Bank Management System

Presented by: Izwa Afzal, Mahrukh
Submitted to: Ma'am Sobia Khalid

Program

```
// OOP project.cpp : Defines the entry point for the console application.
//

#include "stdafx.h"
#define _CRT_SECURE_NO_WARNINGS
#include<iostream>
using namespace std;
class Bank                                  // Base class
{
protected:
char bank_name[100];
char branch_code[100];
char location[1000];
public:
Bank()                                      // default constructor
{
strcpy_s(bank_name, "\0");
strcpy_s(branch_code, "\0");
strcpy_s(location, "\0");
}
Bank(char bname[], char bcode[], char blocation[])  // parametrized constructor
{
strcpy_s(bank_name, bname);
strcpy_s(branch_code, bcode);
strcpy_s(location, blocation);
}
void get_information()                  // Information getting function
{
cout << " Enter the name of the Bank :: ";
cin >> bank_name;
cout << " Enter the Branch code :: ";
cin >> branch_code;
cout << " Enter the location of the Bank :: ";
cin >> location;
}
void display()                          // Display function
{
cout << " ****  NAME OF BANK  **** " << endl;
cout << "    " << bank_name << "    " << endl;
cout << " ****  BRANCH CODE OF THE BANK  **** " << endl;
cout << "    " << branch_code << "    " << endl;
cout << " ****  NAME OF BANK  **** " << endl;
cout << "    " << location << "    " << endl;
}
};
class bank_employee : public Bank               // derive class inherited from Bank
{
private:
int ID;         // id number
protected:
char name[100];
char address[100];
long contact;
int salary;
```

```
public:
bank_employee() :Bank(), ID(0), contact(0), salary(0)   // Default constructor
{
strcpy_s(name, "\0");
strcpy_s(address, "\0");
}
// parametrized constructor
bank_employee(char bname[], char bcode[], char blocation[], int empID, char empname[], char empaddress[], long c, int
s) :Bank(bname, bcode, blocation), ID(empID), contact(c), salary(s)
{
strcpy_s(name, empname);
strcpy_s(address, empaddress);
}
void get_information()
{
cout << " Enter the Name of the Employee :: ";
cin >> name;
cout << " Enter the ID Number of the Employee :: ";
cin >> ID;
cout << " Enter the Address of the Employee :: ";
cin >> address;
cout << " Enter the Contact of the Employee :: ";
cin >> contact;
}

void calculate_salary()    // function for calculating salary of employees
{
int hours;
int extra_hours;
int salary_per_day;
int bonus;
for (int i = 0; i < 5; i++)
{
bank_employee::get_information();
cout << " Enter the working hours in a day :: ";
cin >> hours;
cout << endl;
cout << " Enter the Salary_Per_Day :: ";
cin >> salary_per_day;
cout << endl;
cout << " Enter the Extra_hours of working in a month ";
cin >> extra_hours;
if (extra_hours == 0)
{
bonus = 0;
salary = (hours * salary_per_day) + bonus;
cout << " **** The Calculated Salary is :: ";
cout << salary << endl;
}
else if ((extra_hours > 0) && (extra_hours <= 10))
{
bonus = 5000;
salary = (hours * salary_per_day) + bonus;
cout << "**** The Calculated Salary is :: ";
cout << salary << endl;
}
else if ((extra_hours > 10) && (extra_hours <= 15))
{
bonus = 7000;
```

```
salary = (hours * salary_per_day) + bonus;
cout << "**** The Calculated Salary is :: ";
cout << salary << endl;
}
else if ((extra_hours > 16) && (extra_hours <= 20))
{
bonus = 10000;
salary = (hours * salary_per_day) + bonus;
cout << "**** The Calculated Salary is :: ";
cout << salary << endl;
}
else if (extra_hours > 21)
{
cout << " GOOD JOB " << endl;
cout << " The bank will go through your progress report and will list your name in promotion file. " << endl;

}
else
{
cout << "  INVALID ENTRY  " << endl << " You have entered an invalid character please try again " << endl;
cout << " Regards " << endl;
}
}


}
void display()
{
cout << " ****  RECORD OF EMPLOYEE  **** " << endl;
cout << " ****  NAME  **** " << endl;
cout << "    " << name << "    " << endl;
cout << " ****  ID NUMBER  **** " << endl;
cout << "    " << ID << "    " << endl;
cout << " ****  ADDRESS  **** " << endl;
cout << "    " << address << "    " << endl;
cout << " ****  CONTACT NUMBER  **** " << endl;
cout << "    " << contact << "    " << endl;
cout << " ****  SALARY  **** " << endl;
cout << "    " << salary << "    " << endl;
cout << endl;
cout << endl;
cout << endl;
}
};
class ATMs : public Bank   // Derive class inherited from base class(Bank)
{
protected:
long pin;   // pin number of the customer
char operator_name[100];
long withdraw_money;  // Money to be with drawn
long balance;
public:
ATMs() :Bank(), pin(0), withdraw_money(0),balance(0)  // Default constructor
{

strcpy_s(operator_name, "\0");

}
// parametrized Constructor
```

```
ATMs(char bname[], char bcode[], char blocation[], long p, char opename[], long money, long b) :Bank(bname, bcode,
blocation), pin(p), withdraw_money(money), balance(b)
