Week 13 — December 6, 2018

Homework

- Read pp. 52-53, pp. 168-172, pp. 180-182, and pp. 495-499 in "Beginning C++"
- Read pp. 106-117 in "Writing Scientific Software"
- Read about the STL vector container
- Read about the STL list container

Exercises

1. Consider the following `max` function template (defined in the `std` namespace):

   ```
   template <class T>
   const T& max (const T& a, const T& b) {
       return (a<b)?b:a;
   }
   ```

   This allows us to find the "maximum" of any two variables `a` and `b` of type `T` for which the inequality `a < b` is defined.

   Use the above template as inspiration to write a template for finding the absolute value of an variable `a` of type `T` for which the inequality `a < -a` is defined. Write a short program to test your template with different numerical types (e.g., `int`, `float`, and `double`).

   Your template should not return a reference to a variable of type `T`. Explain why not.

2. The `vector` class from the Standard Template Library (STL) implements a generic list with an underlying array representation of the data. In other words, the elements of the vector are stored contiguously in memory. In this exercise, we will read a series of floating point numbers from a text file and use a `vector` object the store them. After reading all numbers, the program should calculate and output the sum of the numbers and the mean of the numbers.

   Modify your program so that it prints the size of the vector (using the method `size()`) and its capacity (using the method `capacity()`) after reading a number. What is the complexity of appending a number to the vector?

   Hint: Use the `push_back()` method to append a number to the vector.

   You may use the following template:

```
#include <iostream>
#include <vector>
int main(int argc, const char *argv[]) {
    std::vector<double> v;
    // insert code here
    return 0;
}
```

3. Write a program that prompts the user to enter a sequence of strings, and store all the strings using a `list` object. When the user is done entering words, prompt the user to enter a number m and remove all strings that are longer than m characters from the list. Print the remaining elements in the list.

   What is the complexity of removing all strings of length at least m?

   Hint: Use the `push_back()` method in the `list` class to add a string to your list. You can use the `length()` method in the `string` class to find the length of a string.

   You may use the following template:

```
#include <iostream>
#include <list>
int main(int argc, const char *argv[]) {
    std::list<string> L;
    // insert code here
    return 0;
}
```