**Project 1 – Part E – Testing Candidate List**

Test your project using the **p1_d_testing_candidate_list** file and make all necessary corrections.

**Due date:**

- MW class ➔Monday, April 11, at the beginning of class
- TTh class ➔ Tuesday, April 12, at the beginning of class

**Project name: A250_P1_FINAL_Teamname**

## PART A – ALPHA TESTING

Test the project given by checking if all the items in the list below are properly implemented.

| |
|---|
| **PersonType.h** |
| (No need for name header in this file.) |
| **Order** (with spacing) should be:<br><br>1. #ifndef<br>2. #define<br>3. (blank line)<br>4. All #include statements<br>5. (blank line)<br>6. using…<br>7. (blank line)<br>8. Class definition |

**Functions** are in this **order**:

1. Default constructor
2. Overloaded constructor
3. Function setPersonInfo
4. Function getFirstName
5. Function getLastName
6. Function getSSN
7. Function printName
8. Function printPersonInfo
9. Function printSSN
10. Destructor

All **string** parameters are passed by **reference** and as **const**.
All **int** parameters are passed by **value**.

The following functions are **const** functions:

1. Function getFirstName
2. Function getLastName
3. Function getSSN
4. Function printName
5. Function printPersonInfo
6. Function printSSN

**PersonType.cpp**

(No need for name header in this file.)

(No need to include any libraries and/or namespaces. They are already included in the .h file**)**

| |
|---|
| **Functions** are in the same **order** shown in the **Person.h** file.<br>There is a **blank line** separating each function definition. |
| **Default constructor** initializes social security number to 0.<br>**Default constructor** does **NOT** initialize the first and last name variables (**UNLESS** you initialized them as "N/A" or similar). |
| **Overloaded constructor** initializes all three member variables to the given values passed by the parameters. |
| Function **setPersonInfo** initializes all three member variables to the given values passed by the parameters. |
| Function **getFirsName** has one statement only (return). |
| Function **getLastName** has one statement only (return). |
| Function **getSSN** has one statement only (return). |
| Function **printName** prints the person's name as shown in the instructions. |
| Function **printPersonInfo** calls function **printSSN** and prints the person's name as shown in the instructions. |
| Function **printSSN** prints the social security number as shown in the instructions.<br>**NOTE:** If you are casting the **ssn** as a string and using the **substr** function to break the number in three parts, **change your implementation** by using / (division) and % (mod) to handle the **ssn** ➔ <u>This is a more efficient way</u>. |
| **Destructor** is empty. |

**CandidateType.h**

(No need for name header in this file.)

**Order** (with spacing) should be:

1. #ifndef
2. #define
3. (blank line)
4. #include “PersonType.h”
5. (blank line)
6. Other #include statements
7. (blank line)
8. using…
9. (blank line)
10. Constant NUM_OF_DIVISIONS declared and initialized.
11. (blank line)
12. Class definition

Class **inherits** from **PersonType.**

**Functions** are in the **order** specified in the instructions:

1. Default constructor
2. Function updateVotesByDivision
3. Function getTotalVotes
4. Function getVotesByDivision
5. Function printCandidateInfo
6. Function printCandidateTotalVotes
7. Function printCandidateDivisionVotes
8. Destructor

| |
|---|
| All **int** parameters are passed by **value**. |
| The following functions are **const** functions:<br><br>1. Function getTotalVotes<br>2. Function getVotesByDivision<br>3. Function printCandidateInfo<br>4. Function printCandidateTotalVotes<br>5. Function printCandidateDivisionVotes |
| **Member variables:**<br>• An integer to store the total votes.<br>• An array of integers of capacity NUM_OF_DIVISIONS to store votes by division. |
| **CandidateType.cpp** |
| (No need for name header in this file.) |
| (No need to include any libraries and/or namespaces. They are already included in the .h file**)** |
| **Functions** are in the same **order** shown in the **Person.h** file.<br>There is a **blank line** separating each function definition. |
| **Default constructor** initializes total votes to 0.<br>**Default constructor** initializes all indices of the array to 0. |

| |
|---|
| Function **updateVotesByDivision** updates member variables with the values passed by the parameters. |
| Function **getTotalVotes** has one statement only (return). |
| Function **getVotesByDivision** has one statement only (return). |
| Function **printCandidateInfo**<br>Calls parent's functions printSSN and printName to print out the candidate's information.<br>(There should **NOT** be the class qualifier and scope resolution when calling the functions **printSSN** and **printName**. Why?) |
| Function **printCandidateTotalVotes** calls parent's function **printName** and output the rest with a cout statement.<br>(There should **NOT** be the class qualifier and scope resolution when calling the function **printName**. Why?) |
| Function **printCandidateDivisionVotes** alls parent's function **printName** and output the rest with a cout statement.<br>(There should **NOT** be the class qualifier and scope resolution when calling the function **printName**. Why?) |
| **Destructor** is empty. |
| **CandidateList.h** |
| (No need for name header in this file.) |
| (No need to include any libraries and/or namespaces. They are already included in the .h file**)** |

**Order** (with spacing) should be:

1. #ifndef
2. #define
3. (blank line)
4. #include “CandidateType.h” (only)
5. (blank line)
6. #include all STL headers
7. (blank line)
8. using…
9. (blank line)
10. Class Node definition
11. (blank line)
12. Class CandidateList definition

Functions are in the **order** specified in the instructions:

1. Default constructor
2. Function addCandidate
3. Function getWinner
4. Function printCandidateName
5. Function printAllCandidates
6. Function printCandidateDivisionVotes
7. Function printCandidateTotalVotes
8. Function printFinalResults
9. Function destroyList
10. Destructor

The parameter **object** passed by the function **addCandidate** is passed by **reference** and as a **constant**.
All **int** parameters are passed by **value**.

