OBJECT ORIENTED PROGRAMMING (LAB 2)

EXERCISES

1. Write a program in which a class named student has member variables name, roll_no, semester and section. Create 4 objects of this class to store data of 4 different students, now display data of only those students who belong to section A.

Ques-1.cpp | Ques-2.cpp | Ques-3.cpp | [*] Untitled2

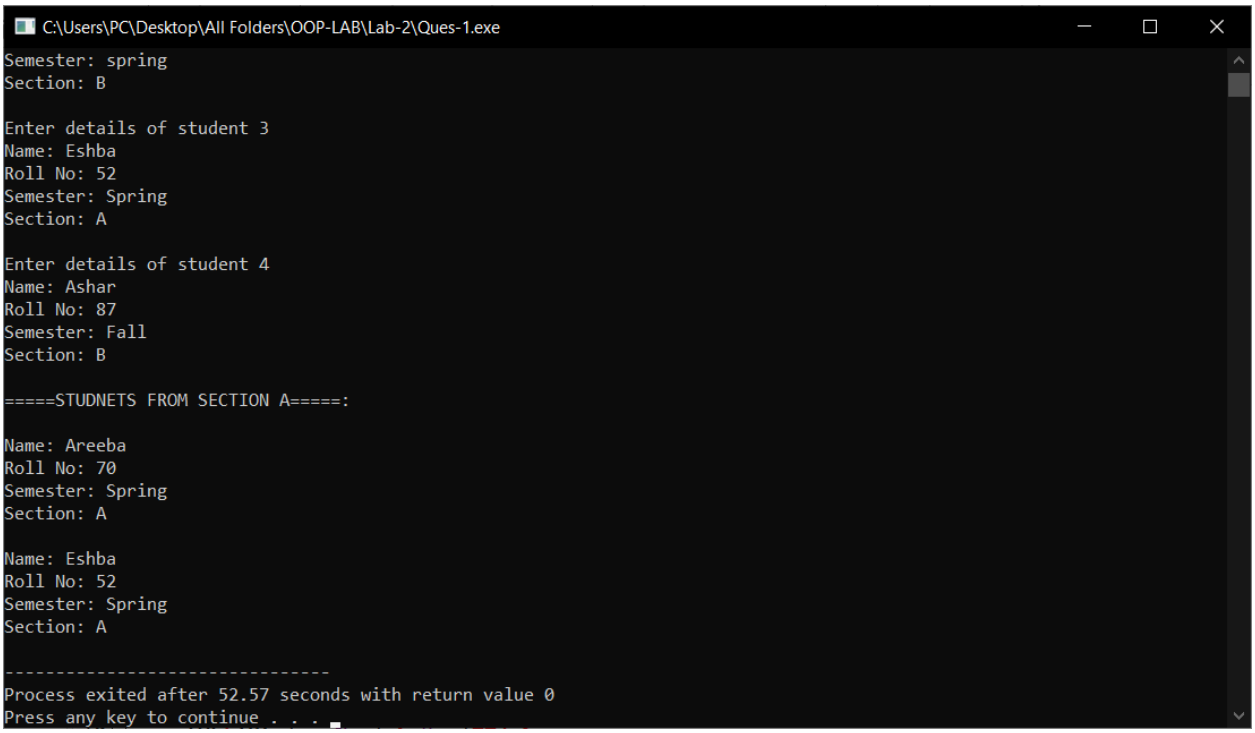
```cpp
class Student {
public:
    string name;
    int roll_no;
    string semester;
    string section;
};

int main() {
    Student s[4];

    for(int i = 0; i < 4; i++) {
        cout << "\nEnter details of student " << i + 1 << endl;
        cout << "Name: ";
        cin >> s[i].name;
        cout << "Roll No: ";
        cin >> s[i].roll_no;
        cout << "Semester: ";
        cin >> s[i].semester;
        cout << "Section: ";
        cin >> s[i].section;
    }

    cout << "\n=====STUDNETS FROM SECTION A=====:\n";
    for(int i = 0; i < 4; i++) {
        if(s[i].section == "A" || s[i].section == "a") {
            cout << "\nName: " << s[i].name;
            cout << "\nRoll No: " << s[i].roll_no;
            cout << "\nSemester: " << s[i].semester;
            cout << "\nSection: " << s[i].section << endl;
        }
    }

    return 0;
}
```



2. You are a programmer for the ABC Bank assigned to develop a class that models the basic workings of a bank account. The class should perform the following tasks:

o Save the account balance.

o Save the number of transactions performed on the account.

o Allow deposits to be made to the account.

o Allow with drawls to be taken from the account.

o Report the current account balance at any time.

o Report the current number of transactions at any time.