{
strcpy_s(operator_name, opename);
}

void pin_verification()        // function for validity of PIN number
{
cout << endl;
cout << endl;
cout << " Please enter your pin number :: " << endl;
cin >> pin;
cout << endl;
if ((pin == 5155) || (pin == 6165) || (pin == 7175) || (pin == 8185) || (pin == 9195))
{
cout << " PIN Number is verified " << endl;
}
else
{
cout << " You had entered invalid PIN Number " << endl;
cout << " Please try again and enter the correct PIN Number " << endl;
}
}
void transaction_details()    // function for withdrawing money
{
cout << " Please enter your name ::";
cin >> operator_name;
cout << endl;
cout << " Balance of your account ::";
cin >> balance;
cout << " Please enter the amount of money you want to withdraw ::";
cin >> withdraw_money;
if (withdraw_money < balance)
{
balance = balance - withdraw_money;
cout << " Now your remaining balance is ::" << balance << endl;
}
else
{
cout << " Your entered amount is greater than your balance " << endl;
cout << " Please try again " << endl;
}

}

void change_pin()
{
int choice;
cout << " Do you want to chnage your PIN Number " << endl;
cout << " If yes than Enter 1 otherwise enter 0 ::";
cin >> choice;
if (choice == 1)
{
cout << " Enter the new PIN Number ::";
cin >> pin;
cout << endl;
}
else if (choice == 0)
{
```

```
}
else
{
cout << " You had entered some wrong number " << endl << " Please try again " << endl;
}
cout << endl;
cout << endl;
cout << endl;
}
void end_session()
{
char date_of_issue[100];
char expiry_date[100];
int validity_years = 5;
cout << " Enter the year of issuing card :: ";
cin >> date_of_issue;
cout << endl;
cout << " Enter the expiry year of ATM card :: ";
cin >> expiry_date;
cout << endl;
if ((date_of_issue - expiry_date) <= validity_years)
{
cout << " Your card is valid you can withdraw money " << endl;
}
else
{
cout << " Your card had expired, you can withdraw money " << endl;
}
}
};
class contact : public Bank
{
public:
void help()
{
cout << endl;
cout << endl;
int choice = 0;
cout << " Well Come to our helpline " << endl;
cout << "Enter 1 for helpline of Rawalpindi branch " << endl;
cout << "Enter 2 for helpline of Karachi branch " << endl;
cout << "Enter 3 for helpline of Peshawar branch " << endl;
cout << "Enter 4 for helpline of Quetta branch " << endl;
cout << "Enter 5 for helpline of Islamabad branch " << endl;
cin >> choice;
if (choice == 1)
{
cout << " Contact Number :: +92 335 7816543 " << endl;
cout << " Gmail :: helpline.pk@gmail.com " << endl;
cout << " Opening Time Of Bank :: 9:00 AM " << endl;
cout << " Break Time Of Bank :: 11:30AM - 1:00PM " << endl;
cout << " Closing Time Of Bank :: 8:00 PM " << endl;
}
else if (choice == 2)
{
cout << " Contact Number :: +92 335 7817853 " << endl;
cout << " Gmail :: helpline.pk@gmail.com " << endl;
cout << " Opening Time Of Bank :: 9:00 AM " << endl;
cout << " Break Time Of Bank :: 11:00AM - 2:00PM " << endl;
```

```
cout << " Closing Time Of Bank :: 7:00 PM " << endl;
}
else if (choice == 3)
{
cout << " Contact Number :: +92 335 4562870 " << endl;
cout << " Gmail :: helpline.pk@gmail.com " << endl;
cout << " Opening Time Of Bank :: 9:00 AM " << endl;
cout << " Break Time Of Bank :: 11:00AM - 1:30PM " << endl;
cout << " Closing Time Of Bank :: 9:00 PM " << endl;
}
else if (choice == 4)
{
cout << " Contact Number :: +92 335 4567321 " << endl;
cout << " Gmail :: helpline.pk@gmail.com " << endl;
cout << " Opening Time Of Bank :: 8:00 AM " << endl;
cout << " Break Time Of Bank :: 11:00AM - 2:00PM " << endl;
cout << " Closing Time Of Bank :: 9:00 PM " << endl;
}
else if (choice == 5)
{
cout << " Contact Number :: +92 334 4567330 " << endl;
cout << " Gmail :: helpline.pk@gmail.com " << endl;
cout << " Opening Time Of Bank :: 9:00 AM " << endl;
cout << " Break Time Of Bank :: 11:30AM - 1:30PM " << endl;
cout << " Closing Time Of Bank :: 10:00 PM " << endl;
}
else
{
cout << " You had entered an invalid number " << endl;
cout << " Please try again " << endl;
}

cout << endl;
cout << endl;
}
};
class Accounts : public Bank
{
protected:
long acc_numb;
float acc_balance;
char name[100];
char acc_type[100];
char branch_location[100];
public:
Accounts() :Bank() // default constructor
{
acc_numb = 0;
acc_balance = 0;
strcpy_s(name, "\0");
strcpy_s(acc_type, "\0");
strcpy_s(branch_location, "\0");
}
// parametrized function
Accounts(char bname[], char bcode[], char blocation[], char acc[], char branchloc[], char namee[], float b, long
account_num) :Bank(bname, bcode, blocation), acc_balance(b), acc_numb(account_num)
{
strcpy_s(name, namee);
strcpy_s(acc_type, acc);
```

```
strcpy_s(branch_location, branchloc);
}

// pure virtual functions
virtual void get_data() = 0;
virtual void display() = 0;
virtual void calculation() = 0;
void put_data()
{
cout << endl;
cout << endl;
