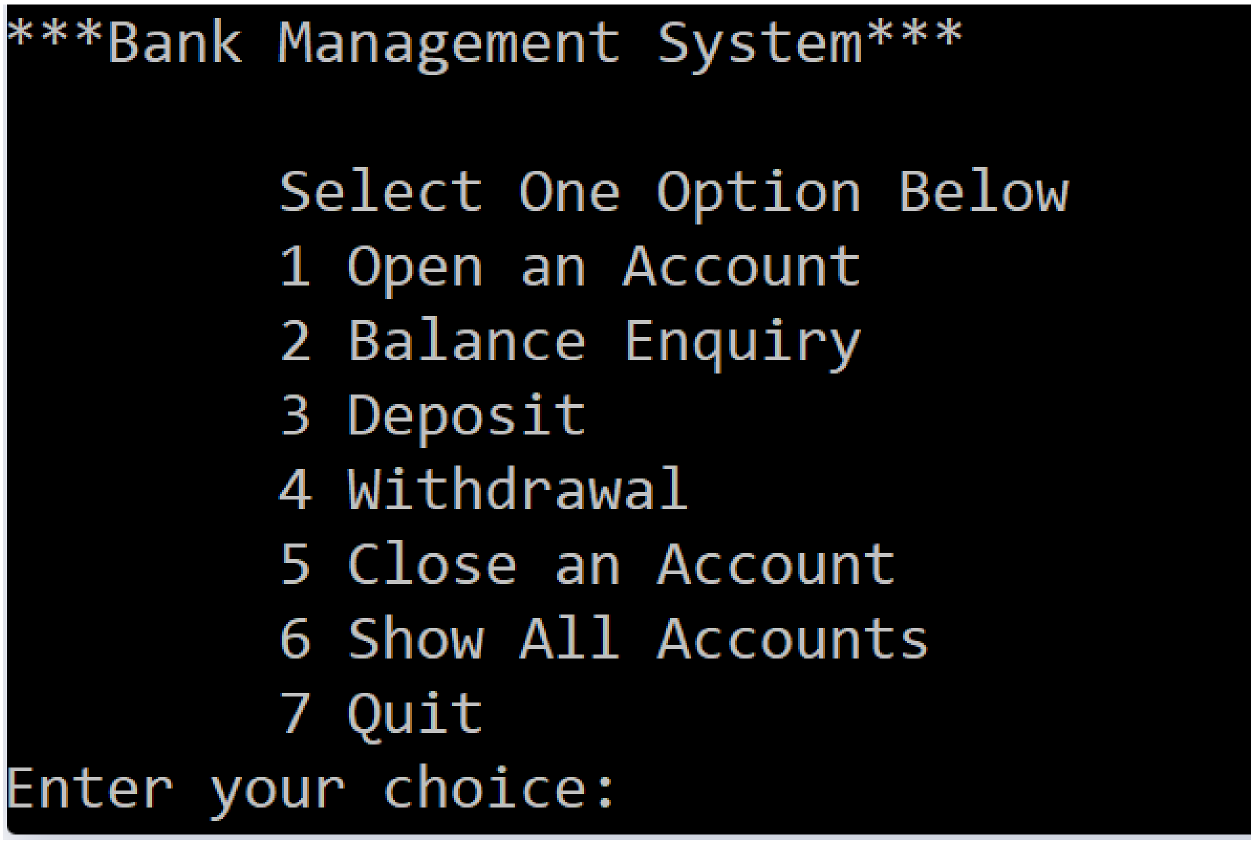
Major Functionalities Used in Bank Management System

The following are the basic characteristics of the bank management system:

- **MAIN SCREEN**

  When you start the project from any compiler or by double-clicking the executable.exe file, you'll see the screen shown below.

