Don Bosco Institute of Technology, Kurla(W)
Department of Electronics and Tele-Communication Engineering
ECL304 - Skill Lab: C++ and Java Programming
Sem III
2021-22

Lab Number:	**3**
Student Name:	**Simran Santosh Koparkar**
Roll No :	**41**

Title:

3.1 Write a C++ program to Create a class Student with two method getData() and printData(). getData() to get the value from the user and display the data in printData(). Create the two objects s1 ,s2 to declare and access the values from class StudentTest.

3.2 Write a C++ program for Basic bank Management System

Learning Objective:

- Students will be able to write C++ and java program for using classes and objects.

Learning Outcome:

- Ability to execute a simple C++ and Java program by accepting and displaying values using functions
- Understanding the classes and objects concept in C++ and Java.

Course Outcome:

ECL304.1	Understand object-oriented programming concepts and implement using C++ and Java

Theory:

Difference between procedural and object oriented language

Procedural Programming: Procedural Programming can be defined as a programming model which is derived from structured programming, based upon the concept of calling procedure. Procedures, also known as routines, subroutines or functions, simply consist of a series of computational steps to be carried out. During a program's execution, any given procedure might be called at any point, including by other procedures or itself.

Object oriented programming can be defined as a programming model which is based upon the concept of objects. Objects contain data in the form of attributes and code in the form of methods. In object oriented programming, computer programs are designed using the concept of objects that interact with real world. Object oriented programming languages are various but the most popular ones are class-based, meaning that objects are instances of classes, which also determine their types.

Application of object orientation

OOP can also be used in manufacturing and design applications, as it allows people to reduce the effort involved. For instance, it can be used while designing blueprints and flowcharts. OOP makes it possible for the designers and engineers to produce these flowcharts and blueprints accurately

Brief introduction to C++ and Java

C++ is a statically typed, compiled, general-purpose, case-sensitive, free-form programming language that supports procedural, object-oriented, and generic programming. C++ is regarded as a middle-level language, as it comprises a combination of both high-level and low-level language features.

Java is a class-based, object-oriented programming language and is designed to have as few implementation dependencies as possible. A general-purpose programming language made for developers to write once run anywhere that is compiled Java code can run on all platforms that support Java

Algorithm :	**Step 1:START** **Step 2:Declare variables name and branch as string,rollno as int,cgpa as float** **Step 3:Create two objects s1 and s2 of class student** **Step 4:Call function getdata()** **Read name,branch,rollno,cgpa from user** **Step 5:Call function printdata()** **Print the values accepted from user** **Step 6:STOP**
Program:	**#include<iostream>** **using namespace std;**

```
class Student
{
        public:
        int rollnum;
        string name;
        string branch;
        float cgpa;

        int getdata()
        {
                cout<<"enter your rollno :"<<endl;
      cin>>rollnum;
      cout<<"enter your name :"<<endl;
      cin>>name;
      cout<<"enter your branch :"<<endl;
      cin>>branch;
      cout<<"enter the CGPA  :"<<endl;
      cin>>cgpa;
        }

        int printdata()
        {

      cout<<"Marksheet of student is as follows : "<<endl;
      cout<<"Roll number : "<<rollnum<<endl;
      cout<<"Name: "<<name<<endl;
      cout<<"Branch: "<<branch<<endl;
      cout<<"Your CGPA : "<<cgpa<<endl;
```

	} **};** **int main()** **{** **Student s1,s2;** **s1.getdata();** **s1.printdata();** **s2.getdata();** **s2.printdata();** **return 0;** **}**
Input given:	**19** **Simran** **Extc** **8.7**

Output Screenshot:	

Algorithm :	**STEP 1: Start** **STEP 2:Enter account no** **STEP 3: Enter name** **STEP 4:Account info** **STEP 5:Stop**
Program:	**#include <iostream>** **using namespace std;** **// class** **class Bank {** **private:** **int acno;** **char name[30];** **long balance;**

```
public:
   void OpenAccount()
   {
      cout << "Enter Account Number: ";
      cin >> acno;
      cout << "Enter Name: ";
      cin >> name;
      cout << "Enter  Balance: ";
      cin >> balance;
   }
   void ShowAccount()
   {
      cout << "Account Number: " << acno << endl;
      cout << "Name: " << name << endl;
      cout << "Balance: " << balance << endl;
   }
   void Deposit()
   {
      long amt;
      cout << "Enter Amount U want to deposit? ";
      cin >> amt;
      balance = balance + amt;
   }
   void Withdrawal()
   {
      long amt;
      cout << "Enter Amount U want to withdraw? ";
      cin >> amt;
```

```
        if (amt <= balance)
            balance = balance - amt;
        else
            cout << "Less Balance..." << endl;
    }
    int Search(int);
};

int Bank::Search(int a)
{
    if (acno == a) {
        ShowAccount();
        return (1);
    }
    return (0);
}

// main code
int main()
{
    Bank C[3];

    int found = 0, a, ch, i;
    for (i = 0; i <= 2; i++) {
        C[i].OpenAccount();
    }

    do {
```

```
// display options
cout << "\n\n1:Display All\n2:By Account No\n3:Deposit\n4:Withdraw\n5:Exit" << endl;


// user input
cout << "Please input your choice: ";
cin >> ch;


switch (ch) {
case 1: // displating account info
   for (i = 0; i <= 2; i++) {
      C[i].ShowAccount();
   }
   break;
case 2: // searching the record
   cout << "Account Number? ";
   cin >> a;
   for (i = 0; i <= 2; i++) {
      found = C[i].Search(a);
      if (found)
         break;
   }
   if (!found)
      cout << "Record Not Found" << endl;
   break;
case 3: // deposit operation
   cout << "Account Number To Deposit Amount? ";
   cin >> a;
```

```
        for (i = 0; i <= 2; i++) {
            found = C[i].Search(a);
            if (found) {
                C[i].Deposit();
                break;
            }
        }
        if (!found)
            cout << "Record Not Found" << endl;
        break;
    case 4: // withdraw operation
        cout << "Account Number To Withdraw Amount? ";
        cin >> a;
        for (i = 0; i <= 2; i++) {
            found = C[i].Search(a);
            if (found) {
                C[i].Withdrawal();
                break;
            }
        }
        if (!found)
            cout << "Record Not Found" << endl;
        break;
    case 5: // exit
        cout << "Have a nice day" << endl;
        break;
    default:
        cout << "Wrong Option" << endl;
```

	} **} while (ch != 5);** **return 0;** **}**
Input given:	**81** **Ram** **700**
Output Screenshot: