# Department of Computing

**CS-213: Advanced Programming**

**Class: BSCS 7AB**

# Lab Quiz # 01

# Task

Write a program of bank management system to manage the account information using inheritance concept.

Create a class “Bank Account” with the customer\_name, account\_number etc. as member variables. Create the derived classes for two types of accounts i.e. current and saving. The derived classes will update the balance and handle the deposit and withdraw cases. Customers should be able to get updated balance after deposit and withdrawal amounts.

**Answer:**

|  |
| --- |
| Solution |
| Task Code:  // ConsoleApplication1.cpp : Defines the entry point for the console application.  //  #include "stdio.h"  #include <iostream>  #include <string.h>  using namespace std;  class Bank\_Account{  private:  int account\_number;  float balance;  string cname;  public:  Bank\_Account( string name, int num, float bal){  cname = name;  account\_number = num;  balance = bal;  }  void display();  };  void Bank\_Account::display()  {  cout << "\n Customer Info: ";  cout << "\n Name: " << cname;  cout << "\n Account No. : " << account\_number;  cout << "\n Balance : " << balance;  cout << "\n END!";  }  class Current : public Bank\_Account{  private:  float balance;  public:  Current(float bal, char name, int num) : Bank\_Account(name, num, bal){  balance = bal;  }  void deposit();  void withdraw();  };  void Current::deposit(){  float amount;  cout << "\n Enter Deposit Amount = ";  cin >> amount;  balance += amount;  }  void Current::withdraw(){  float amount;  cout << "\n Enter Withdraw Amount = ";  cin >> amount;  if (amount>balance)  cout << "\n Cannot Withdraw Amount";  else  balance -= amount;  }  class Savings : public Bank\_Account{  private:  float balance;  public:  Savings(float bal, char name, int num) : Bank\_Account(name, num, bal){  balance = bal;  }  void deposit();  void withdraw();  };  void Savings::deposit(){  float amount;  cout << "\n Enter Deposit Amount = ";  cin >> amount;  balance += amount;  }  void Savings::withdraw(){  float amount;  int withdraw\_limit = 3000;  cout << "\n Enter Withdraw Amount = ";  cin >> amount;  if (amount>balance)  cout << "\n Cannot Withdraw Amount";  else if (amount > withdraw\_limit)  cout << "\n Cannot Withdraw Large Amount";  else  balance -= amount;  }  int main()  {  int acc\_no;  string name;  float balance;  int option;  cout << "\n1. New Account \n 2. Existing Account";  cin >> option;  cout << "\n Enter Details: \n";  cout << "-----------------------";  cout << "\n Account No. ";  cin >> acc\_no;  cout << "\n Name : ";  cin >> name;  if (option == 1){  cout << "\n Balance : ";  cin >> balance;  cout << "\n 1.Savings \n 2.Current";  cin >> option;  if (option == 1){  Savings(balance, name, acc\_no);  }  }  else if (option == 2){  cout << "\n 1.Savings \n 2.Current";  }  }  Task Output Screenshot: |

### Deliverables

Compile a single word document by filling in the solution part and submit this Word file on LMS.