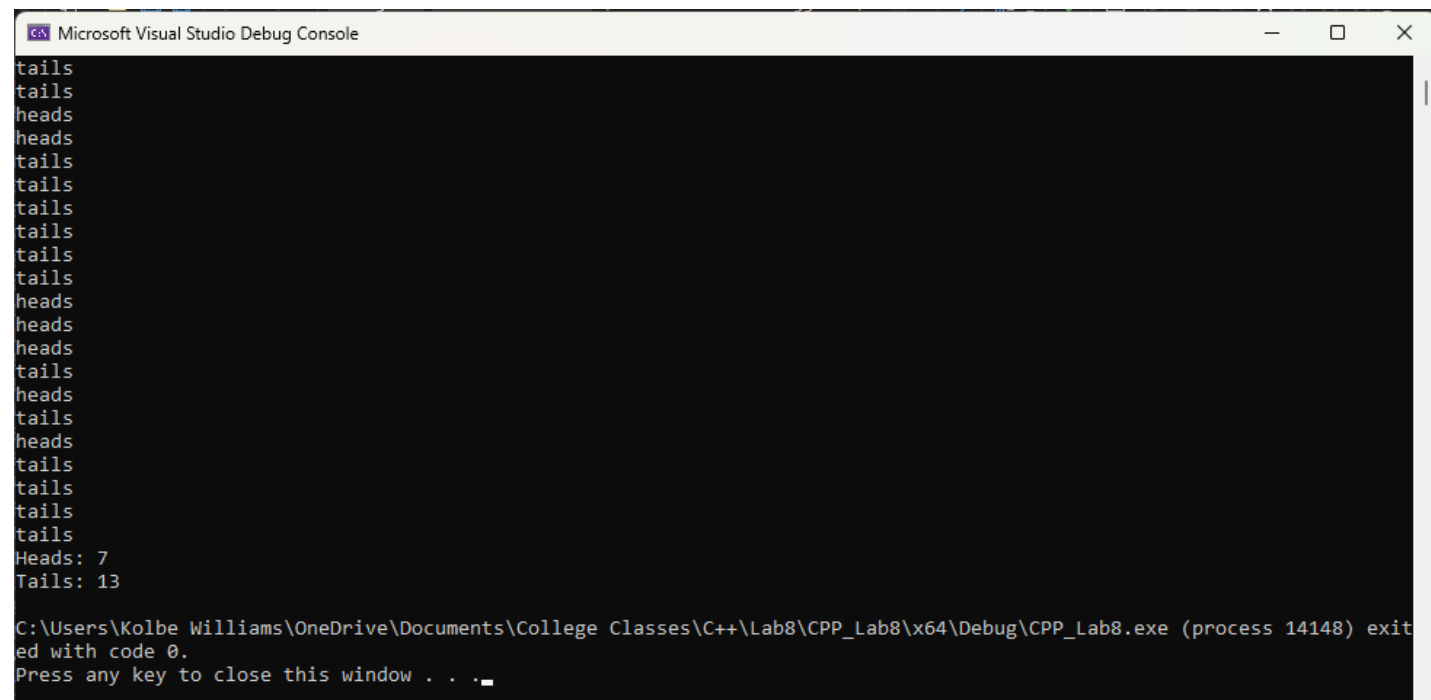
    //2. Declare and initialize heads and tails count variables
    int heads = 0, tails = 0;

    //3. Print begininng string value of sideup member variable
    cout << c1.getSideUp() << endl;

    //4. Use a for loop to toss the coin 20 times and record the number of heads
    and tails
    for (int i = 0; i < 20; i++)
    {
        c1.toss();
        if (c1.getSideUp().compare("heads") == 0)
            heads++;
        else
            tails++;
        cout << c1.getSideUp() << endl;
    }

    //5. Display the results
    cout << "Heads: " << heads << endl;
    cout << "Tails: " << tails << endl;
    return 0;
}
```

## Ouput:



A screenshot of the Microsoft Visual Studio Debug Console window. The window has a title bar with the Visual Studio logo and the text "Microsoft Visual Studio Debug Console". The console area is black with white text. The output shows a sequence of "tails" and "heads" results, followed by a summary "Heads: 7" and "Tails: 13". At the bottom, it shows the program path and exit code, followed by a prompt to press any key to close the window.

```
Microsoft Visual Studio Debug Console
tails
tails
heads
heads
tails
tails
tails
tails
tails
tails
heads
heads
heads
tails
heads
tails
heads
tails
tails
tails
Heads: 7
Tails: 13

C:\Users\Kolbe Williams\OneDrive\Documents\College Classes\C++\Lab8\CPP_Lab8\x64\Debug\CPP_Lab8.exe (process 14148) exited with code 0.
Press any key to close this window . . .
```