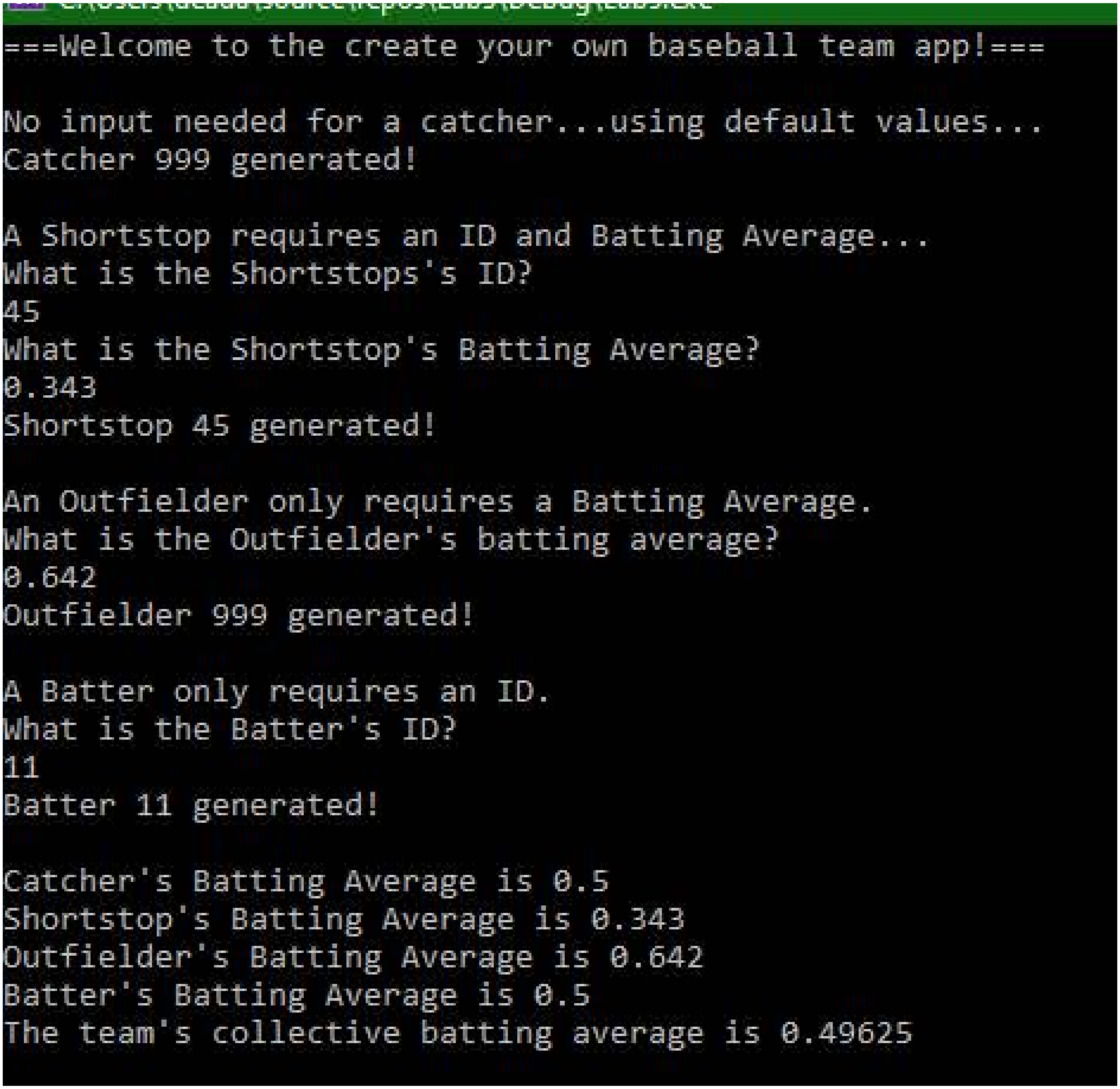
```
/*
Daniel Avila    March 4th 2020  Section 19
Lab 5: Overloaded Classes
Description: In this lab, we implement a Class and with its default constructor, we overload it 3 times
*/
#ifndef BASEBALL_H//start of class
#define BASEBALL_H
#include <iostream>
#include <string>
using namespace std;
//Class for everything related to the umbrella of a baseballplayer(eg. position, bavg, id)
class BaseBallPlayer
{
private:
    string role;//for the postion name
    int ID;//for the number identification
    double BAVG;//for the average between 0-1

public:
    BaseBallPlayer();//default constructor
    BaseBallPlayer(int ssID, double ssBAVG);//overloaded shortstop constructor
    BaseBallPlayer(double ofBAVG);//overloaded outfielder constructor
    BaseBallPlayer(int bID);//overloaded batter constructor
    void displayBAVG();//for collective and individual batting average
    double getBAVG();//used for the display of all's batting average

};
BaseBallPlayer::BaseBallPlayer()//Default constructor
{
    role = "Catcher";//Default role of catcher
    ID = 999;//Default ID
    BAVG = 0.5;//Default batting average
    cout << role << " " << ID << " generated!" << endl << endl;//Player and ID are generated
}
BaseBallPlayer::BaseBallPlayer(int ssID, double ssBAVG)//Shortstop constructor
{
    role = "Shortstop";//Role in overloaded constructor 1
    ID = ssID;//ID in overloaded constructor 1
    BAVG = ssBAVG;//BAVG in overloaded constructor 1
    cout << role << " " << ID << " generated!" << endl << endl;//Player and ID are generated
}
BaseBallPlayer::BaseBallPlayer(double ofBAVG)//Outfielder constructor
{
    role = "Outfielder";//Role in overloaded constructor 2
    ID = 999;//ID in overloaded constructor 2
    BAVG = ofBAVG;//BAVG in overloaded constructor 2
    cout << role << " " << ID << " generated!" << endl << endl;//Player and ID are generated
}
BaseBallPlayer::BaseBallPlayer(int bID)//Battter constructor
{
    role = "Batter"; //Role in overloaded constructor 3
    ID = bID; //ID in overloaded constructor 3
    BAVG = 0.5; //BAVG in overloaded constructor 3
    cout << role << " " << ID << " generated!" << endl << endl;//Player and ID are generated
}
double BaseBallPlayer::getBAVG()//Gets player batting average
{
    return BAVG;//returns the batting average that was inputted
}
void BaseBallPlayer::displayBAVG()//Displays the battinga average solely and collectively
{
    cout << role << "'s Batting Average is " << getBAVG() << endl;//iterates through the roles and display's theri average
}
#endif // !BASEBALL_H       end of class
```

```cpp
/*
Daniel Avila    March 4th 2020  Section 19
Lab 5: Overloaded Classes
Description: In this lab, we implement a Class and with its default constructor, we overload it 3 times
*/
#include "Baseball.h"//To include the class on the Baseball header

