Chandler Teigen
PA1
CPTS 223

A.

The problem is to populate a singly linked list in C++ with integers that are read from a text file. Then, the max, min, and median of the integers are to be found along with the time taken to populate the list, and calculate each of the stated values.

B.

My algorithm for finding the min of the list was very simple. Because the list is a singly linked list, there exists a pointer to the first node in the list. So, the first element of the list can be found using the STL container function std::list::begin(). Then the minimum of the list is found because the first element in the sorted list is the minimum. The maximum was found by finding the size of the list using std::list::size(), then iterating through the integers in the list until the last item was reached. This item was the max of the list because it is a sorted list. The median was more complicated to calculate because it depended on whether the size of the list was even or odd. If the list was of off size, the middle element was found using the size value, then the integer stored at that position in the list was returned. If the list was even sized, the two middle integers were found, their average was computed, and the value was returned. These algorithms were as efficient as they could have been given the restriction of using a singly linked list. The minimum value could have been found by iterating through the entire list, but since it is a sorted list, the minimum is simply the first element. The same can be said for the median and maximum. Because the list is sorted, many data comparisons can be avoided which makes the algorithm more efficient.

C.

My experimental setup consisted of using a Macbook Pro 2015 laptop to run the code, and a MacOS Unix environment. I repeated the experiment three separate times and averaged the results in order to account for variations in the program run-time, and obtain more accurate results.

D.

The result of my experiment was finding a max of 2057, a min of 1 and a median value of 2057 for the first data set. The time to sort the entire data set was 7418 microseconds. The time to find the max, min and median were on average 14 microseconds, 0 microseconds, and 1 microsecond respectively. I was unable to complete the test on the larger data set because the run time exceeded an hour with less than half of the data processed. Because the processing time for each data point became larger as more values were inserted into the list, the second half of the data was going to take many times longer to process than the first half. At the point that I realized this, I terminated the program and decided that a better program design would be needed to sort the numbers in the second file. My suggestion would be to use an indexable data structure so that a logarithmic algorithm could be used to insert the new data points. The linked list requirement from the assignment is not an efficient data structure for use in a insert-in-order problem because the list has to be traversed in order rather than the list allowing random access. If random access was possible, an algorithm similar to binary search could be used to reduce the remaining data in the list by half each iteration of the insert algorithm rather than having to iteration linearly through the list.