CSCI-1200 Data Structures — Fall 2012
Lab 5 — Vectors and Iterators

Introduction

This lab explores our implementation of the STL `vector` class and the use of vector iterators. Please download:

```
http://www.cs.rpi.edu/academics/courses/fall12/ds/labs/05_vectors_iterators/vec.h
http://www.cs.rpi.edu/academics/courses/fall12/ds/labs/05_vectors_iterators/test_vec.cpp
```

Checkpoint 1

Complete the implementation and testing of the `resize` and `erase` member functions of the `Vec<T>` class (the DS implementation mimicking the STL `vector` class). The main function in `test_vec.cpp` includes code to do the testing for you.

To verify your code is does not contain memory error or memory leaks, you can run Valgrind or Dr. Memory on your local machine (see instructions on the course webpage under "Memory Debugging". Or submit your code to the homework server (check the button next to lab 5), which is configured to run Valgrind for this exercise.

To complete this checkpoint, show a TA your tested & debugged program.

Checkpoint 2

Write and test a function named `reverse_1` that reverses the contents of an STL `vector` of integers. For example, if the contents of the vector are in increasing order before the call to `reverse_1`, then they will be in decreasing order afterwards. For this checkpoint, use **indexing/subscripting** on the vector, not iterators (or pointers). You may not use a second vector or array.

The trick is to step through the vector one location at a time, swapping values between the first half of the vector and the second half. As examples, the value at location 0 and the value at location `size()-1` must be swapped, and the value at location 1 and the value at location `size()-2` must be swapped.

Write a main function to test the function you have written. The main function should (a) create a vector of integers, (b) output the contents, (c) pass the vector to the reverse function, and then (d) output the resulting vector. To help with this, you should write an additional function that prints the size and the contents of a vector (so you don't need to keep writing for loops over and over). Your main function should test special cases of empty vectors and vectors of one or two values. Then you should test "typical" cases. Once your code is debugged using the STL `vector`, switch to using the DS `Vec` version instead. This change should be minimal because the interfaces are similar.

To complete this checkpoint, show a TA the completed and correct reverse function and the test main function, and then show the TA the compilation and correct output. Explain what you had to do to switch the implementation from the STL `vector` class to the DS `Vec` class.