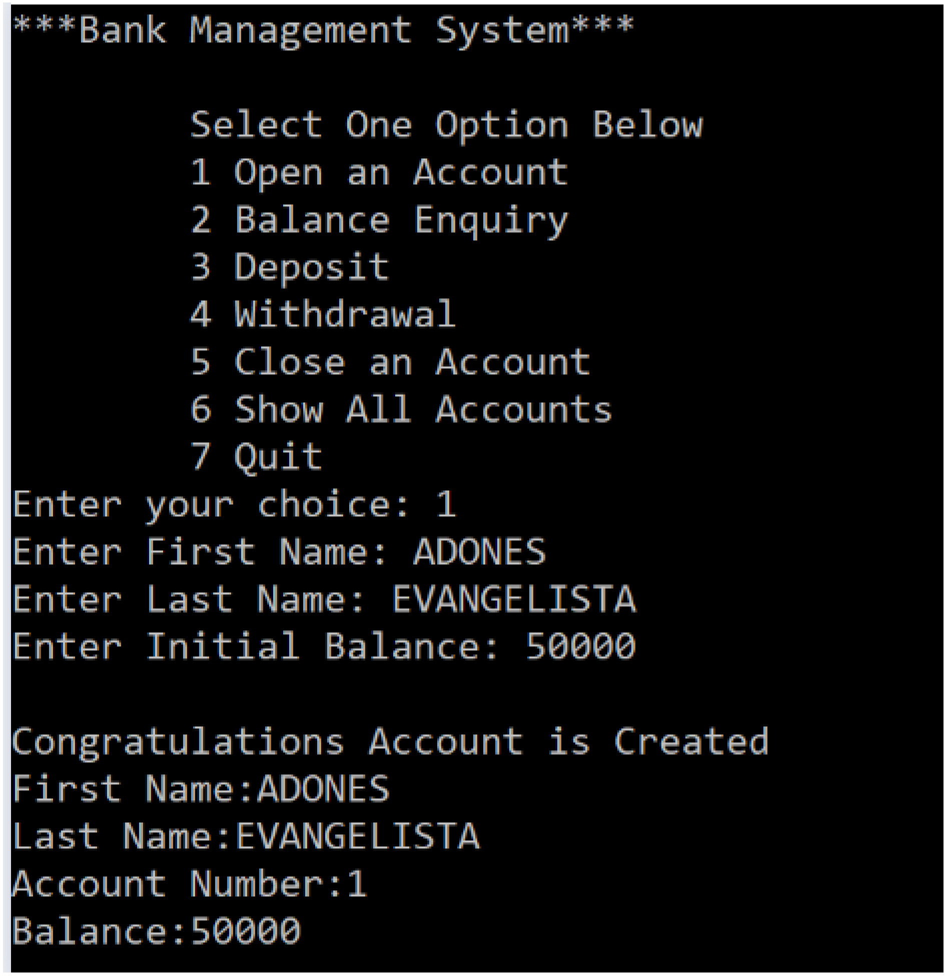


```
***Bank Management System***

        Select One Option Below
        1 Open an Account
        2 Balance Enquiry
        3 Deposit
        4 Withdrawal
        5 Close an Account
        6 Show All Accounts
        7 Quit
Enter your choice:
```

OPEN AN ACCOUNT

When the client selects number 1 as their option, the following screen appears such as first name, last name, and initial balance.



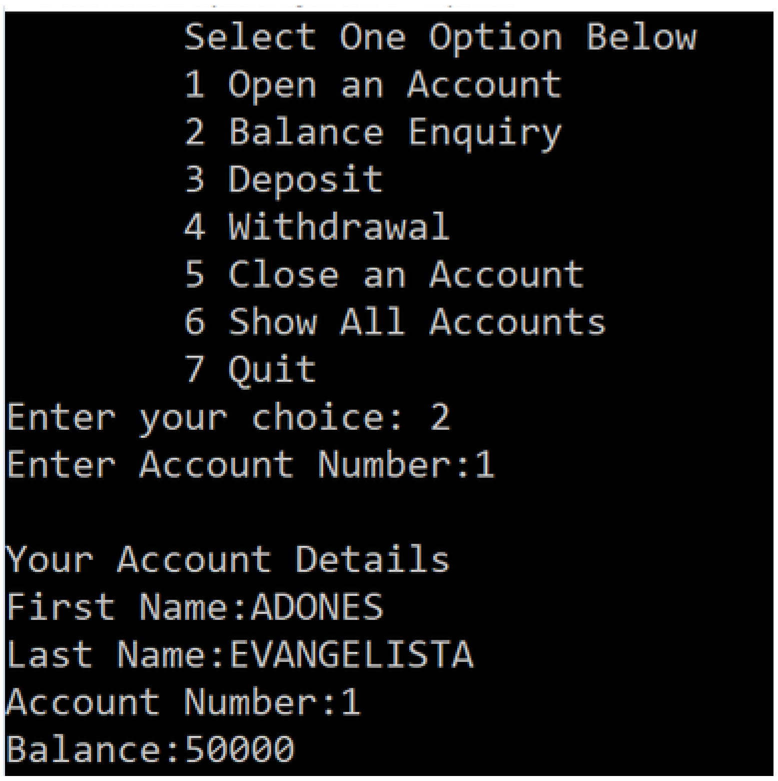
```
***Bank Management System***

        Select One Option Below
        1 Open an Account
        2 Balance Enquiry
        3 Deposit
        4 Withdrawal
        5 Close an Account
        6 Show All Accounts
        7 Quit
Enter your choice: 1
Enter First Name: ADONES
Enter Last Name: EVANGELISTA
Enter Initial Balance: 50000

Congratulations Account is Created
First Name:ADONES
Last Name:EVANGELISTA
Account Number:1
Balance:50000
```