Bank::get_information();
cout << " Enter your name :: ";
cin >> name;
cout << " Enter the type of account :: ";
cin >> acc_type;
cout << endl;
cout << " Enter the location of the branch :: ";
cin >> branch_location;
cout << endl;
cout << " Enter the account balance :: ";
cin >> acc_balance;
cout << endl;
}
void show_data()
{
Bank::display();
cout << " **** NAME **** " << endl;
cout << "    " << name << "    " << endl;
cout << " **** ACCOUNT BALANCE **** " << endl;
cout << "    " << acc_balance << "    " << endl;
cout << " **** ACCOUNT TYPE **** " << endl;
cout << "    " << acc_type << "    " << endl;
cout << " **** BRANCH LOCATION **** " << endl;
cout << "    " << acc_type << "    " << endl;
}

void credit()  // function for adding money in account
{
float balance = 0;
cout << "Enter the amount of balance you wanted to add in your account : ";
cin >> balance;
acc_balance = acc_balance + balance;
cout << "Now your Account Balance is : " << acc_balance;
cout << endl;
}
void debit()  // function for withdrawing money from account
{
float c;
cout << "Enter the amount of balance you wanted to Withdraw from your account : ";
cin >> c;
cout << endl;
if (c > acc_balance)
{
cout << " Your Withdraw amount is greater than the account balance " << endl;
cout << " You cannot withdraw this amount " << endl;
cout << "Please enter the amount again" << endl;
}
else
```

```
{
acc_balance = acc_balance - c;
cout << "Now your Account Balance is :: " << acc_balance;
cout << endl;
}
}
};

class statement : public Accounts
{
protected:
char time[30];
char date[30];
public:
statement() :Accounts()
{
strcpy_s(time, "\0");
strcpy_s(date, "\0");
}
statement(char bname[], char bcode[], char blocation[], char acc[], char branchloc[], char namee[], float b, long
account_num, char t[], char d[]) :Accounts(bname, bcode, blocation, acc, branchloc, namee, b, account_num)
{
strcpy_s(time, t);
strcpy_s(date, d);
}
void get_data()
{
Accounts::put_data();
cout << " Time of Transaction is :: ";
cin >> time;
cout << endl;
cout << " Date of Transaction is :: ";
cin >> date;
cout << endl;
}
void display()
{
Accounts::show_data();
cout << " Your Remaining balance is :: " << acc_balance << endl;
cout << " Time of Transaction is :: " << time << endl;
cout << " Date of Transaction is :: " << date << endl;
cout << endl;
cout << endl;
}

void calculation()
{ }

};
class current_account : public Accounts
{
protected:
int fixed_fee;
int PIN;
public:
current_account() : Accounts(), fixed_fee(0), PIN(0) // default constructor
{}
// parametrized constructor
```

```
current_account(char bname[], char bcode[], char blocation[], char acc[], char branchloc[], char namee[], float b, long
account_num, int f, int p) :Accounts(bname, bcode, blocation, acc, branchloc, namee, b, account_num), fixed_fee(f),
PIN(p)
{}
void get_data()
{
cout << " **** Current Account **** " << endl;
Accounts::put_data();
cout << endl;
cout << " Enter your Account Number :: ";
cin >> acc_numb;
cout << " Enter your PIN :: ";
cin >> PIN;
cout << endl;
cout << " Bank will enter the fixed fee amount :: ";
cin >> fixed_fee;
cout << endl;
}
void display()
{
cout << " **** Displaying the details of Current Account **** " << endl;
Accounts::show_data();
cout << endl;
cout << " **** ACCOUNT NUMBER **** " << endl;
cout << "    " << acc_numb << "    " << endl;
cout << " **** PIN **** " << endl;
cout << "    " << PIN << "    " << endl;
cout << " **** FIXED FEE AMOUNT **** " << endl;
cout << "    " << fixed_fee << "    " << endl;
}
void calculation()
{
cout << " **** Credit Function **** " << endl;
Accounts::credit();
cout << " Dedit Function " << endl;
Accounts::debit();
acc_balance = acc_balance - fixed_fee;
cout << " Now your current balance after deducting fixed_fee is :: " << acc_balance << endl;
cout << endl;
cout << endl;
cout << endl;
}
};


class saving_account : public Accounts
{
protected:
int pin;
public:
saving_account() :Accounts(), pin(0) // Default constructor
{}
// parametrized constructor
saving_account(char bname[], char bcode[], char blocation[], char acc[], char branchloc[], char namee[], float b, long
account_num, int p) :Accounts(bname, bcode, blocation, acc, branchloc, namee, b, account_num), pin(0)
{}
void get_data()
{
cout << " **** Saving Account **** " << endl;
```

```
Accounts::put_data();
cout << endl;
cout << " Enter your PIN :: ";
cin >> pin;
cout << endl;
}
void display()
{
cout << " **** Displaying the details of Saving Account **** " << endl;
Accounts::show_data();
cout << " **** PIN NUMBER **** " << endl;
cout << pin;
cout << endl;
}
void calculation()
{
cout << " **** CREDIT FUNCTION **** " << endl;
Accounts::credit();
cout << " **** DEBIT FUNCTION **** " << endl;
Accounts::debit();
cout << " Your minimum balance should be 500 in your account. " << endl;
cout << " Deduction of interest every year " << endl;
