<COMPUTER PROGRAMMING 2023 Project 1>

DUE DATE : 05/16 (Tue) 23:59

이 과제는 실생활에서 사용할 수 있는 예제를 바탕으로 OOP 프로그래밍을 연습하는 것이 목적입니다. 은행에서의 작업(송금, 입금, 인출, 대출)을 제어하는 3 가지 유형의 은행 계좌로 은행 시스템을 구현하는 과제입니다.

- cf) 은행이자 및 대출수수료는 작성할 때 기간등은 고려하지 않으며 이자 및 대출 수수료는 1 회성으로 발생한다 가정하여 작성하시면됩니다.

class BankAccount

이것은 모든 유형의 은행 계좌에 대한 추상 기본 클래스입니다.

MEMBER DATA(protected 로 선언)

1. `int acctnum;`
   - 계좌번호
2. `float bal;`
   - 현재 계좌의 잔액
3. `String bank_namel;`
   - 은행이름 :난수 생성(rand())을 통해 %3 을 하여 나머지가 0 이면 하나은행, 1 이면 우리은행, 2 이면 신한은행(거래시 같은 은행이면 수수료 0, 아니면 5)
4. `int creditl;`
   - 신용도
   - 현재 계좌잔액(bal)이 1000 미만이면 신용도 1, 대출 불가
   - 현재 계좌잔액(bal)이 1000 이상 2000 미만이면 신용도 2
   - 현재 계좌잔액(bal)이 2000 이상이면 신용도 3

MEMBER FUNCTIONS

1. `BankAccount(int num,float bal);`
   - 매개변수를 num, bal 을 사용하여 계좌번호(acctnum)와 계좌잔액(bal) 멤버 변수에 값을 할당하는 생성자를 만듭니다.
2. `void deposit(float amount);`
   - 현재 계좌잔액(bal)을 금액(amount)만큼 증가
3. `virtual int withdraw(float amount);`
   - amount 만큼 현재 계좌잔액을 감소

- 0 값을 return 함 => return 0;

4. `int getAcctnum();`
   - acctnum 변수에 대한 getter 함수

5. `float getBalance();`
   - bal 변수에 대한 getter 함수

6. `int getCredit(); // getter function`
   - credit 변수에 대한 getter 함수

7. `string getBankname(); // getter function`
   - Bankname 변수에 대한 getter 함수

8. `virtual void print() = 0;`
   - pure virtual print function. Define 할 필요 없습니다

9. `void loan(float amount);`
   - 현재 계좌잔액(bal)을 금액(amount)만큼 대출받아 증가(Banksystem.cpp 에 void loan(BankAccount *b, float amount)에 대한 함수와 연동, main.cpp 에는 없습니다. 다른 main.cpp 에서 실행할 예정)
   - 대출의 상환개념은 다루지 않습니다
   - 현재 계좌잔액(bal)이 1000 미만이면 신용도 1, 대출 불가
     - 아래와 같이 출력
       - The amount cannot be loaned
   - 계좌 잔액이 1000 이상 2000 미만이면 신용도 2: 대출시 빌려주는 금액의 10%를 대출수수료 받음
     - 신용도 2 의 고객 대출금액 한도: 100~500, 대출수수료 10%
       - 대출 한도외의 금액이 amount 에 설정되면 아래와 같이 출력
         - The amount cannot be loaned
       - 예시) 100(amount)를 빌리면 100*0.1(대출수수료 10)으로 10 제외하고 bank account 에 90 만큼 계좌잔액 증가
   - 계좌 잔액이 2000 이상이면 신용도 3: 대출시 빌려주는 금액의 5%를 대출수수료로 받음
     - 신용도 3 의 고객 대출금액 한도:100~1000, 대출 대출수수료 5%
       - 대출 한도외의 금액이 amount 에 설정되면 아래와 같이 출력
         - The amount cannot be loaned
       - 예시) 100(amount)를 빌리면 100*0.05(대출수수료 5)으로 5 제외하고 bank account 에 95 만큼 계좌잔액 증가

class Savings

이 클래스는 BankAccount 에서 상속됩니다. 저축 계좌(Saving account)를 뜻하며 이자(interest)를 받을 수 있습니다.

MEMBER DATA

1. float intrate;

- 저축예금 이자율(interest rate) (**연 이자율**)

MEMBER FUNCTIONS

1. Savings(int num=0,float bal=0,float rate=3.5);

- 매개변수 num, bal, 그리고 rate 를 Acctnum, bal, 그리고 intrate 에 할당하는 생성자를 생성합니다.