BALANCE ENQUIRY

If the client selects number 2 as their option it means he/she choose a balance enquiry for their account, the following screen appears and the system will ask you to enter your registered account.

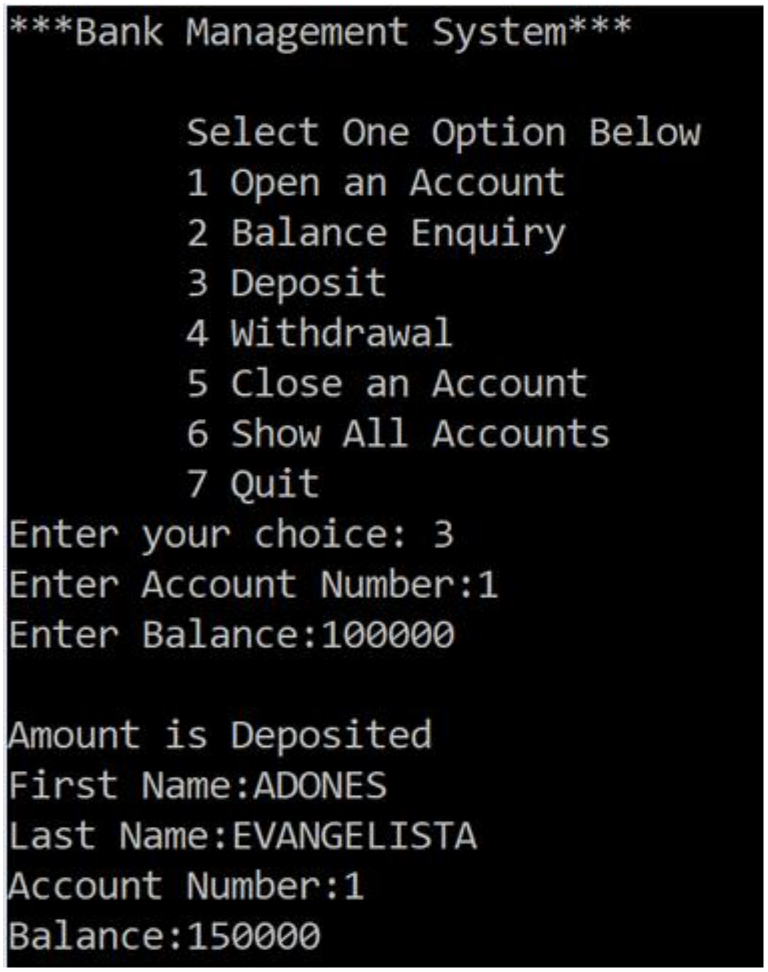


```
        Select One Option Below
        1 Open an Account
        2 Balance Enquiry
        3 Deposit
        4 Withdrawal
        5 Close an Account
        6 Show All Accounts
        7 Quit
Enter your choice: 2
Enter Account Number:1

Your Account Details
First Name:ADONES
Last Name:EVANGELISTA
Account Number:1
Balance:50000
```

DEPOSIT

When the client selects number 3 as their option it means he/she choose to deposit for their account, the following screen appears and the system will ask you to enter your registered account.