if (acc_balance >= 500)
{
acc_balance = acc_balance-(acc_balance* 0.025);
cout << " Now your current balance is :: " << acc_balance;
cout << endl;
}
else
{
cout << " Your account should contain a minimum balance of 500. " << endl;
cout << " Add the minimum balance in the account or your account will be blocked " << endl;
}
cout << endl;
cout << endl;
cout << endl;
}
};
class loan_account : public Accounts
{
protected:
long loan_numb;
char loan_type[100];
public:
loan_account() :Accounts(), loan_numb(0)
{
strcpy_s(loan_type, "\0");
}
loan_account(char bname[], char bcode[], char blocation[], char acc[], char branchloc[], char namee[], float b, long
account_num, long lnumb, char ltype[]) :Accounts(bname, bcode, blocation, acc, branchloc, namee, b, account_num),
loan_numb(lnumb)
{
strcpy_s(loan_type, ltype);
}
void get_data()
{
cout << "** Loan Account **" << endl;
Accounts::put_data();
cout << " Enter the loan number issued to you ::";
```

```
cin >> loan_numb;
cout << " Enter the loan type ::";
cin >> loan_type;
}
void display()
{
cout << " **** Displaying the details of Loan Account ****" << endl;
Accounts::show_data();
cout << " **** LOAN NUMBER **** " << endl;
cout << "    " << loan_numb << "    " << endl;
cout << " **** LOAN_TYPE ****" << endl;
cout << "    " << loan_type << "    " << endl;
cout << " Your application will be considered by the Bank " << endl;
cout << " You will be informed later whether you can avail loan or not " << endl;
cout << " Thanks " << endl;
cout << endl;
cout << endl;
}
void calculation()
{
}

};


class online_transaction :public Accounts
{
private:
int cust_pin;   // pin number
long CNIC_numb;
long card_numb;
char password[100];
public:
online_transaction() :Accounts(), cust_pin(0), CNIC_numb(0), card_numb(0)
{
strcpy_s(password, "\0");
}
online_transaction(char bname[], char bcode[], char blocation[], char acc[], char branchloc[], char namee[], float b, long
account_num, int p, long cnic, long cardnumb, char passwordd[]) :Accounts(), cust_pin(p), CNIC_numb(cnic),
card_numb(cardnumb)
{
strcpy_s(password, passwordd);
}

void calculation()
{
cout << " **** ONLINE TRANSACTION **** ";
Accounts::debit();
cout << endl;
cout << endl;
cout << endl;
}
void get_data()
{
cout << " **** Online Transaction **** " << endl;
Accounts::put_data();
cout << " Enter the PIN Number ::";
cin >> cust_pin;
cout << " Enter CNIC Number ::";
```

```
cin >> CNIC_numb;
cout << " Enter the CARD Number ::";
cin >> card_numb;
cout << " Enter password ::";
cin >> password;
}

void display()
{
cout << " **** Displaying the details of Online transaction **** " << endl;
Accounts::show_data();
cout << " **** PIN NUMBER **** " << endl;
cout << "    " << cust_pin << "    " << endl;
cout << " **** CNIC NUMBER **** " << endl;
cout << "    " << CNIC_numb << "    " << endl;
cout << " **** CARD NUMBER **** " << endl;
cout << "    " << cust_pin << "    " << endl;
cout << " **** REMAINING BALANCE **** " << endl;
cout << "    " << acc_balance << "    " << endl;
}
};


int _tmain(int argc, _TCHAR* argv[])
{cout << endl;
cout << "                ||:)  WELLCOME TO BANK MANAGEMENT SYSTEM  ||:)                  ";
cout << endl;
int choice;
cout << "Enter 1 for the Bank Employee Information " << endl;
cout << "Enter 2 for the ATM Machine  " << endl;
cout << "Enter 3 for the Helpline " << endl;
cout << "Enter 4 for the Accounts " << endl;
cin >> choice;
if (choice == 1)
{
bank_employee obj1;
obj1.calculate_salary();
obj1.display();
}
else if (choice == 2)
{
ATMs obj2;
obj2.end_session();
obj2.pin_verification();
obj2.transaction_details();
obj2.change_pin();
}
else if (choice == 3)
{
contact obj3;
obj3.help();
}
else if (choice == 4)
{
Accounts* p[10];
int choice;
for (int i = 0; i < 10; i++)
{
cout << "Enter 1 for the Saving account " << endl;
```

```
cout << "ENter 2 for the Current account " << endl;
cout << "Enter 3 for the Loan account " << endl;
cout << "ENter 4 for the Online Transaction " << endl;
cout << "Enter 5 for the Statement Print " << endl;
cin >> choice;
if (choice == 1)
{
p[1] = new saving_account;
p[1]->get_data();
p[1]->display();
p[1]->calculation();
}
else if (choice == 2)
{
p[2] = new current_account;
p[2]->get_data();
p[2]->display();
p[2]->calculation();
}
else if (choice == 3)
{
p[3] = new loan_account;
p[3]->get_data();
p[3]->display();
p[3]->calculation();
}
else if (choice == 4)
{
p[4] = new online_transaction;
p[4]->get_data();
p[4]->display();
p[4]->calculation();
}
else if (choice == 5)
{
p[5] = new statement;
p[5]->get_data();
p[5]->display();
}
else
{
cout << " You had enter wrong number " << endl;
cout << "Please try again " << endl;
}
}

}


else
{
cout << " You had enter wrong number " << endl;
cout << "Please try again " << endl;
}

system("pause");

        return 0;
}
```