2. void interest();

- 현재 잔액(bal)과 interest rate 을 기준으로 월(연간 이율이 아님) 이자를 계산하고 이 값을 현재 잔액에 더합니다.
- Ex) 현재 잔액(bal)이 1200 이고 이자율(intrate)이 10%이면 그 달의 이자는 10 입니다. 이를 1200 에 더합니다.
    - Cf) 월 이자 : intrate/100./12.

3. int withdraw(float amount);

- 현재 잔액이 해당 금액보다 작거나 같은 경우 출금을 실행하지 마십시오.
    - 잔액이 해당 금액(amount)보다 작거나 같은 경우 아래와 같이 메시지를 출력하십시오.
    - Cannot withdraw $amount 값 on account #acctnum 값 because the balance is low.
    - 이후 return 0
- 그렇지 않으면 금액(amount)만큼 잔액(bal)을 줄입니다. 이후 return 1

4. virtual void print();

- 아래와 같이 출력해야 합니다(출력화면에는 bank name 과 credit 이 옆으로 붙어있는데 아래와 같이 출력해주셔도 되고 출력화면처럼 출력해주셔도 됩니다).
- Savings Account: (account number)
  Bankname(bankname),
  Credit(credit)
  Balance: (current balance)
  Interest: (interest rate)

class Checking

이 클래스는 BankAccount 에서 상속됩니다. checking account 계좌를 뜻합니다

MEMBER DATA

1.float minimum;

- 최소 유지해야할 계좌 잔액(잔고)

2. float charge;

- 출금시에 minimum 이 유지되지 않았을 때 내는 벌금

MEMBER FUNCTIONS

1. Checking(int num=0,float bal=0,float min=1000,float chg=2);

- 4 개의 매개변수를 받아 변수 acctnum, bal, minimum 및 charge 에 할당하는 생성자를 생성합니다.

2. int withdraw(float amount);

- 현재 잔액이 해당 금액보다 작거나 같은 경우 출금을 실행하지 마십시오.
  - 대신 아래와 같은 메시지를 출력하십시오
  - Cannot withdraw $(amount) on account #(acctnum) because the balance is low.
  - Then return 0.
- 그렇지 않으면 현재 잔액(bal)이 최소 금액(min)미만인지 먼저 확인하세요.
  - Yes 일 경우 (amount + charge)값만큼 잔액을 줄입니다(잔액을 최소 금액 이상으로 유지하지 않은 경우 요금을 지불함).
  - minimum 미만이 아닌 경우, 해당 amount 만큼만 감소
  - Then return 1.

3. virtual void print();

- 아래와 같이 출력해야합니다(출력화면에는 bank name 과 credit 이 옆으로 붙어있는데 아래와 같이 출력해주셔도 되고 출력화면처럼 출력해주셔도 됩니다).
- Checking Account: (account number)
  Bankname(bankname)
  Credit(credit)
  Balance: (current balance)
  Minimum to Avoid Charges: (minimum balance)
  Charge per Check: (charge amount)

class InterestChecking

이 클래스는 Checking 에서 상속됩니다. 이자(interest)가 이 있는 checking account 계좌를 나타냅니다.

MEMBER DATA

```
1. float intrate;
```

- checking account 계좌의 이자율(연이율)

```
2. float minint;
```

- 이자(interest)를 받기 위해 필요한 최소 잔고 잔액(minimum balance)

```
3. float moncharge;
```

- 매월 이자 발생 시 최소한의 잔고가 없을 때 부과되는 벌금

MEMBER FUNCTIONS

```
1. InterestChecking(int num=0,float bal=0,float cmin=1000,float imin=2500,float chg=2,float rate=2.5,float monchg=10);
```

- 위의 매개변수를 acctnum, bal, minimum, minint, charge, intrate, and moncharge 에 할당하여 생성자 생성

```
2. void interest();
```

- 현재 잔액(bal)을 기준으로 월(연간 아님) 이자를 계산하고 이자율을 계산하고 이 값을 현재 잔액에 추가합니다.
- 잔액(bal)이 계좌의 최소 잔액(minimum balance)보다 적을 경우 해당 월에 대한 이자를 추가하지 않고 계좌 잔액(bal)에서 moncharge 만큼 차감합니다.
    - Cf) 월 이자 계산: intrate/100./12. : 이자는 월마다의 복리가 아니라 연이자를 월별로 나눠서 계산

```
3. virtual void print();
```

- Ex) 현재 잔액(bal)이 1200 이고 이자율(intrate)이 10%이면 그 달의 이자는 10 입니다. 이를 1200 에 더합니다.
- 아래와 같이 출력이 필요합니다.(출력화면에는 bank name 과 credit 이 옆으로 붙어있는데 아래와 같이 출력해주셔도 되고 출력화면처럼 출력해주셔도 됩니다.)

