UEE 1303(1067/1069): Object-Oriented Programming

Programming HW #2

Due: 2014/5/16 23:59

1. (40%) Create a class called `RationalNumber` for performing arithmetic with fractions.

   Provide public member functions that perform each of the following tasks

   a) Create a constructor that prevents a 0 denominator in a fraction reduces or simplifies fractions that are not in reduced form and avoids negative denominator.
   b) Overload the addition, subtraction, multiplication and division operators for this class.
   c) Overload the relational and equality operators for this class.
   d) Overload the stream insertion (<<) and stream extraction (>>) operator for this class.

   Main program (hw2-1.cpp):

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

int main()
{
    RationalNumber c( 7, 3 ), d( 3, 9 ), x;

    cout << c << " + " << d << " = ";
    x = c + d; // test overloaded operators + and =
    cout << x << '\n';

    cout << c << " - " << d << " = ";
    x = c - d; // test overloaded operators - and =
    cout << x << '\n';

    cout << c << " * " << d << " = ";
    x = c * d; // test overloaded operators * and =
```

```
    cout << x << '\n';

    cout << c << " / " << d << " = ";
    x = c / d; // test overloaded operators / and =
    cout << x << '\n';

    cout << c << " is:\n";
    // test overloaded greater than operator
    cout << ( ( c > d ) ? "    > " : "    <= " );
    cout << d << " according to the overloaded > operator\n";

    // test overloaded less than operator
    cout << ( ( c < d ) ? "    < " : "    >= " );
    coud << d << " according to the overloaded < operator\n";

    // test overloaded greater than or equal to operator
    cout << ( ( c >= d ) ? "    >= " : "    < " );
    cout << d " according to the overloaded >= operator\n";

    // test overloaded less than or equal to operator
    cout << ( ( c <= d ) ? "    <= " : "    > " );
    cout << d << " according to the overloaded <= operator\n";

    // test overloaded equality operator
    cout << ( ( c == d ) ? "    == " : "    != " );
    cout << d << " according to the overloaded == operator\n";

    // test overloaded inequality operator
    cout << ( ( c != d ) ? "    != " : "    == " );
    cout << d << " according to the overloaded != operator" << endl;
} // end main
```

2. (60%) Create class `IntegerSet` for which each object can hold integers in the range 0 through 100. A set is represented internally as an array of ones and zeros. Array element `a[i]` is 1 if integer `i` is in the set. Array element `a[j]` is 0 if integer `j` is not in the set. The default constructor initializes a set to the so-called "empty set," i.e., a set whose array representation contains all zeros.

a) Provide member functions for the common set operations. For example, provide a `unionOfSets` member function that creates a third set that is the set-theoretic union of two existing sets (i.e., an element of the third set's array is set to 1 if that element is 1 in either or both of the existing sets, and an element of the third set's array is set to 0 if that element is 0 in each of the existing sets).
b) Provide an `intersectionOfSets` member function which creates a third set which is the set-theoretic intersection of two existing sets (i.e., an element of the third set's array is set to 0 if that element is 0 in either or both of the existing sets, and an element of the third set's array is set to 1 if that element is 1 in each of the existing sets).
c) Provide an `insertElement` member function that inserts a new integer `k` into a set (by setting `a[k]` to 1). Provide a `deleteElement` member function that deletes integer `m` (by setting `a[m]` to 0).
d) Overload the stream insertion (<<) operator for this class. It prints a set as a list of numbers separated by spaces. Print only those elements that are present in the set (i.e., their position in the array has a value of 1). Print --- for an empty set.
e) Overload the stream extraction (>>) operator for this class. If the input is out of range (0-100), you should print an error message. For example, if you want to insert value 305 into the set, display an error message “Invalid Element 305.”.
f) Provide an `isEqualTo` member function that determines whether two sets are equal.
g) Provide an additional constructor that receives an array of integers and the size of that array and uses the array to initialize a set object.

Main program (hw2-2.cpp):

```
#include <iostream>
#include "IntegerSet.h" // IntegerSet class definition
using namespace std;

int main()
{
    IntegerSet a;
    IntegerSet b;
    IntegerSet c;
```

```
IntegerSet d;

cout << "Enter set A:\n";
cin >> a;
cout << "\nEnter set B:\n";
cin >> b;
c = a.unionOfSets( b );
d = a.intersectionOfSets( b );
cout << "\nUnion of A and B is:\n";
cout << c;
cout << "Intersection of A and B is:\n";
cout << d;

if ( a.isEqualTo( b ) )
    cout << "Set A is equal to set B\n";
else
    cout << "Set A is not equal to set B\n";

cout << "\nInserting 77 into set A...\n";
a.insertElement( 77 );
cout << "Set A is now:\n";
cout << a;

cout << "\nDeleting 77 from set A...\n";
a.deleteElement( 77 );
cout << "Set A is now:\n";
cout << a;

const int arraySize = 10;
int intArray[ arraySize ] = { 25, 67, 2, 9, 99, 105, 45, -5, 100, 1 };
// here are two invalid value -5 and 105
IntegerSet e( intArray, arraySize );

cout << "\nSet E is:\n";
cout << e;

cout << endl;
```

```
} // end main
```

Sample output:

```
[lptung@oop ~]$ ./hw2-2
Enter set A:
Enter an element (-1 to end): 32
Enter an element (-1 to end): 9
Enter an element (-1 to end): -100
Invalid Element -100.
Enter an element (-1 to end): 39
Enter an element (-1 to end): -1
Entry complete

Enter set B:
Enter an element (-1 to end): 9
Enter an element (-1 to end): 100
Enter an element (-1 to end): -1
Entry complete

Union of A and B is:
{      9    32    39 100      }
Intersection of A and B is:
{      9      }
Set A is not equal to set B

Inserting 77 into set A...
Set A is now:
{      9    32    39    77      }

Deleting 77 from set A...
Set A is now:
{      9    32    39      }
Invalid Element 105.
Invalid Element -5.

Set E is :
{      1      2    9    25    45    67    99 100    }
```