Output:

```
                ||:)  WELLCOME TO BANK MANAGEMENT SYSTEM  ||:)
Enter 1 for the Bank Employee Information
Enter 2 for the ATM Machine
Enter 3 for the Helpline
Enter 4 for the Accounts
4
Enter 1 for the Saving account
ENter 2 for the Current account
Enter 3 for the Loan account
ENter 4 for the Online Transaction
Enter 5 for the Statement Print
2
 **** Current Account ****


 Enter the name of the Bank :: aLFALA
 Enter the Branch code :: 12345
 Enter the location of the Bank :: rAWALPINDI
 Enter your name :: IZWA
 Enter the type of account :: CURRENT

 Enter the location of the branch :: DHA

 Enter the account balance :: 200000


 Enter your Account Number :: 56
 Enter your PIN :: 12344

 Bank will enter the fixed fee amount :: 34

 **** Displaying the details of Current Account ****
 ****  NAME OF BANK  ****
    aLFALA
 ****  BRANCH CODE OF THE BANK  ****
    12345
 ****  NAME OF BANK  ****
    rAWALPINDI
 **** NAME ****
    IZWA
 **** ACCOUNT BALANCE ****
    200000
 **** ACCOUNT TYPE ****
```

```
   IZWA
**** ACCOUNT BALANCE ****
   200000
**** ACCOUNT TYPE ****
   CURRENT
**** BRANCH LOCATION ****
   CURRENT

**** ACCOUNT NUMBER ****
   56
**** PIN ****
   12344
**** FIXED FEE AMOUNT ****
   34
**** Credit Function ****
Enter the amount of balance you wanted to add in your account : 46778
Now your Account Balance is : 246778
 Dedit Function
Enter the amount of balance you wanted to Withdraw from your account : 12356

Now your Account Balance is :: 234422
 Now your current balance after deducting fixed_fee is :: 234388



Enter 1 for the Saving account
ENter 2 for the Current account
Enter 3 for the Loan account
ENter 4 for the Online Transaction
Enter 5 for the Statement Print
```

