CPP Problem Design

Subject: BankAccount

Contributor: 邱韋霖, 鄭永泰, 范茗翔

Main testing concept: Basic I/O、Class

Basics	Functions
■ C++ BASICS	□ SEPARATE COMPILATION AND NAMESPACES
■ FLOW OF CONTROL	□ STREAMS AND FILE I/O
■ FUNCTION BASICS	□ RECURSION
□ PARAMETERS AND OVERLOADING	□ INHERITANCE
□ ARRAYS	□ POLYMORPHISM AND VIRTUAL FUNCTIONS
■ STRUCTURES AND CLASSES	□ TEMPLATES
■ CONSTRUCTORS AND OTHER TOOLS	□ LINKED DATA STRUCTURES
□ OPERATOR OVERLOADING, FRIENDS,AND REFERENCES	□ EXCEPTION HANDLING
□ STRINGS	□ STANDARD TEMPLATE LIBRARY
□ POINTERS AND DYNAMIC ARRAYS	□ PATTERNS AND UML

Description:

Write a definition of a class named BankAccount that will be used to store and amount of money call balance which are integers. You will need to declare and implement the following things:

a. Create a constructor "BankAccount(x)" that set the balance with initial value x. also have a default constructor "BankAccount()" that set balance with initial 0.
b. save(x): A member function to save money in the bank with an amount by the argument.
c. withdraw(x): A member function to withdraw money in the bank with an amount by the argument.
d. getBalance(): a const inspector functions to retrieve the current balance of the bank.
e. A static variable named: allMoneyInBank that track the total amount of BankAccounts have store.
f. A static function named: getAllMoneyInBank() that return the value of allMoneyInBank.

Noticed that the balance of BankAccount can be negative number so as allMoneyInBank

Input:

Replace main.cpp

Output:

See sample output.

Sample Input / Output：

Sample Input	Sample Output

#include "BankAccount.h"
int main(void) {
BankAccount bankAccount1(200), bankAccount2, bankAccount3(-100);
cout << BankAccount::getAllMoneyInBank() << endl;
bankAccount1.withdraw(100);
cout << bankAccount1.getBalance() << endl;
cout << BankAccount::getAllMoneyInBank() << endl;
bankAccount2.save(50);
cout << bankAccount2.getBalance() << endl;
	100
	100
	0
	50
	50

```
        cout << BankAccount::getAllMoneyInBank() << endl;
        system("PAUSE");
        return 0;
}
```

■ Eazy,Only basic programming syntax and structure are required.
□ Medium,Multiple programming grammars and structures are required.
□ Hard,Need to use multiple program structures or complex data types.

Expected solving time:
10 minutes

Other notes:
Replace main.cpp.
Only include BankAccount.h in main.cpp.