**C++ Basic Course**

**6/16/2018**

# Lab Assignment 12

**Exercise 1 (40%)**: **(About inheritance and dynamic binding)** Write a base class `Fraction` and a derived class `RFraction` that inherits `Fraction`. In the base class `Fraction`, use `int` variables `numer` and `denom` to represent numerator (分子) and denominator (分母). In `RFraction`, represent `Fraction` in its reduced form. Provide constructors and proper IO operator overloading.

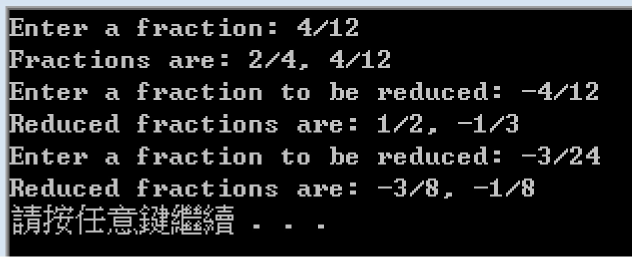
*You cannot define any member data in `RFraction`.

*The `operator>>` and `operator<<` should be only defined for `Fraction`.

Use the following client code and sample runs to test your program.

```
#include "Fraction.h"
#include "RFraction.h"
#include <iostream>
using namespace std;

int main(){
   Fraction f1(2, 4);
   Fraction f2;
   cout << "Enter a fraction: ";
   cin >> f2;
   cout << "Fractions are: ";
   cout << f1 << ", ";
   cout << f2 << endl;
//
   RFraction rf1(2, 4);
   RFraction rf2;
   cout << "Enter a fraction to be reduced: ";
   cin >> rf2;
   cout << "Reduced fractions are: ";
   cout << rf1 << ", ";
   cout << rf2 << endl;
//
   Fraction* rfp1 = new RFraction(-6, 16);
   Fraction* rfp2 = new RFraction();
   cout << "Enter a fraction to be reduced: ";
   cin >> *rfp2;
   cout << "Reduced fractions are: ";
   cout << *rfp1 << ", ";
   cout << *rfp2 << endl;
   delete rfp1;
   delete rfp2;
}
```

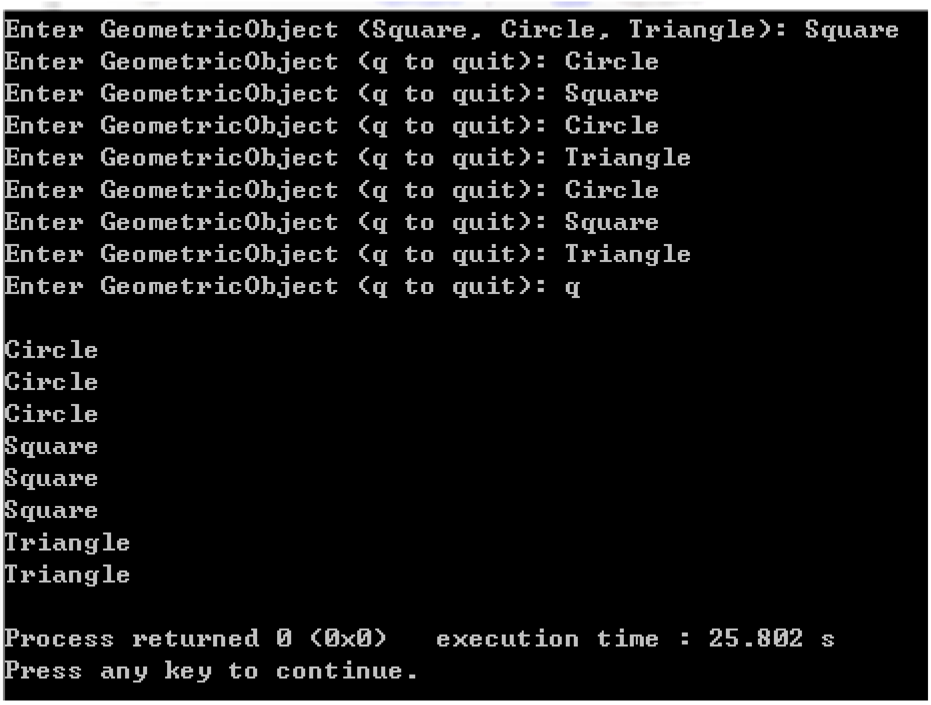


```
Enter a fraction: 4/12
Fractions are: 2/4, 4/12
Enter a fraction to be reduced: -4/12
Reduced fractions are: 1/2, -1/3
Enter a fraction to be reduced: -3/24
Reduced fractions are: -3/8, -1/8
請按任意鍵繼續 . . .
```

**Exercise 2 (40%)**: **(About polymorphism)** Design an abstract base class `GeometricObject` and three derived classes named `Circle`, `Square` and `Triangle` that extends `GeometricObject`. Allow users to create a few `Circle`, `Square` and `Triangle` objects on the fly, insert these objects into a `vector<GeometricObject*>` container and sort these objects based on their class name in an alphabetic order. Use the following client code and sample run to test your program.

```
int main() {
    vector<GeometricObject*> coll;
    string type;
    cout << "Enter GeometricObject (Square, Circle, Triangle): ";
    while (cin >> type && type != "q"){
        GeometricObject* g = createGeometricObject(type);
        coll.push_back(g);
        cout << "Enter GeometricObject (q to quit): ";
    }
    sort(coll.begin(), coll.end(), cmpGeom);
    cout << endl;
    for (const auto& e : coll){
        cout << *e << endl;
    }
// clean the resource
    for (auto& e : coll){
        delete e;
    }

    return 0;
}
```



```
Enter GeometricObject (Square, Circle, Triangle): Square
Enter GeometricObject (q to quit): Circle
Enter GeometricObject (q to quit): Square
Enter GeometricObject (q to quit): Circle
Enter GeometricObject (q to quit): Triangle
Enter GeometricObject (q to quit): Circle
Enter GeometricObject (q to quit): Square
Enter GeometricObject (q to quit): Triangle
Enter GeometricObject (q to quit): q

Circle
Circle
Circle
Square
Square
Square
Triangle
Triangle

Process returned 0 (0x0)   execution time : 25.802 s
Press any key to continue.
```