REPORT

Our project is about **"Bank Management System"** using this software user can credit, debit, store information, check balance, online transaction ,help centre, users have saving account, loan account, and it can print the statement of the bank, user can perform operation through ATM machine anywhere in the world.

Procedure:

Base Class:

First of all we have created a base class named as bank, in which bank name, bank location and branch code is entered by the user, and then we have created default and parametrized constructors, and we created two functions that can get data by the user and then display it.

Derived Class:

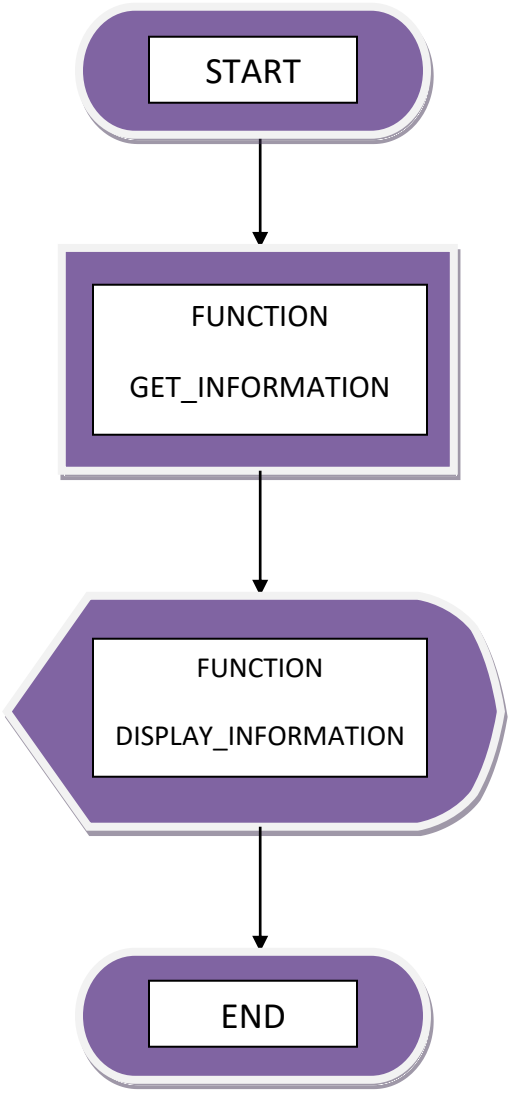
We have created a derived class named as bank employee which has private data member ID and protected data member name, address, contact number and employee salary, then we have created default and parametrized constructor. They have created a function to get data from user and then we have created another function to calculate the salary if a user worked some extra hours and he must get bonus of his/her extra time and if extra hours exceed from limit then he is consider in promotion list and the we created display function to display all the data.

We have also created another class named as ATM from bank class which has data member pin code, name of operator, withdraw money amount and balance of the account. Then we created default and parametrized constructors. We have hard code come pin if user enter a pin which does not match with these pins then compiler will display message of incorrect pin. We have created a function for transaction if user will debit amount, amount will be debited or if it exceed from actual amount, It will display message Your entered amount is greater than your balance, and if user wants to change the pin code he/she can also do that. We have created another function that ATM card has a limit for 5 years, after 5 years the ATM card will expire.
Then we created another derive class helpline for the user if user wants to some help contact number of bank, opening, closing and break time of bank he/she can easily get the information, and also the email address also he/she can leave a message and bank will reply soon.
We have created another derived class named as Accounts, which has data members account number, account balance, account type,branch location and name of user. Then we have created default and parametrized constructors. Then we have created three pure virtual functions in it to get data of user, display the data and perform calculation. We have created two function to perform debit and credit operation.
We have created another derived class statement, derived from accounts class, which prints the statement of bank time and date of transaction and credit.
We have created another derived class current account, class has data member fixed fee and PIN, then we have created default and parametrized constructor. Then we call pure virtual function of