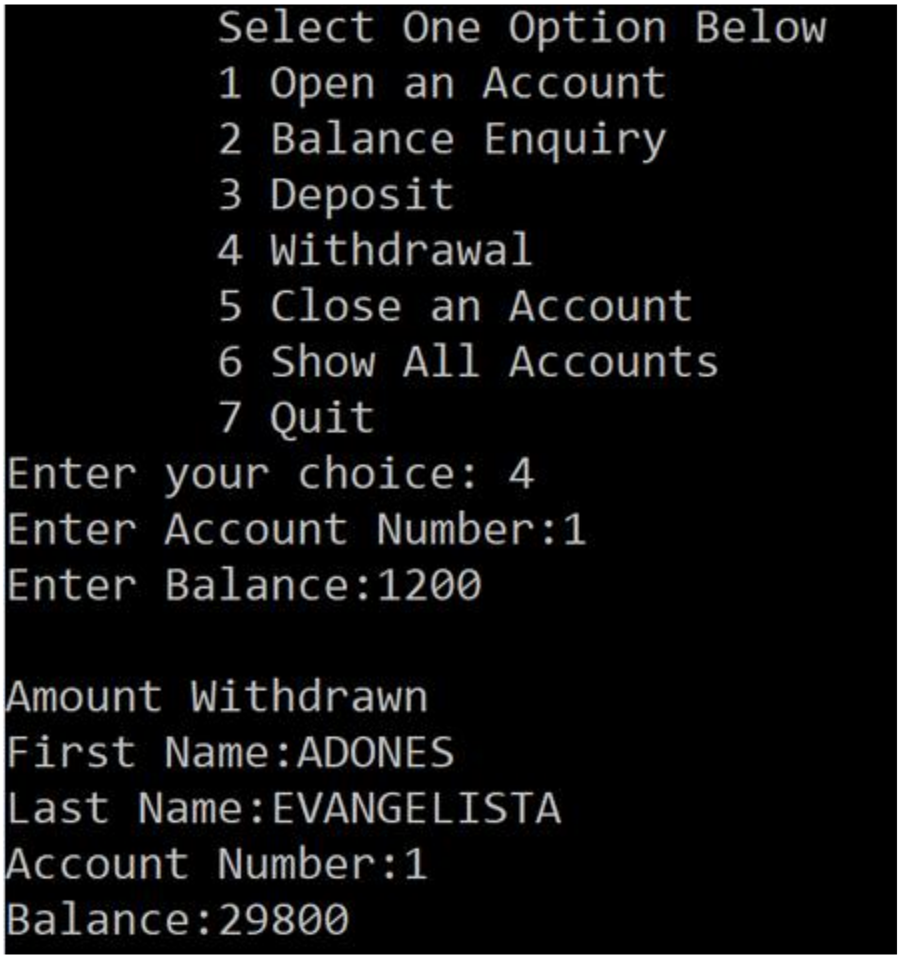
```
***Bank Management System***

        Select One Option Below
        1 Open an Account
        2 Balance Enquiry
        3 Deposit
        4 Withdrawal
        5 Close an Account
        6 Show All Accounts
        7 Quit
Enter your choice: 3
Enter Account Number:1
Enter Balance:100000

Amount is Deposited
First Name:ADONES
Last Name:EVANGELISTA
Account Number:1
Balance:150000
```

WITHDRAWAL

When the client selects number 4 as their option it means he/she choose to withdraw for their account, the following screen appears and the system will ask you to enter your registered account.

