1 MÔ TẢ HỆ THỐNG MÃ NGUỒN

1.1 Bài 1: Tài khoản ngân hàng

1. **Đầu tiên ta thực hiện việc tạo tài khoản ngân hàng với Class TaiKhoan sẽ gồm có:**

```
class TaiKhoan
{
protected:
	string name;		// tên tài khoản
	string cmnd;		// số chứng minh nhân dân hoặc thẻ căn cước
	string phone;		// số điện thoại
	string email;		// địa chỉ email
	string ngaySinh;	// ngày tháng năm sinh

	string account;
	double soDu;		// số tiền trong tài khoản

public:

	virtual void Nhap(); // thực hiện nhập xuất
	virtual void Xuat();
};
```

Đồng thời xuất ra các điều khoản ngân hàng cho người dùng.

2. **Tiếp theo sẽ có 4 hình thức thẻ mà người dùng sẽ chọn: Tài khoản thanh toán, Tài khoản thẻ tín dụng, Tài khoản thẻ tích điểm và Tài khoản thẻ hoàn tiền**. Ứng với tài khoản thanh toán sẽ kế thừa class TaiKhoan trên:

```
class PaymentAccount : public TaiKhoan
{
private:
	double balance; // số dư trong tài khoản
	string ngayPhatHanh; // ngày phát hành thẻ
	vector<PaymentHistory> historyList; // lịch sử các cuộc giao dịch được lưu với dạng vector
public:
	void Nhap();

	void setAccount(string);
```

```
        void setAmount(double);

        void NapTien();
        double getBalance();
        bool transferTo(double amount);
        void showHistory();
};
```

- void Nhap() sẽ nhập số tài khoản, ngày phát hành (ngày tạo thẻ)
- void NapTien() thực hiện việc nạp tiền vào tài khoản thanh toán của bạn, ứng với từng mệnh giá
- double getBalance() trả về số tiền hiện có trong tài khoản
- bool transferTo(double amount) thực hiện việc chuyển tiền, yêu cầu nhập tài khoản thụ hưởng, ngày giao dịch và nội dung chuyển khoản

```
void PaymentAccount::showHistory()
{
        cout << "-- Lich su thanh toan -- ";
        if (historyList.size() == 0)
                cout << " Ban chua thuc hien thanh toan nao!";
        else {
                for (auto& i : historyList)
                        i.Xuat();
        }
}
```

- Xuất ra thông tin các cuộc giao dịch thông qua class PaymentHitory.

3. Tài khoản thẻ tín dụng cũng sẽ được kế thừa từ lớp cha là class TaiKhoan

- double getBalance() thực hiện việc trả về số dư hiện có trong tài khoản
- bool charge(double amount) nếu thanh toán được khi amount + balance <= creditLimit thì thanh toán thành công và ngược lại.

```
void CreditCard::payment(double amount)
{
        cout << " Lua chon thanh toan:";
        cout << " 1. Thanh toan het.";
        cout << " 2. Thanh toan so tien toi thieu.";
        cout << " Neu khong thanh toan se phai chiu phat !";

        if (choice == 1) {
                balance -= amount;
                cout << " Thanh toan thanh cong !";
        }
        else if (choice == 2) {
                minPayment = 0.05 * amount;
                balance -= minPayment;
                cout << " Thanh toan thanh cong !";
        }
        else {
                cout << " Ban chap nhan chiu phat 2000000 VND + lai so tien trong ki sao ke nay !"
                balance -= latePenalty;
```

```
                balance -= interestRate;
        }
}
```

- Hàm bool releaseCard() để thực hiện việc phát hành thẻ với 1 thẻ chính và tối đa 5 thẻ phụ, nếu quá số lượng thì trả về false. Những thẻ này sẽ được quản lí bởi 1 class riêng là class Card.

```
bool CreditCard::releaseCard()
{
        if (CardList.size() < 6)
        {
                long number = 0;
                string day;
                cout << "\n\t--  Nhap --" << endl;
                if (CardList.size() == 0) { // thẻ chính
                        cout << " So the chinh: "; cin >> number;
                }
                else { // thẻ phụ
                        cout << " So the phu: "; cin >> number;
                }
                return true;
        }
        else
        {
                cout << " Qua so luong the!" << endl;
                return false;
        }
}
```

4. Tài khoản thẻ tích điểm sẽ kế thừa lớp cha là class CreditCard

```
class RewardCard : public CreditCard
{
        double rewardRate;
        double currentPoints;
public:
        RewardCard();
        ~RewardCard();
        double getCurrentPoints();
        bool charge(double amount);
        bool payWithPoints(int pAmount);

        void Nhap();
};
```

- double getCurrentPoints() để trả về số tiền hiện có trong tài khoản
- bool payWithPoints(int pAmount) trả tiền ứng với số điểm hiện có trong tài khoản

```
bool RewardCard::payWithPoints(int pAmount)
{

	if (pAmount > currentPoints)
		return false; // không trả được vì không đủ điểm
	else {
		currentPoints -= pAmount;
		return true; // trả thành công
	}
}
```

5. Tài khoản thẻ hoàn tiền sẽ kế thừa lớp cha là class CreditCard

```
class CashbackCard : public CreditCard
{
	// cho phép người dùng nhận được số tiền hoàn lại trên mỗi giao dịch
	double cashBackRate;
	double currentCashBack;
public:
	double getCurrentCashBack();
	bool charge(double amount);
	void redeemCashBack();
};
```

- `double cashBackRate` là một số thực biểu thị tỉ lệ hoàn tiền.
- `double currentCashBack` là số tiền có thể hoàn lại hiện tại trong tài khoản.
- `double getCurrentCashBack()` để trả về số tiền hiện có trong tài khoản

```
void CashbackCard::redeemCashBack()
{
	balance -= currentCashBack;
	currentCashBack = 0;
	cout << "=> Hoan tien thanh cong!" << endl;
}
```

6. Lớp PaymentHistory để lưu lại lịch sử các cuộc giao dịch

```
class PaymentHistory
{
	string account_ht;
	string day;
	string nd;
	double money;
public:
```

```
        PaymentHistory();
        PaymentHistory(string acc, string _day, string _nd, double _money);
        PaymentHistory(string acc, double _money);
        ~PaymentHistory();
        void Nhap();
        virtual void Xuat();
};
```

- Sẽ chứa thông tin: Số tài khoản người hưởng thụ, ngày giao dịch, nội dung giao dịch và số tiền giao dịch.

7. Lớp Card thực hiện việc phát hành thẻ, mỗi thẻ có các thông tin về số thẻ, ngày phát hành và tài khoản thẻ liên kết đến

```
class Card
{
        long number; // số thẻ
        string day; // ngày phát hành
public:
        Card();
        ~Card();
        Card(long _number, string _day);
};
```

```
Card::Card(long _number, string _day)
{
        number = _number;
        day = _day;
}
```

Kết thúc