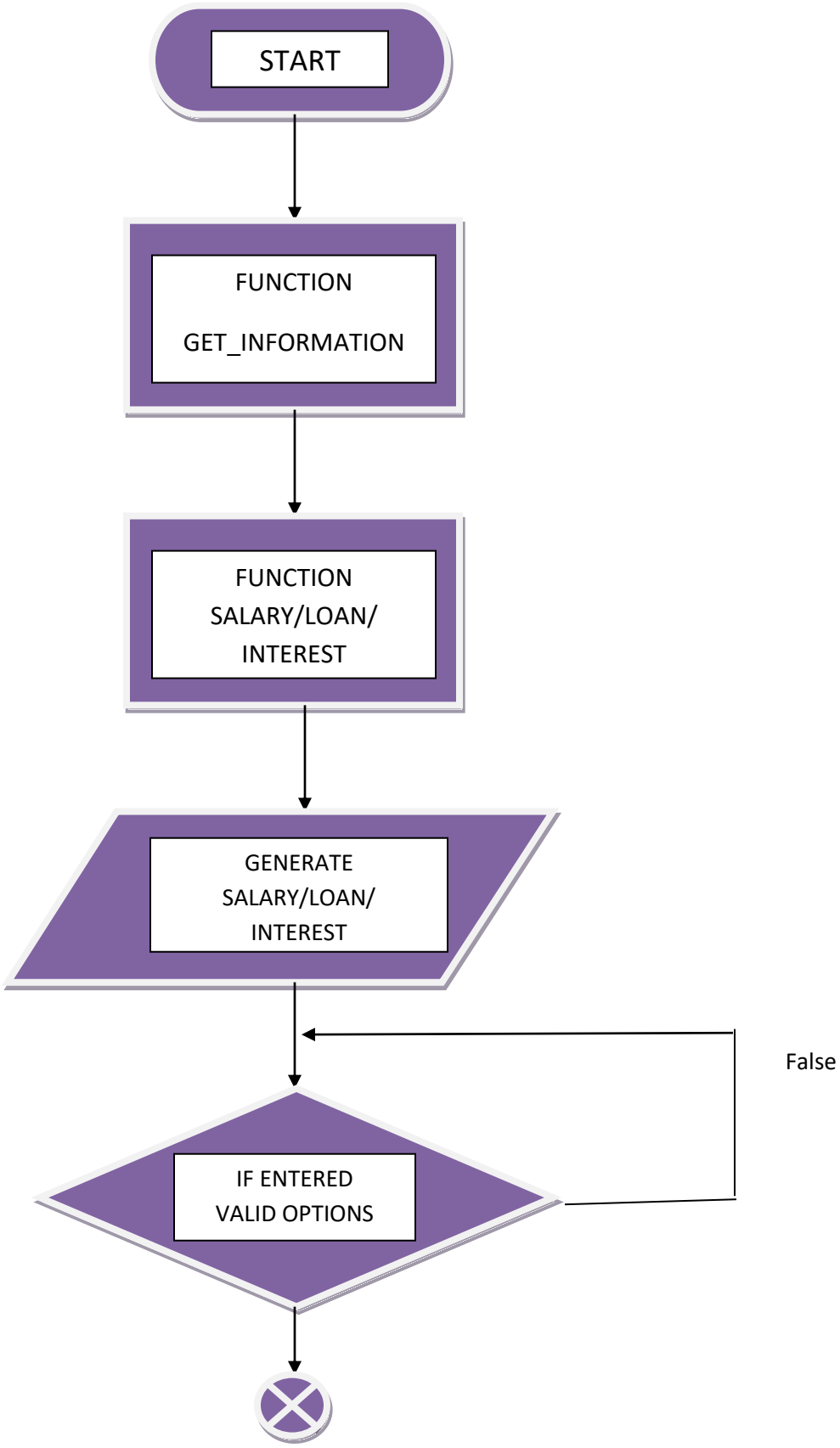
base class to get data and perform calculation. When a user debit some amount, fixed fee is subtracted from account balance.
We have created another derived class saving account, derived from accounts that has data member PIN. And we get the data from user. The perform calculation that user must have 500 rupees in his/her account and it will display data.
We have created another derive class loan account, has data member loan number and loan type. It will get the data by user and the application will be submitted to bank. And it will display the data.
We have created another derived class online transaction, has data member customer PIN, CNIC, card number, password of account. We have created default and parametrized constructor. And the data will be input by the user. Password is hard code by the bank. If password does not match it will give error and display the data.
In main() we have choice what user want to do. Enter 1 for the Bank Employee Information, Enter 2 for the ATM Machine, Enter 3 for the Helpline, Enter 4 for the Accounts. User will select option by himself.