Menu

1. Display the account balance

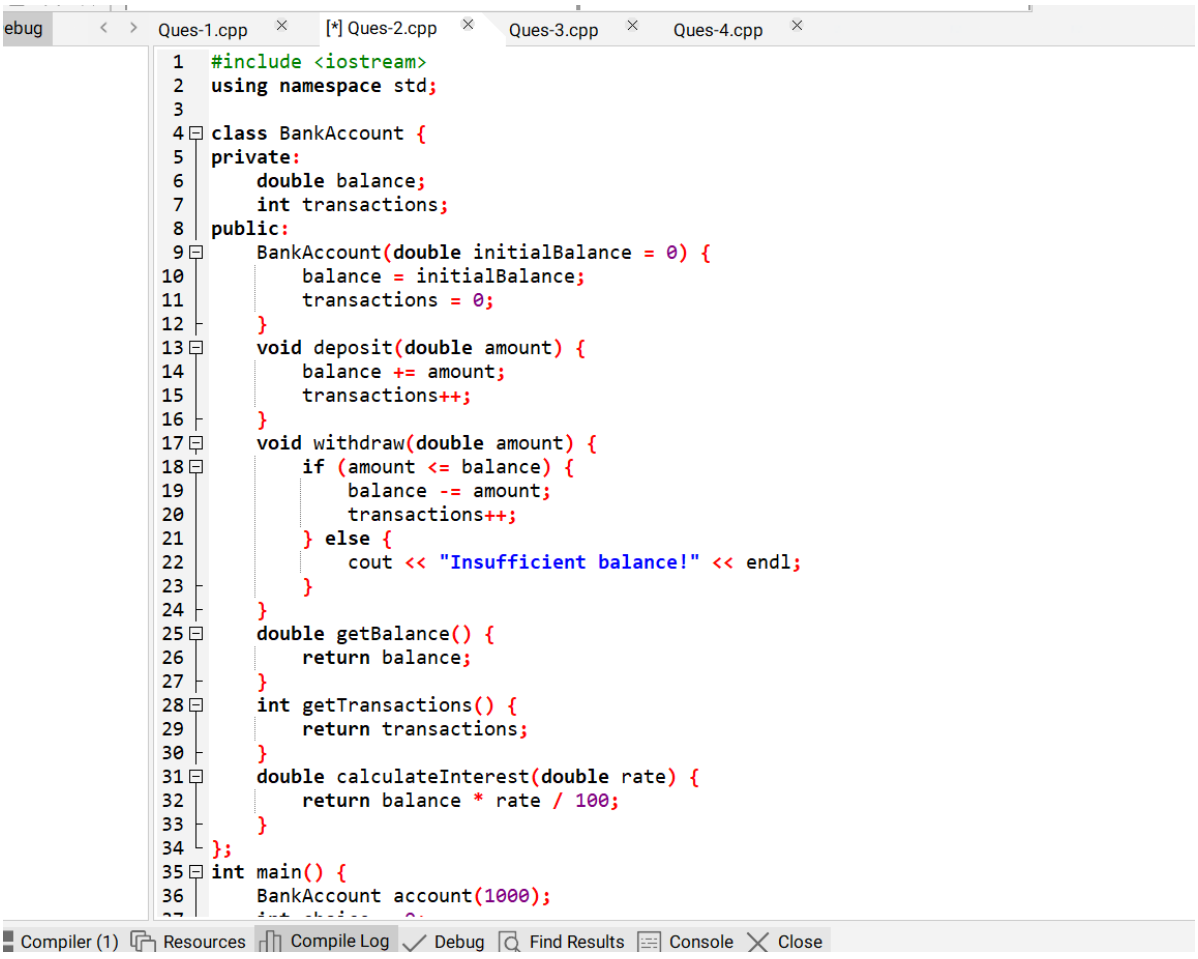
2. Display the number of transactions

3. Display interest earned for this period

4. Make a deposit

5. Make a withdrawal

6. Exit the program



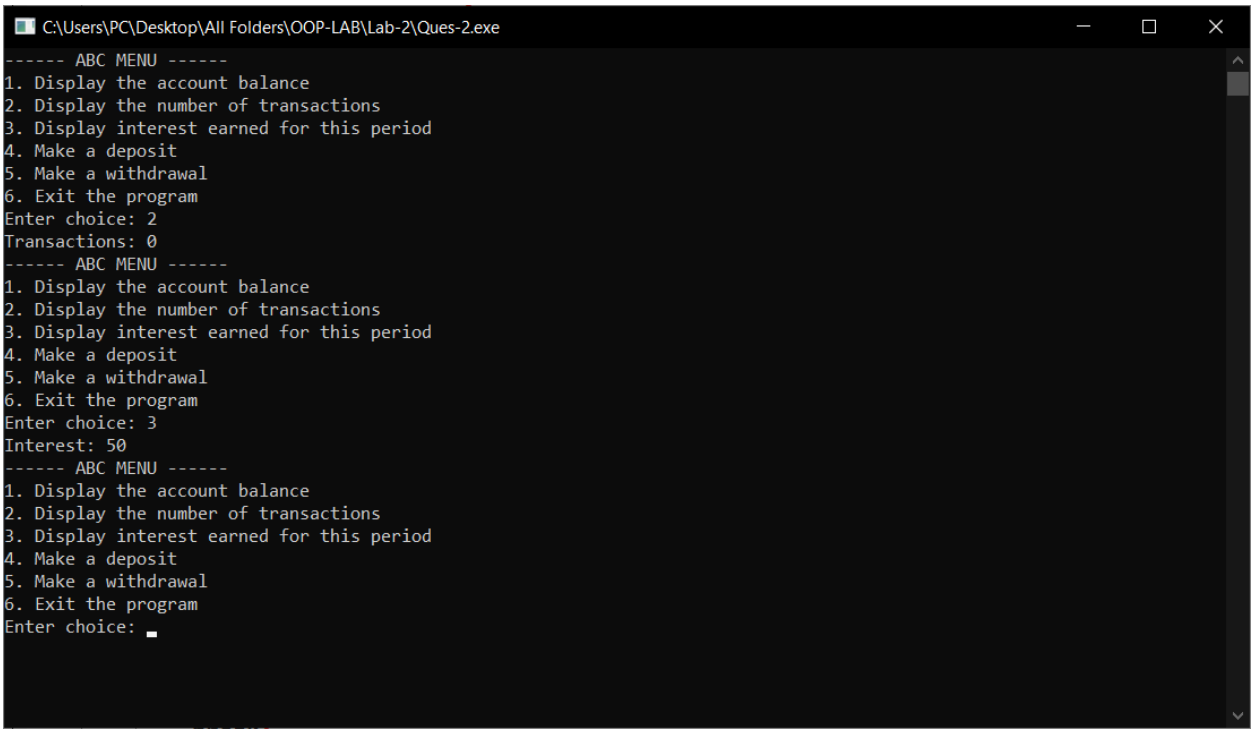
Debug | Ques-1.cpp | [*] Ques-2.cpp | Ques-3.cpp | Ques-4.cpp

```
35  int main() {
36      BankAccount account(1000);
37      int choice = 0;
38      double amount, rate = 5.0;
39
40      while (choice != 6) {
41          cout << "------ ABC MENU ------" << endl;
42          cout << "1. Display the account balance" << endl;
43          cout << "2. Display the number of transactions" << endl;
44          cout << "3. Display interest earned for this period" << endl;
45          cout << "4. Make a deposit" << endl;
46          cout << "5. Make a withdrawal" << endl;
47          cout << "6. Exit the program" << endl;
48          cout << "Enter choice: ";
49          cin >> choice;
50
51          switch (choice) {
52          case 1:
53              cout << "Balance: " << account.getBalance() << endl;
54              break;
55          case 2:
56              cout << "Transactions: " << account.getTransactions() << endl;
57              break;
58          case 3:
59              cout << "Interest: " << account.calculateInterest(rate) << endl;
60              break;
61          case 4:
62              cout << "Enter deposit amount: ";
63              cin >> amount;
64              account.deposit(amount);
65              break;
66          case 5:
```

