HW15 – STL Containers & Iterators [100 pts]

Answer the following questions by modifying the hw15.cpp source file and/or answering the question directly:

1. [read sgi.com – intro to stl, read wikipedia - stl] What is an stl container? stl iterator? stl algorithm? Give examples of each.
2. [read sgi.com – intro to stl, read wikipedia - stl] Explain how the iterator architecture makes it possible to decouple algorithms from containers. Why is this important?
3. [read learncpp.com – stl containers, read cplusplus.com – standard containers, sgi.com - containers] The *list* container class must implement constant iterators *begin* and *end* as do other stl containers. Note the forward declaration of the *const_iterator* class within *list's* definition. Add the missing constant *begin* and *end* iterators to *list*.
4. [read learcpp.com – stl iterators, read cprogramming.com – stl iterators, cplusplus.com – standard iterators, read sgi.com - iterators, read cprogramming.com – const correctness] Complete the missing implementation details for class *iterator*. Implement overloaded operators: *++, --, *, ==, !=* for *iterator*. An outline for class *const_iterator* is provided. *const_iterator* is much like *iterator* only all operations are *const*. No modifications to data pointed to by *const_iterator* are allowed. Add a *constructor* & overloaded *const* operators: *++, --, *, ==, !=*. Operators must return a *const pointer or const reference* where applicable.
5. [ALL reading material] The *low_doubles* algorithm will find the lowest value in an array of doubles. *low_doubles* has local variables *l* and *low* which are a source of errors. What kinds of issues might arise? Implement the generic (i.e. templated) *low* algorithm. *low* takes *iterator* arguments which point to the beginning, *iterator first*, and one past the end, *iterator last*, of a sequence of container elements. Take

advantage of the type parameter *iterator* to eliminate local variables *l* and *low* in writing the *low* algorithm. In what ways are the algorithms *low_doubles* and *low* similar? different? In *main* the output for finding the lowest value in the first half of vector *v* differs for *low_doubles* vs *low*. What is happening here?

Include comments in your code to indicate which code segment answers which question. Appended written answers to the bottom of the hw15.cpp source file (as source comments via //).

Use the command script to capture your interaction compiling and running the program, including all operations, as shown below:

CS1C Spring 2021 TTH HW15 100 pts **Due: Th 5/6/2021**

```
cs1c@cs1c-VirtualBox ~/cs1c/hw/15 $ script hw15.scr
Script started, file is hw15.scr
cs1c@cs1c-VirtualBox ~/cs1c/hw/15 $ date
...
cs1c@cs1c-VirtualBox ~/cs1c/hw/15 $ ls -l
...
cs1c@cs1c-VirtualBox ~/cs1c/hw/15 $ make all
...
cs1c@cs1c-VirtualBox ~/cs1c/hw/15 $ ls -l
...
cs1c@cs1c-VirtualBox ~/cs1c/hw/15 $ ./hw15

... // print out output from steps 1 thru 5

cs1c@cs1c-VirtualBox ~/cs1c/hw/15 $ exit
Script done, file is hw15.scr
cs1c@cs1c-VirtualBox ~/cs1c/hw/15 $ make tar
```

HW15 – STL Containers & Iterators [100 pts]

...

Submit the tar package file hw15.tar by Thursday May 6, 2021.