Interest Checking
Account: (account number)
Bankname(bankname)
Credit(credit)
Balance: (current balance)
Minimum to Avoid Charges: (minimum balance)
Charge per Check: (charge amount)
Minimum balance for getting interest and No Monthly Fee: (minimum interest balance) Interest: (interest rate) %
Monthly Fee: (monthly charge)

class BankSystem

이 클래스는 구현한 모든 종류의 account 들과 관련된 작업을 제어하는 은행 시스템을 나타냅니다.

MEMBER FUNCTIONS

```
void transaction(BankAccount* from, BankAccount* to, float amount);
```

- 두 개의 은행 계좌가 필요합니다. 첫 번째 계좌에서 해당 금액의 금액을 인출하여 두 번째 계좌로 입금합니다.
- 송금인은 거래 수수료(transaction_fee)를 지불해야 합니다. 거래수수료는 5 입니다.
- 만약 같은 bank_name 을 가지면 거래수수료를 받지 않습니다(거래수수료 0).
    - **Hint: Strcmp 를 사용하여 비교: #include <cstring> 사용**
- 송금인은 잔고에서 금액(amount) + 수수료(transaction_fee)을 잃지만 받는 사람은 금액만 받습니다(수수료(transaction fee) 제외).

```
void deposit(BankAccount* b, float amount);
```

- 송금인과 수신인이 없습니다. 하나의 계좌(account)만 가지고 잔액(balance)에 금액(amount)을 입금합니다.

```
void withdraw(BankAccount* b, float amount);
```

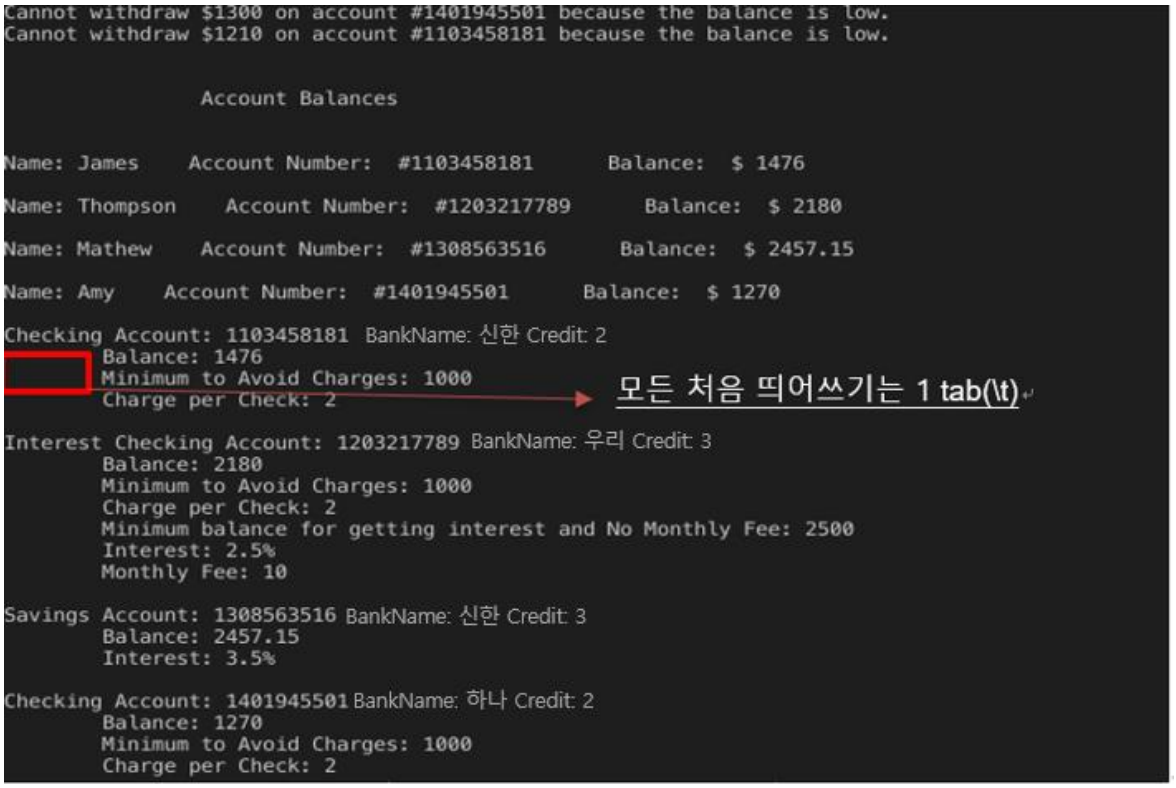
- 송금인과 수신인이 없습니다. 하나의 계좌(account)만 사용하고 잔액(balance)에서 금액(amount)을 인출합니다.

```
void loan(BankAccount* b, float amount);
```

