Project 2 (Part B): STL Candidate List

Due date:

- **MW class** ➔Monday, May 9, at the beginning of class
- **TTh class** ➔ Tuesday, May 10, at the beginning of class

Project name: A250_P2B_Teamname

SECTION 1 – Managing memory

You certainly do **not** want an object that contains a whole vector. It would be better to use a pointer that points to a vector outside the object. Therefore, for this part of the project, you will change the member variable **candidates** of the class **CandidateList** to a **pointer** that points to a **vector** of **type CandidateType**.

Once you have changed the member variable, you will need to modify a few sections in the member functions of the class:

- **Constructor**: Since you have a pointer to a vector now, then you need to create a **dynamic STL vector** of type **CandidateType**.
- **Dot operator**: You are now dealing with a **member variable** (**candidates**) that is a **pointer**; therefore the dot operator will not work. You need to change it to an **arrow operator** ➔ Do this throughout the file. (While doing this, make sure you reason on what you are changing, rather than just doing it.)
- **Destructor:** Since you have a dynamic variable, you will need to manage memory; therefore, delete the vector the pointer is pointing to and null the pointer. You do **not** need to go through the whole vector, because the vector will take care of itself, but you need to start the process by using the **delete** operator.

IMPORTANT: Since now the class **CandidateList** creates objects that contain only a **pointer**, the function **addCandidate** should be a **const** function. Why?

SECTION 2 – Printing final results

Now that you know about the **STL <map>**, you can make your function **printFinalResults** more efficient than it is.

Include **<map>** in the **CandidateList.h** file. Modify the **printFinalResults** function as follows:

- Create a **map** and copy all the **social security numbers** (**first** item of the pair) and their correspondent **names** (first name and last name as **second** item of the pair) from the vector into the **map**.
- The **map** will have the **social security numbers in ascending order**. To print them in **descending** order, simply use a **reverse** iterator.
- Go to **cplusplus.com** and search for "**right**." This will take you to the **<ios>** function **right**, which justifies output to the right. There is an example that shows how to use the function **width**. You will use these two functions to produce the output of selection 5 as shown below ➔ **Change** the **candidates_data.txt** file with the **new** one provided.
- The **vertical line** (**|**) is a symbol on the keyboard, where the backslash (\ ) is.

Replace the **candidates_data.txt** file with the new one and test your program against the **output.exe** file given.

```
*** MAIN MENU ***

Select one of the following:

    1: Print all candidates
    2: Print a candidate's division votes
    3: Print a candidate's total votes
    4: Print winner
    5; Print final results
    6: To exit

Enter your choice: 5

FINAL RESULTS
-------------
    1 | 233 | Edsger Dijkstra
    2 | 221 | Grace Hopper
    3 | 208 | John McCarthy
    4 | 199 | Erna Schneider
    5 | 173 | Robert Tarjan
    6 | 167 | Meg Whitman
    7 | 166 | James Cooley
    8 | 159 | Alan Turing
    9 | 135 | Charles Babbage
   10 | 132 | Steve Whittaker
   11 | 130 | Joseph Kruskal
   12 | 108 | Anita Borg
   13 | 105 | Ada Lovelace
   14 | 101 | Byarne Stroustrup
   15 |  99 | Bert Bos
   16 |  94 | Claude Shannon
   17 |  86 | Alan Kay
   18 |   8 | Bill Joy

Press any key to continue . . .
```

The Big Three

Since you have **dynamic variables** in your class, you should make sure that **the big three** are implemented. You already have the **destructor**, but you will need to add a **copy constructor** and the **overloaded assignment operator**. This is simpler than it sounds, but it requires some thinking. You need to make sure that both the copy constructor and the assignment operator create new containers. To check this, you can either look in the debugger to see if the addresses are different, or you can create temporary functions to check that your copy constructor and overloaded assignment operator are doing what is expected.