CS 251: Intermediate Software Design

Programming Assignment 2

The second assignment gives you an opportunity to use the following more advanced features of C++:

- **STL algorithms**. STL algorithms (*e.g.*, `std::fill()`, `std::equal()`, and `std::copy()`) must be used to implement the methods in the `Array` class. In particular, you won't write any loops explicitly.
- **Templates**. A limitation of the `Array` class from assignment 1 is that it only works on arrays of chars. To generalize this behavior, change the `Array` class to be a parameterized type by using C++ **templates**, which allow you to parameterize the class with the desired data type.
- **All errors propagated via exceptions rather than return values**. In the first assignment the undergrads indicated range errors in their `get()` and `set()` methods via return values. In the second assignment all errors need to be propagated via exceptions, just like the grad students did with their first assignment.
- **Strong exception handling guarantees**. Your new implementation of the `Array` class methods must support strong exception handling guarantees via the use of `std::auto_ptr` (or better yet `std::unique_ptr` if you have a C++11 compiler) and the `scoped_array` class we discussed in class. See `www.cs.wustl.edu/~schmidt/exceptions` for some articles about C++ exception handling in general and strong exception guarantees in particular.
- **STL iterators** Students taking the class for graduate credit will need to implement STL-like iterators for the `Array`.

The semantics of the `Array` have changed so that if you try to `set()` outside the current range the `Array` will grow automatically rather than throw an exception. In addition, a `resize()` method has been added that can be called to grow the array explicitly (please use `Array::resize()` in your implementation of `Array::set()`). You must ensure that the default value (if any) is used to initialize any new array elements when the `Array::resize()` method is called. Likewise, the default value (if any) must be propagated properly via the `Array` assignment operator and copy constructor.

You can get the "shells" for the program from `www.dre.vanderbilt.edu/~schmidt/cs251/assignment2`. The Makefile, `main.cpp`, and `Array.h` files are written for you. All you need to do is edit the `Array.cpp` and `Array.inl` files to add the methods that implement the Array class. Note that the definition of the methods in your `Array.cpp` and `Array.inl` files changed from assignment 1 to use the C++ template and exception syntax.

If you are taking CS 251 for graduate credit please use the shells that are in the `grad` directory at the URL above. If you are taking it for undergraduate credit please use the shells in the `ugrad` directory at the URL above. Naturally, undergraduates are welcome to implement the graduate version, as well.