- 현재 계좌잔액(bal)이 1000 미만이면 신용도 1, 대출 불가
    - 아래와 같이 출력
        - The amount cannot be loaned
- 계좌 잔액이 1000 이상 2000 미만이면 신용도 2: 대출시 빌려주는 금액의 10%를 대출수수료 받음
    - 신용도 2 의 고객 대출금액 한도: 100~500, 대출수수료 10%
        - 대출 한도외의 금액이 amount 에 설정되면 아래와 같이 출력
            - The amount cannot be loaned
        - 예시) 100(amount)를 빌리면 100*0.1(대출수수료 10)으로 10 제외하고 bank account 에 90 만큼 계좌잔액 증가
- 계좌 잔액이 2000 이상이면 신용도 3: 대출시 빌려주는 금액의 5%를 대출수수료로 받음
    - 신용도 3 의 고객 대출금액 한도:100~1000, 대출 대출수수료 5%
        - 대출 한도외의 금액이 amount 에 설정되면 아래와 같이 출력
            - The amount cannot be loaned
        - 예시) 100(amount)를 빌리면 100*0.05(대출수수료 5)으로 5 제외하고 bank account 에 95 만큼 계좌잔액 증가

Main.cpp 를 사용했을 때 출력 화면:

0



IMPORTANT NOTE for the assignment

클래스와 주요 기능 파일에 대한 모든 헤더 파일을 포함한 code bundle 을 사용하시면 됩니다.

위와 같은 출력결과를 나오도록 해주십시오. 난수로 생성된 은행들의 거래 수수료 차이로 5~15 의 금액 차이가 날수 있습니다. 감안하여 채점하도록 하겠습니다. loan 함수는 다른 4 개의 main.cpp 에서 테스트할 예정입니다.

HOW TO BUILD?

헤더 파일들과 main.cpp 가 같은 디렉토리에 있다는 전제 하에, terminal 로 그 해당 디렉토리에 접근하여 g++ main.cpp 를 실행하면 실행파일이 생성될 것입니다. 이때 생성된 실행파일로 main.cpp 의 결과를 확인해 볼 수 있습니다.

아래와 같이 실행해도 됩니다.

g++ main.cpp ~(나머지 모든 .cpp 파일들)~ -o main

PRINT RESULT?

printf 가 아닌 cout 을 사용했을 때 소수점이 저희가 제공한 결과와 동일하게 나오게됩니다. 그러니 cout 을 사용하시길 추천드립니다.

GRADING CRITERIA FOR Project 1

1. 주어진 것과 동일한 출력(난수로 생성된 은행이름으로 거래수수료로 인해 5~15 정도 금액이 다를 수 있습니다. 감안하여 채점하도록 하겠습니다.)<공백(/tab) 등 포함>(40pt)
2. 4 개의 다른 main.cpp(고객이 가지고 있는 금액이 다르거나 인출금액이 다른 상황이나 loan 함수를 사용하는 경우)를 실행하여 테스트. 15pt each. 총(60pt)

TOTAL: 100pt

SUBMISSION:

모든 파일을 제공된 header 와 함께 제출해주세요. 제공된 헤더는 수정할 수 있습니다.

다른 4 개의 main.cpp 는 공개하지 않고 채점후에 알려드리도록 하겠습니다.

메인 파일만 테스트할 것이므로 헤더는 다른 4 개의 main.cpp 에서 실행되어야 합니다. 테스트 예정인 다른 4 개의 Main.cpp 들은 "main.cpp" 파일에 표시된 것과 동일한 헤더를 추가해서 테스트 예정입니다.

모든 파일을 압축해주시고 압축 파일이름은 아래와 같이 하여 etl 에 제출해주세요.

"cp2023project1_(학번)_(이름).zip"

ex)cp2023project1_20xx-xxxxx_홍길동.zip