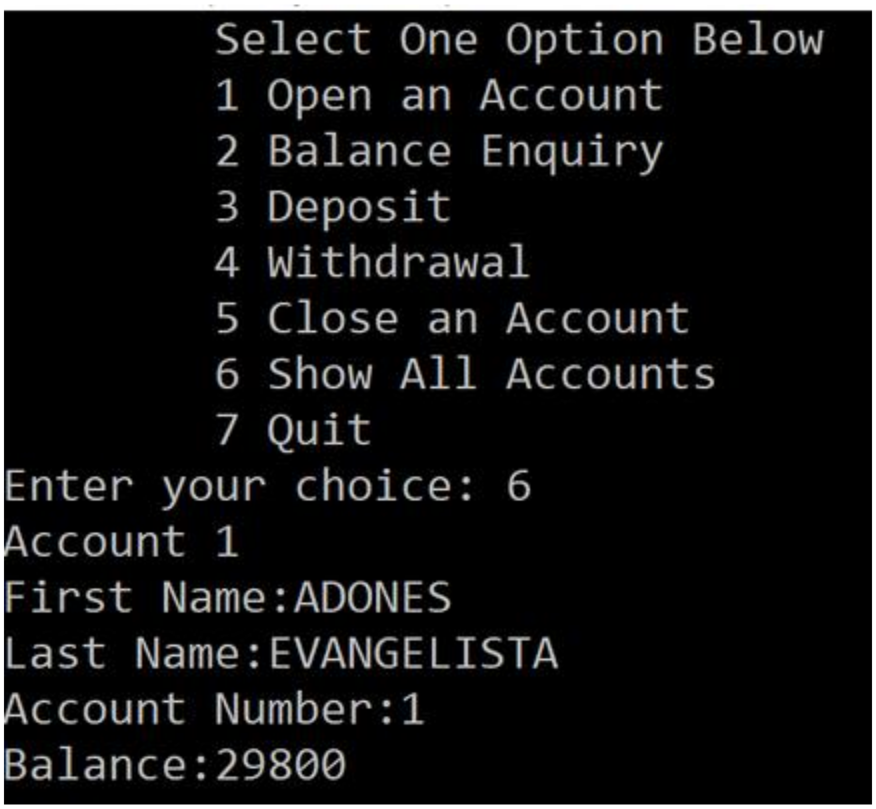


```
        Select One Option Below
        1 Open an Account
        2 Balance Enquiry
        3 Deposit
        4 Withdrawal
        5 Close an Account
        6 Show All Accounts
        7 Quit
Enter your choice: 4
Enter Account Number:1
Enter Balance:1200

Amount Withdrawn
First Name:ADONES
Last Name:EVANGELISTA
Account Number:1
Balance:29800
```

SHOW ALL ACCOUNTS

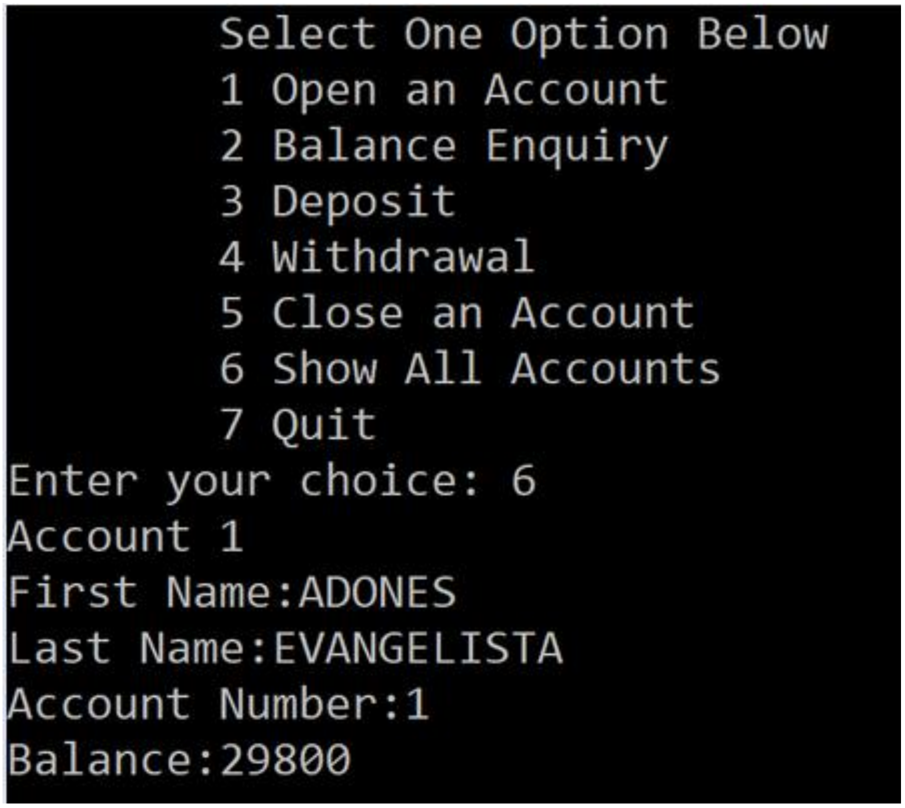
When the client selects number 6 as their option it means he/she choose to Show all Accounts for their account, the following screen appears and the system will ask you to enter your registered account.



```
         Select One Option Below
         1 Open an Account
         2 Balance Enquiry
         3 Deposit
         4 Withdrawal
         5 Close an Account
         6 Show All Accounts
         7 Quit
Enter your choice: 6
Account 1
First Name:ADONES
Last Name:EVANGELISTA
Account Number:1
Balance:29800
```