Compiler (1) | Resources | Compile Log | Debug | Find Results | Console | Close

```
66          case 5:
67              cout << "Enter withdrawal amount: ";
68              cin >> amount;
69              account.withdraw(amount);
70              break;
71          case 6:
72              cout << "Program exited." << endl;
73              break;
74          default:
75              cout << "Invalid choice." << endl;
76          }
77      }
78      return 0;
79  }
```

Resources | Compile Log | Debug | Find Results | Console | Close



3. Create a class called Employee that includes three pieces of information as data members—a first name (type char* (DMA)), a last name (type string) and a monthly salary (type int). Your class should have a setter function that initializes the three data members. Provide a getter function for each data member. If the monthly salary is not positive, set it to 0. Write a test program that demonstrates class Employee's capabilities. Create two Employee objects and display each object's yearly salary. Then give each Employee a 10 percent raise and display each Employee's yearly salary again. Identify and add any other related functions to achieve the said goal.

```cpp
#include <iostream>
#include <cstring>
#include <string>
using namespace std;
class Employee {
private:
    char* firstName;        // DMA
    string lastName;
    int monthlySalary;
public:
    // Constructor
    Employee() {
        firstName = NULL;
        lastName = "";
        monthlySalary = 0;
    }
    // Destructor
    ~Employee() {
        delete[] firstName;
    }
    // Setter function
    void setEmployee(const char* fName, string lName, int salary) {
        delete[] firstName;                    // free old memory
        firstName = new char[strlen(fName)+1];  // allocate memory
        strcpy(firstName, fName);               // copy name
        lastName = lName;
        if (salary > 0)
            monthlySalary = salary;
        else
            monthlySalary = 0;
    }
    // Getters
    const char* getFirstName() const { return firstName; }
    string getLastName() const { return lastName; }
    int getMonthlySalary() const { return monthlySalary; }
```

Resources | Compile Log | Debug | Find Results | Console | Close

```cpp
    int getYearlySalary() const {
        return monthlySalary * 12;
    }
    // Raise function
    void giveRaise(int percent) {
        monthlySalary += (monthlySalary * percent) / 100;
    }
    // Display function
    void displayInfo() const {
        cout << "Employee: " << firstName << " " << lastName << endl;
        cout << "Monthly Salary: " << monthlySalary << endl;
        cout << "Yearly Salary: " << getYearlySalary() << endl;
    }
};
int main() {
    Employee emp1, emp2;

    emp1.setEmployee("Aliza", "Beth", 70000);
    emp2.setEmployee("Areeba", "Khan", 90000);

    cout << "===== Initial Salary =====" << endl;
    emp1.displayInfo();
    cout << endl;
    emp2.displayInfo();
    cout << endl;

    emp1.giveRaise(10);
    emp2.giveRaise(10);

    cout << "===== After 10% Raise =====" << endl;
    emp1.displayInfo();
    cout << endl;
    emp2.displayInfo();

    return 0;
}
```

Resources | Compile Log | Debug | Find Results | Console | Close

```
C:\Users\PC\Desktop\All Folders\OOP-LAB\Lab-2\Ques-3.exe
===== Initial Salary =====
Employee: Aliza Beth
Monthly Salary: 70000
Yearly Salary: 840000

Employee: Areeba Khan
Monthly Salary: 90000
Yearly Salary: 1080000

===== After 10% Raise =====
Employee: Aliza Beth
Monthly Salary: 77000
Yearly Salary: 924000

Employee: Areeba Khan
Monthly Salary: 99000
Yearly Salary: 1188000

--------------------------------
Process exited after 0.4705 seconds with return value 0
Press any key to continue . . .
```

