C SC 397a, Spring 2010
Assignment 7
Due: Monday, April 5 at 22:00:00

Problem 1. (55 points) String

A simple String class that is very similar to the String class presented in the slides can be found in $FILES/a7/String.{h,cc}. You are to make the following improvements to the class:

(1) Modify the String(char) constructor so that it can be optionally invoked with an int length, too:

```
String s('c');        // Creates a string consisting of one 'c': "c"
String s('c', 5);     // Creates a string consisting of 5 'c's: "ccccc"
```

This constructor is designated as explicit at present; maintain that designation in your version.

(2) Add an assignment operator that specifically handles assignment of a char to a String. It results in a String with a length of 1:

```
String s("testing");
s = 'c';  // s is now "c"
```

(3) Add an overloaded negation operator that produces a reversed copy of the string:

```
String x("abc");
String y = -x;   // y is "cba", x is still "abc"
```

(4) Add an overloaded unary exclamation mark operator that produces a copy of the string with all lowercase characters converted to uppercase. Other characters are not affected.

```
String upped = !String("while (ctr1-- < Max)");
    // upped is "WHILE (CTR1-- < MAX)"
```

Note: Use toupper(c) for this; include the <cctype> header to gain access to it.

(5) Overload the multiplication operator so that "multiplying" a string by an integer produces a string that is a concatenation of n copies of the string:

```
String ab = String("ab") * 3;            // ab is "abababab"
String abc = 2 * (ab + String('c'));     // abc is "abababcabababc"
```

If n is zero, a zero-length String is produced. If n is negative terminate execution with an assertion failure, using wording of your choice. (Include the <cassert> header to use the assert macro.)

(6) Overload the division operator so that String / int produces a prefix or suffix of the string, depending on the sign of the integer. Examples:

```
String lets = "abcdefg";
String pfx = lets / 3;    // pfx is "abc"
String sfx = lets / -2;   // sfx is "fg" – negative value indicates suffix
```

```
lets / 0        // yields empty String
lets / 100      // yields "abcdefg"—the full string is produced if the
                //   length is exceeded
lets / -100     // yields "abcdefg"
```

Take advantage of other String operations and be as lazy as possible. Don't worry about performance! If you find yourself doing a lot of explicit memory management and writing tedious code involving pointers, look for a simpler solution. Hint: In my solution, new and delete are used only in the constructors, destructor, and operator+.

Problem 2. (55 points) IntList

In $FILES/a7/IntList.{h,cc} is a very simple class that represents an arbitrarily long list of integers. Here is an example of usage:

```
IntList L1;

for (int i = 0; i < 20; i += 3)
   L1.add(i);

L1.print("L1: ");
```

Output:

```
L1: 0, 3, 6, 9, 12, 15, 18
```

You are to make the following improvements to the IntList class:

(1) As-is, statements such as L2 = L1; and IntList L3(L2); cause a sharing of data that is undesirable. Make changes to eliminate that sharing. For example, after execution of the two statements immediately above, adding values to L1 should have no effect on L2 or L3.

(2) Overload the indexing operator so that L[n] returns the integer in position n (zero-based). Do not allow operations like L[n] = 3, i.e., ensure that a compilation error will be produced if L[n] appears on the left hand side of an assignment.

If n is out of bounds, terminate execution with an assertion failure, using wording of your choice. If n is negative, it is out of bounds.

(3) Overload the multiplication operator so that an IntList can be multiplied by an integer: L * n **and** n * L. The result is a new list (R) such that R[i] == L[i] * n, for all i. Example:

```
L.print("L: ");     //     L: -10, 17, 0
L2 = L * 5;
L2.print("L2: "); //    L2: -50, 85, 0    (L is unchanged)
```

(4) Overload the addition operator so that two IntLists can be "added", as follows. Given IntLists L1 and L2 , L1 + L2 produces a new list (R) such that R[i] == L1[i] + L2[i], for all i.

If the lists are of unequal length the result should have the length of the longer list. Positions in the result that correspond to non-existent positions in the shorter list should have the value from that

position in the longer list. (In other words, treat non-existent positions in the shorter list as if they had the value zero.)

Example:

```
IntList L1, L2;
L1.add(15); L1.add(20); L1.add(30);
L2.add(20); L2.add(10);

L1.print("L1: ");          //  L1: 15, 20, 30
L2.print("L2: ");          //  L2: 20, 10

IntList L3 = L1 + L2;
L3.print("L3: ");          //  L3: 35, 30, 30

IntList L4 = L1 + L2 + L3;
L4.print("L4: ");          //  L4: 70, 60, 60
```

"Adding" two empty lists produces an empty list. Adding an empty list and a non-empty list produces a list with the same values as the non-empty list.

(5) Add a member function so that using an IntList where an int is required causes the sum of the integers to be used instead. Example:

```
IntList hundred;
for (int i = 1; i <= 100; i++)
   hundred.add(i);

int sum = hundred;
printf("sum = %d\n", sum);      //  sum = 5050

int val = hundred + -1 * hundred;
printf("val = %d\n", val);          //  val = 0
```

Problem 3. (10 points) answers.txt

Create a text file named answers.txt with answers to the following questions. (5 points each)

(1) Slide 92 states *..the language feature that "sealed the deal" to include references is operator overloading*. Why are references needed to effectively support operator overloading?

(2) The solutions for problem 2(7) on assignment 5 say that although it's incorrect, using delete to free memory obtained with malloc() doesn't necessarily cause a program to blow up. Following the mindset exhibited by the various examples of C++ language design decisions that we've seen, briefly address this proposed change to C++: *A warning or perhaps even an assertion failure should be produced if delete is used to free memory obtained using malloc()*. (In other words, respond to that proposal as you think Stroustrup might respond it.)

Problem 4. (Extra Credit) extra.txt

Submit a plain text file named extra.txt with the following.

(a) (1 point extra credit) Estimate how long it took you to complete this assignment. Other comments about the assignment are welcome, too. I appreciate all feedback, favorable or not.

(b) (1-3 points extra credit) Cite an interesting course-related observation (or observations) that you made while working on the assignment. The observation should have at least a little bit of depth.

Think of me saying "Good!" as one point, "Interesting!" as two points, and "Wow!" as three points. I'm looking for quality, not quantity.

Miscellaneous

You must make good use of C++, including but not limited to:
Using bool, const, and member initializers when appropriate
No public data members
Using new and delete rather than malloc and free

Be sure to avoid memory leaks.

Test programs with names of the form s?.cc and il?.cc, and corresponding executable reference versions are in $FILES/a7. Each of the test programs has a comment designating the point value. Each is all or nothing for the specified point value; if there's a diff, even one byte, it'll be zero points for that one. (Use diff to be sure that the output of your solutions exactly matches the output of the reference versions!) The set of test programs, reference versions, and any associated data files will freeze exactly one week (168 hours) prior to the due date/time.

$FILES/a7/{teststring,testintlist} are simple test scripts.

Feel free to use comments to document your source code as you see fit, but note that no comments are required.

You may distribute code between .h and .cc files as you see fit.

You should be able to complete this assignment using the material on slides 1-189.

Don't hesitate to ask the instructor for hints and/or help if you have trouble with a problem.

Deliverables

Use `turnin` with the tag 397a_7 to submit your solutions for grading. The deliverables for this assignment are String.h, String.cc, IntList.h, IntList.cc, answers.txt, and if you choose to submit it, extra.txt.