CLOSE AN ACCOUNT

When the client selects number 5 as their option it means he/she choose to close their account, the following screen appears and the system will ask you to enter your registered account.



```
        Select One Option Below
        1 Open an Account
        2 Balance Enquiry
        3 Deposit
        4 Withdrawal
        5 Close an Account
        6 Show All Accounts
        7 Quit
Enter your choice: 6
Account 1
First Name:ADONES
Last Name:EVANGELISTA
Account Number:1
Balance:29800
```

SOURCE

```
#include<iostream>

#include<fstream>

#include<cstdlib>

#include<vector>

#include<map>

using namespace std;

#define MIN_BALANCE 100

class deficient_funds{};

class Cl_Accounts

{
```

```
private:

long Accnt_No;

string client_fname;

string client_lname;

float client_balance;

static long Nxt_Accnt_No;


public:

Cl_Accounts(){}

Cl_Accounts(string fname,string lname,float client_balance);

long getAccNo(){return Accnt_No;}

string getFName(){return client_fname;}

string getLName(){return client_lname;}

float getBlnce(){return client_balance;}

void Deposit(float amount);

void Withdraw(float amount);

static void setLstAccntNo(long Accnt_No);

static long getLstAccntNo();

friend ofstream & operator<<(ofstream &ofs,Cl_Accounts &acc);

friend ifstream & operator>>(ifstream &ifs,Cl_Accounts &acc);

friend ostream & operator<<(ostream &os,Cl_Accounts &acc);

};

long Cl_Accounts::Nxt_Accnt_No=0;


class Bank
```

```
{

private:

map<long,Cl_Accounts> accounts_cl;

public:

Bank();

Cl_Accounts Cl_Open_Account(string fname,string lname,float balance);

Cl_Accounts Cl_Balance_Enquiry(long Account_no);

Cl_Accounts Deposit(long Account_no,float amt);

Cl_Accounts Withdraw(long Account_no,float amt);

void CloseAccount(long Account_no);

void ShowAllAccounts();

~Bank();

};

int main()

{

Bank b;

Cl_Accounts acc;

int option;

string fname,lname;

long account_no;

float blnced;

float amnts;

cout<<"***Bank Management System***"<<endl;

do

{
```

```
cout<<"\n\tSelect One Option Below ";

cout<<"\n\t1 Open an Account";

cout<<"\n\t2 Balance Enquiry";

cout<<"\n\t3 Deposit";

cout<<"\n\t4 Withdrawal";

cout<<"\n\t5 Close an Account";

cout<<"\n\t6 Show All Accounts";

cout<<"\n\t7 Quit";

cout<<"\nEnter your choice: ";

cin>>option;

switch(option)

{

case 1:

cout<<"Enter First Name: ";

cin>>fname;

cout<<"Enter Last Name: ";

cin>>lname;

cout<<"Enter Initial Balance: ";

cin>>blnced;

acc=b.Cl_Open_Account(fname,lname,blnced);

cout<<endl<<"Congratulations Account is Created"<<endl;

cout<<acc;

break;

case 2:

cout<<"Enter Account Number:";
```

```
cin>>account_no;

acc=b.Cl_Balance_Enquiry(account_no);

cout<<endl<<"Your Account Details"<<endl;

cout<<acc;

break;

case 3:

cout<<"Enter Account Number:";

cin>>account_no;

cout<<"Enter Balance:";

cin>>amnts;

acc=b.Deposit(account_no, amnts);

cout<<endl<<"Amount is Deposited"<<endl;

cout<<acc;

break;

case 4:

cout<<"Enter Account Number:";

cin>>account_no;

cout<<"Enter Balance:";

cin>>amnts;

acc=b.Withdraw(account_no, amnts);

cout<<endl<<"Amount Withdrawn"<<endl;

cout<<acc;

break;

case 5:

cout<<"Enter Account Number:";
```

```
