Data Structures (810:052) Lab 4 Name:__________________

**Objectives:** You will gain experience:

- using STL containers and algorithms (see http://www.cplusplus.com/reference/stl/)

Download the following file to your desktop: http://www.cs.uni.edu/~fienup/cs052s10/labs/lab4.zip
Extract this file by right-clicking on lab4.zip icon and selecting Extract All.

**Part A:** In lecture 7 we discussed the STL container classes. Because of the difference in the way elements are arranged in memory, inserting new elements at various locations (e.g., back, front, etc.) take a different amount of work. The `lab4.zip` file you downloaded and extracted contains a `InsertDeleteContainerTest` folder with a Visual Studio C++ project file: `ContainerTest.sln` inside. Double-click on it to open this project in Visual Studio.

a) Run the current main program (in `main.cpp`) to complete the following tables and answer the corresponding questions.

| Container | Timing of 150,000 Elements (seconds) | | | |
|---|---|---|---|---|
| | Filled using `push_back` member function | Emptied using `erase` with iterator at the beginning | Filled using `insert` with iterator at the beginning | Emptied using `pop_back` member function |
| vector | | | | |
| list | | | | |

- For the vector, why did filling using `push_back` take must less time than filling using `insert` with an iterator at the beginning?

- For the vector, why did emptying using `pop_back` take must less time than `erase` with an iterator at the beginning?

- For the list, why did all of the filling and emptying take very little time?

b)

| Container | Timing of 150,000 Elements (seconds) | |
|---|---|---|
| | Filled using `insert` | Emptied using `erase` |
| set | | |
| map | | |

- Why did filling the set and map containers take more time than filling the vector by using `push_back`?

**After you have answered the above questions, raise your hand and explain your answers.**

**Part B:** Because of the difference in the way elements are arranged in memory in the various containers, searching for a target element take a different amount of work. The `lab4.zip` file you downloaded and extracted contains a `FindContainerTest` folder with a Visual Studio C++ project file: `ContainerTest.sln` inside. Double-click on it to open this project in Visual Studio.

a) Run the current main program (in `main.cpp`) to complete the following tables and answer the corresponding questions.

| Container | Time to Successfully Find all 20,000 Elements in random order. (seconds) |
|---|---|
| vector using `find` algorithm | |
| list using `find` algorithm | |
| set using `find` member function | |
| map using `find` member function | |

- Why did the `find`s for the vector and list take about the same about of time?

- Why do you suppose that the `find`s for the set and map take less time than the vector and list?

**After you have answered the above questions, raise your hand and explain your answers.**

**Part C:** The `lab4.zip` file you downloaded and extracted contains a `FindSortedContainerTest` folder with a Visual Studio C++ project file: `ContainerTest.sln` inside. Double-click on it to open this project in Visual Studio.

a) This code is similar to Part B, but it sorts the randomly generated integers in the vector so the `binary_search` algorithm can be used. The sorted elements are then added to the other contains in sorted order for a fair comparison. Run the main program (in `main.cpp`) to complete the following tables and answer the corresponding questions.

| Container | Time to Successfully Find all 20,000 Elements in random order. (seconds) |
|---|---|
| vector using `binary_search` algorithm<br>(elements were previously sorted) | |
| list using `binary_search` algorithm<br>(elements were inserted into list in sorted order) | |
| set using `find` member function<br>(elements were inserted into set in sorted order) | |
| map using `find` member function<br>(elements were inserted into map in sorted order) | |

- Why did the `find`s for the vector in Part B take longer than the `binary_search` on the vector in Part C?

- Why did the `binary_search`s for the list in Part C take longer than the `find`s on the list in Part B?

**After you have answered the above questions, raise your hand and explain your answers.**