4. Write C++ code to represent a hitting game by using OOP concept. The details are as follows: This game is being played between two teams (i.e. your team and the enemy team). The total number of players in your team is randomly generated and stored accordingly. The function generates a pair of numbers and matches each pair. If the numbers get matched, the following message is displayed: "Enemy got hit by your team!" Otherwise, the following message is displayed: "You got hit by the enemy team!" The number of hits should be equal to the number of players in your team. The program should tell the final result of your team by counting the hits of both the teams. Consider the following sample output:

```
Total No. Of Players in your team: 3

Pair of numbers:
Number1: 3
Number2: 3
Enemy got hit by your team!

Pair of numbers:
Number1: 1
Number2: 1
Enemy got hit by your team!

Pair of numbers:
Number1: 5
Number2: 1
You got hit by the enemy team!
Game Over! You won
```

Ques-1.cpp Ques-2.cpp Ques-3.cpp [*] Ques-4.cpp

```cpp
#include <iostream>
#include <cstdlib>
#include <ctime>
using namespace std;

class Team {
private:
    int players;
    int hits;
public:
    Team(int numPlayers = 0) {
        players = numPlayers;
        hits = 0;
    }
    void setPlayers(int numPlayers) {
        players = numPlayers;
    }
    int getPlayers() {
        return players;
    }
    void addHit() {
        hits++;
    }
    int getHits() {
        return hits;
    }
};
class HittingGame {
private:
    Team yourTeam;
    Team enemyTeam;
public:
    HittingGame() {
        srand(time(0));
        int numPlayers = rand() % 5 + 3; // Random between 3-7 players
        yourTeam.setPlayers(numPlayers);
        enemyTeam.setPlayers(numPlayers);
    }

    void play() {
        cout << "Total No. Of Players in your team: " << yourTeam.getPlayers() <<

        for (int i = 1; i <= yourTeam.getPlayers(); i++) {
            int number1 = rand() % 5 + 1;
            int number2 = rand() % 5 + 1;

            cout << "Pair of numbers:\n";
            cout << "Number1:" << number1 << "\n";
            cout << "Number2: " << number2 << "\n";

            if (number1 == number2) {
                cout << "Enemy got hit by your team!\n";
                yourTeam.addHit();
            } else {
                cout << "You got hit by the enemy team!\n";
                enemyTeam.addHit();
            }
        }

        cout << "Game Over! ";
        if (yourTeam.getHits() > enemyTeam.getHits()) {
            cout << "You won\n";
        } else if (yourTeam.getHits() < enemyTeam.getHits()) {
            cout << "Enemy won\n";
        } else {
            cout << "It's a tie\n";
        }
    }
};

int main() {
    HittingGame game;
```

```
65            } else {
66                cout << "It's a tie\n";
67            }
68        }
69    };
70
71    int main() {
72        HittingGame game;
73        game.play();
74        return 0;
75    }
```

```
C:\Users\PC\Desktop\All Folders\OOP-LAB\Lab-2\Ques-4.exe
Total No. Of Players in your team: 6
Pair of numbers:
Number1:2
Number2: 1
You got hit by the enemy team!
Pair of numbers:
Number1:4
Number2: 2
You got hit by the enemy team!
Pair of numbers:
Number1:4
Number2: 3
You got hit by the enemy team!
Pair of numbers:
Number1:1
Number2: 2
You got hit by the enemy team!
Pair of numbers:
Number1:5
Number2: 5
Enemy got hit by your team!
Pair of numbers:
Number1:2
Number2: 1
You got hit by the enemy team!
Game Over! Enemy won

--------------------------------
Process exited after 0.664 seconds with return value 0
Press any key to continue . . .
```

EXAMPLES

Ques-1.cpp | Ques-2.cpp | Ques-3.cpp | Ques-4.cpp | Example-1.cpp

```cpp
#include <iostream>
using namespace std;
class firstprogram {
private: // we declare a as private to hide it from outside
int number1;
public:
void set(int input1){//set() function to set the value of a
number1 = input1;
}
int get() { // get() function to return the value of a
return number1;
}
};
// main function
int main() {
firstprogram myInstance;
myInstance.set(10);
cout<<myInstance.get()<<endl;
return 0;
}
```

```
C:\Users\PC\Desktop\All Folders\OOP-LAB\Lab-2\Example-1.exe
10

--------------------------------
Process exited after 0.5214 seconds with return value 0
Press any key to continue . . .
```