cin>>account_no;

b.CloseAccount(account_no);

cout<<endl<<"Account is Closed"<<endl;

cout<<acc;

case 6:

b.ShowAllAccounts();

break;

case 7: break;

default:

cout<<"\nEnter corret choice";

exit(0);

}

}while(option!=7);

return 0;

}

Cl_Accounts::Cl_Accounts(string fname,string lname,float client_balance)

{

Nxt_Accnt_No++;

Accnt_No=Nxt_Accnt_No;

client_fname=fname;

client_lname=lname;

this->client_balance=client_balance;

}


void Cl_Accounts::Deposit(float amt)
```

```
{

client_balance+=amt;

}

void Cl_Accounts::Withdraw(float amt)

{

if(client_balance-amt<MIN_BALANCE)

throw deficient_funds();

client_balance-=amt;

}

void Cl_Accounts::setLstAccntNo(long Accnt_No)

{

Nxt_Accnt_No=Accnt_No;

}

long Cl_Accounts::getLstAccntNo()

{

return Nxt_Accnt_No;

}

ofstream & operator<<(ofstream &ofs,Cl_Accounts &acc)

{

ofs<<acc.Accnt_No<<endl;

ofs<<acc.client_fname<<endl;

ofs<<acc.client_lname<<endl;

ofs<<acc.client_balance<<endl;

return ofs;

}
```

```
ifstream & operator>>(ifstream &ifs,Cl_Accounts &acc)

{

ifs>>acc.Accnt_No;

ifs>>acc.client_fname;

ifs>>acc.client_lname;

ifs>>acc.client_balance;

return ifs;

}

ostream & operator<<(ostream &os,Cl_Accounts &acc)

{

os<<"First Name:"<<acc.getFName()<<endl;

os<<"Last Name:"<<acc.getLName()<<endl;

os<<"Account Number:"<<acc.getAccNo()<<endl;

os<<"Balance:"<<acc.getBlnce()<<endl;

return os;

}


Bank::Bank()

{

Cl_Accounts acnt;

ifstream infile;

infile.open("Bank.data");

if(!infile)

{

//cout<<"Error in Opening! File Not Found!!"<<endl;
return;
}
while(!infile.eof())
{
infile>>acnt;
accounts_cl.insert(pair<long,Cl_Accounts>(acnt.getAccNo(),acnt));
}
Cl_Accounts::setLstAccntNo(acnt.getAccNo());
infile.close();
}
Cl_Accounts Bank::Cl_Open_Account(string fname,string lname,float balance)
{
ofstream outfile;
Cl_Accounts acnt(fname,lname,balance);
accounts_cl.insert(pair<long,Cl_Accounts>(acnt.getAccNo(),acnt));
outfile.open("Bank.data", ios::trunc);
map<long,Cl_Accounts>::iterator itr;
for(itr=accounts_cl.begin();itr!=accounts_cl.end();itr++)
{
outfile<<itr->second;
}
outfile.close();
return acnt;
}
Cl_Accounts Bank::Cl_Balance_Enquiry(long Accnt_No)
```

```
{
map<long,Cl_Accounts>::iterator itr=accounts_cl.find(Accnt_No);
return itr->second;
}
Cl_Accounts Bank::Deposit(long Accnt_No,float amt)
{
map<long,Cl_Accounts>::iterator itr=accounts_cl.find(Accnt_No);
itr->second.Deposit(amt);
return itr->second;
}
Cl_Accounts Bank::Withdraw(long Accnt_No,float amt)
{
map<long,Cl_Accounts>::iterator itr=accounts_cl.find(Accnt_No);
itr->second.Withdraw(amt);
return itr->second;
}

void Bank::CloseAccount(long Accnt_No)
{
map<long,Cl_Accounts>::iterator itr=accounts_cl.find(Accnt_No);
cout<<"Account Deleted"<<itr->second;
accounts_cl.erase(Accnt_No);
}

void Bank::ShowAllAccounts()
```

```
{
map<long,Cl_Accounts>::iterator itr;
for(itr=accounts_cl.begin();itr!=accounts_cl.end();itr++)
{
cout<<"Account "<<itr->first<<endl<<itr->second<<endl;
}
}

Bank::~Bank()
{
ofstream outfile;
outfile.open("Bank.data", ios::trunc);
map<long,Cl_Accounts>::iterator itr;
for(itr=accounts_cl.begin();itr!=accounts_cl.end();itr++)
{
outfile<<itr->second;
}
outfile.close();
}
```