int main()
{
    int id;//For input variable on the overloaded parameters
    double bavg;//For input variable on the overloaded parameters

    cout << "===Welcome to the create your own baseball team app!===" << endl << endl;
    cout << "No input needed for a catcher...using default values..." << endl;
    BaseBallPlayer Catcher;//Creates a Catcher object with default values

    //Calls overloaded constructor with the necessary parameters(1 double, 1 int)
    cout << "A Shortstop requires an ID and Batting Average..." << endl;
    //BaseBallPlayer Shortstop;
    cout << "What is the Shortstops's ID?" << endl;
    cin >> id;
    cout << "What is the Shortstop's Batting Average?" << endl;
    cin >> bavg;
    BaseBallPlayer Shortstop(id, bavg);//Creates a Shortstop object with default values

    //Calls overloaded constructor with the necessary parameters(1 double)
    cout << "An Outfielder only requires a Batting Average." << endl;
    //BaseBallPlayer Outfielder;//Creates a Outfielder object with default values
    cout << "What is the Outfielder's batting average?" << endl;
    cin >> bavg;//Used to set the Outfielder's Batting Average
    BaseBallPlayer Outfielder(bavg);

    //Calls overloaded constructor with the necessary parameters(1 int)
    cout << "A Batter only requires an ID." << endl;
    //BaseBallPlayer Batter;//Creates a Batter object with default values
    cout << "What is the Batter's ID?" << endl;
    cin >> id;
    BaseBallPlayer Batter(id);

    Catcher.displayBAVG();//Displays the final batting average for the catcher
    Shortstop.displayBAVG();//Displays the final batting average for the shortstop
    Outfielder.displayBAVG();//Displays the final batting average for the outfielder
    Batter.displayBAVG();//Displays the final batting average for the batter
    cout << "The team's collective batting average is ";
    cout << (Catcher.getBAVG() + Shortstop.getBAVG() +  //Used to call the individual player's batting average and then
          Outfielder.getBAVG() + Batter.getBAVG()) / 4; //adds them up and divides all of them by 4 for a collective average
    system("pause>nul");
    return 0;
}
```



```
===Welcome to the create your own baseball team app!===

No input needed for a catcher...using default values...
Catcher 999 generated!

A Shortstop requires an ID and Batting Average...
What is the Shortstops's ID?
45
What is the Shortstop's Batting Average?
0.343
Shortstop 45 generated!

An Outfielder only requires a Batting Average.
What is the Outfielder's batting average?
0.642
Outfielder 999 generated!

A Batter only requires an ID.
What is the Batter's ID?
11
Batter 11 generated!

Catcher's Batting Average is 0.5
Shortstop's Batting Average is 0.343
Outfielder's Batting Average is 0.642
Batter's Batting Average is 0.5
The team's collective batting average is 0.49625
```