CPP Problem Design Example

Subject: School Inheritance

Contributor: JY

Main testing concept: Class/Inheritance/Overloading/Virtual Functions

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
	□ STANDARD TEMPLATE LIBRARY
□ STRINGS	□ PATTERNS AND UML
□ POINTERS AND DYNAMIC ARRAYS	

Write a main program that creates three types of school (one from each class: School, PrivateSchool, and PublicSchool) and transfers students from each to another one.

Create a base class called **School** that has member variables:

- the **name** of the school (a string)
- the **studentAmount** of the school currently
- **studentAmountNextYear** indicating the number of students the school could have next year, where when constructing is same as the student amount this year.
- ***admissions(float amount)***
  // adds the amount (passed as a parameter) to the total student amount this year (if the amount is nonnegative),
- ***dropouts(float amount)***
  //subtracts the amount from the number of students this year (if the amount is nonnegative and less than or equal to the student amount)
- ***transfer(float amount, School &toSchool),*** deducts from the student amount of current and transfers them to another school (passed as a parameter), implemented calling ***dropouts(amount)*** and ***toSchool.admissions(amount)*.**

Also, create a class called **PrivateSchool** that is derived from **School**. In a **PrivateSchool** one wave of dropouts are allowed. After the single wave, a penalty of 100 is deducted from the amount of students the school is able to admit next year for every ***dropouts*** called. Hence, the class must have a data member to keep track of the times of dropouts and override the ***dropouts function***.

Finally, create a **PublicSchool** class derived from **School,** which in addition to having the name and balance has a member variable ***growing_rate*** **(=0.05)** and a member function ***apply_growth()***, which increases amount of students able to admit next year by adding: **studentAmountNextYear +=** ***growing_ rate****** studentAmountNextYear**.
PublicSchool incur penalties when large amount of students(>100) leave the school at once. A dropout of such amount induces a loss of 5% of **studentAmountNextYear, truncating the decimal places**. Again, the ***dropouts function*** must override the one in the base class.

If violations occur (Ex. Subtracting more than existing amount), we do not do the operation.
For all 3 classes create constructors (default and with parameters) and overloaded << (output) operator, reuse constructors and operator << of the base class in the derived classes.

Input：

No Inputs.

Output：

Output format will be with name, tstudentAmount and tstudentAmountNextYear. Each separate by tab('\t').

**** "name\tstudentAmount\tstudentAmountNextYear", separated with tab**

Please refer to sample output for output format.

Sample Input	Sample Output
According to the given main.cpp in Other Notes	NTUST 12500 12500 NTUT 85000 85000 FJCU 25000 25000 NTUST 12700 12500 NTUST 12500 12500 NTUST 12500 12500 FJCU 26000 25000 FJCU 25950 25000 FJCU 24950 24900 NTUT 86000 85000 NTUT 86000 89250 NTUT 85000 84787 NTUT 85000 84787 NTUT 84000 80547 NTUST 13500 12500 FJCU 24950 24900 NTUST 13500 12500

□ **Easy, 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:

25 minutes

Other Notes:

```
#include <iostream>
#include "School.h"
#include <string>
using namespace std;

int main()
{
   //init 3 different account types
   School ntust("NTUST", 12500);
   PublicSchool ntut("NTUT", 85000);
   PrivateSchool fjcu("FJCU", 25000);
```

```
    //state info all 3
    cout<<ntust<<endl;
    cout<<ntut<<endl;
    cout<<fjcu<<endl;

    //test all methods on School
    ntust.admissions(200);
    cout<<ntust<<endl;

    ntust.dropouts(200);
    cout<<ntust<<endl;

    ntust.dropouts(100000);
    cout<<ntust<<endl;

    //test all methods on PrivateSchool
    fjcu.admissions(1000);
    cout<<fjcu<<endl;

    fjcu.dropouts(50);
    cout<<fjcu<<endl;

    fjcu.dropouts(1000);
    cout<<fjcu<<endl;

    //test all methods on PublicSchool
    ntut.admissions(1000);
    cout<<ntut<<endl;

    ntut.apply_growth();
    cout<< ntut <<endl;

    ntut.dropouts(1000);
    cout<< ntut <<endl;

    //Transfer method
    cout << ntut << endl;
    ntut.transfer(1000, ntust);
    cout << ntut << endl;
    cout << ntust << endl;

    fjcu.transfer(30000, ntust);
    cout << fjcu << endl;
    cout << ntust << endl;

    return 0;

}
```