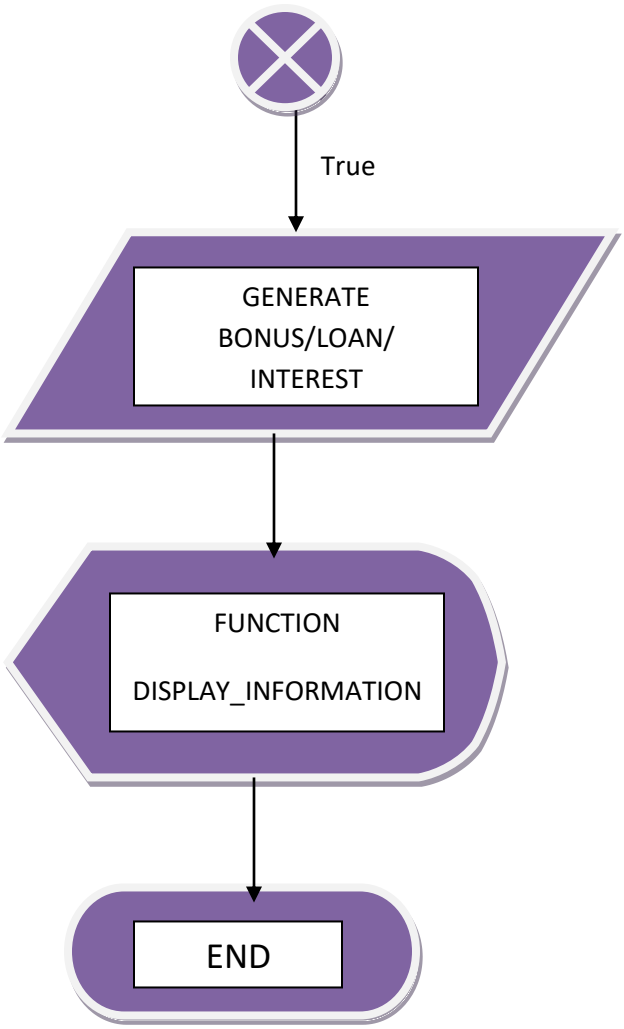
Flow chart

BASE CLASS:

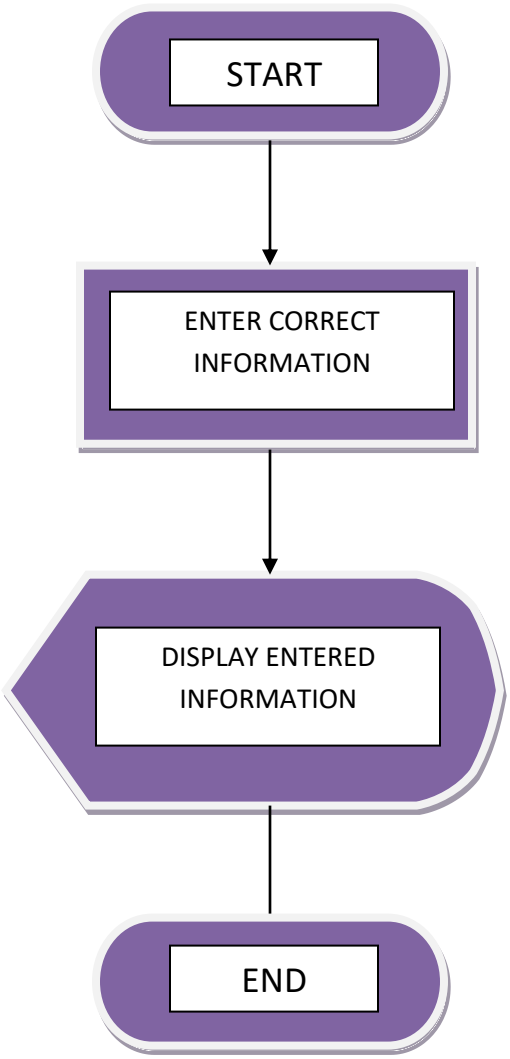


DERIVE CLASS


START
FUNCTION
GET_INFORMATION
FUNCTION
SALARY/LOAN/
INTEREST
GENERATE
SALARY/LOAN/
INTEREST
IF ENTERED
VALID OPTIONS
False


True
GENERATE
BONUS/LOAN/
INTEREST
FUNCTION
DISPLAY_INFORMATION
END

MAIN:


START
ENTER CORRECT INFORMATION
DISPLAY ENTERED INFORMATION
END