Function **addCandidate**: **CandidateType** parameter is passed by **reference** and as a **const**.

The following functions are **const** functions:

1. Function getWinner
2. Function printCandidateName
3. Function printAllCandidates
4. Function printCandidateDivisionVotes
5. Function printCandidateTotalVotes
6. Function printFinalResults

**CandidateList.cpp**

(No need for name header in this file.)

**Default Constructor**

- sets both pointers to NULL
- sets the count to zero

Function **addCandidate**

- adds to the back of the list
- creates a node containing information passed by parameters
- does **not** traverse the list, because it uses the pointer **last** to add to the back of the list
- updates count

Function **getWinner**

**IF block:**

- considers when the list is empty
- (uses cerr to output empty list message)

**ELSE block:**

- declares a pointer to traverse the list
  (other variables might be needed)
- while loop to process counting of the votes and keep track of higher votes
- while loop contains if/else statement
- returns a social security number (make sure all paths return a value)

Function **printCandidateName**

**IF block:**

- considers when the list is empty
- (uses cerr to output empty list message)

**ELSE block:**

- declares a pointer to traverse the list
- declares boolean variable to stop the loop
- while loop to find candidate => loop **stops** when candidate is found and prints out information
- while loop contains if/else statement
- if candidate not found, outputs error message (uses cerr)

Function **printAllCandidates**

**IF block:**

- considers when the list is empty
- (uses cerr to output empty list message)

**ELSE block:**

- declares a pointer to traverse the list
- while loop to traverse the list and print information

Function **printCandidateDivisionVotes**

**IF block:**

- considers when the list is empty
- (uses cerr to output empty list message)

**ELSE block:**

- declares a pointer to traverse the list
- declares boolean variable to stop the loop
- while loop to find candidate => loop **stops** when candidate is found and prints out information
- while loop contains if/else statement

Function **printCandidateTotalVotes**

**IF block:**

- considers when the list is empty
- (uses cerr to output empty list message)

**ELSE block:**

- declares a pointer to traverse the list
- declares boolean variable to stop the loop
- while loop to find candidate => loop **stops** when candidate is found and prints out information
- while loop contains if/else statement

| |
|---|
| Function **printCandidateTotalVotes**<br><br>**IF block:**<br>• considers when the list is empty<br>• (uses cerr to output empty list message)<br>**ELSE block:**<br>• No specific instructions were given to implement this function. Analyze your implementation to determine whether it is efficient and whether it can be improved. Your implementation should not exceed an **O(n^2)** running time. |
| Function **destroyList**<br><br>• declares a pointer to traverse the list<br>• while statement deletes each node<br>• count is updated<br>• first and last pointers are set to NULL |
| **Destructor:** Calls function **destroyList** |
| **Main.cpp** |
| File has been modified to allow additional selection to the menu. |
| **OTHER** |
| There is **NO horizontal scrolling**. All statements are short enough. |
| There is a **space** before and after **operators**. |

| |
|---|
| All code is properly indented. |
| All **variables** and **objects** have a **descriptive** identifier. |
| All **variables** and **objects** use the **camelCase** convention. |

## PART B – BETA TESTING

Test the program using the test cases given below. Write what you would expect the output to be and then copy and paste the output the program gives you. Mark with an "X" if the output does not match.

Each response to menu selections should terminate with “Press any key to continue…”

| | **Expected Output** |
|---|---|
| Choice: 1 | 123-45-6789 - Duck, Donald<br>756-43-9876 - Mouse, Mickey<br>321-45-2345 - Pan, Peter<br>968-45-3641 - Bear, Baloo<br>862-93-8766 - Cricket, Jiminy<br>784-33-8893 - Wanderquack, Webby<br>783-45-6789 - Mary, Fairy<br>823-77-4545 - Horsecollar, Horace<br>576-29-9877 - Pistoles, Panchito<br>876-54-2887 - Rabitt, Roger<br>234-56-4564 - Elephant, Elmer<br>783-49-8932 - Khan, Shere<br>111-22-2333 - Pelekai, Lilo<br>123-67-4543 - Lollygagger, Gus<br>789-56-7856 - Bunny, Bugs<br>342-93-2712 - Robinson, Wilbur<br>987-65-4321 - Cow, Clarabelle |

| | |
|---|---|
| Choice: 2<br>123456789 | Duck, Donald<br>Division 0: 89<br>Division 1: 34<br>Division 2: 45<br>Division 3: 5 |
| Choice: 2<br>987654321 | Cow, Clarabelle<br>Division 0: 89<br>Division 1: 54<br>Division 2: 0<br>Division 3: 24 |
| Choice: 2<br>999887777 | SSN not in the list. |
| Choice: 2<br>876542887 | Rabitt, Roger<br>Division 0: 4<br>Division 1: 75<br>Division 2: 24<br>Division 3: 5 |
| Choice: 3<br>123456789 | Duck, Donald<br>Total votes: 173<br><br>(There should be a line right after this, before the “Press any key to continue…” message, even if it is not shown in the output.exe file provided previously.) |
| Choice: 3<br>987654321 | Cow, Clarabelle<br>Total votes: 167<br><br>(There should be a line right after this, before the “Press any key to continue…” message, even if it is not shown in the output.exe file provided previously.) |

| | |
|---|---|
| Choice: 3<br>999888777 | SSN not in the list. |
| Choice: 3<br>576299877 | Pistoles, Panchito<br>Total votes: 166<br><br>(There should be a line right after this, before the “Press any key to continue…” message, even if it is not shown in the output.exe file provided previously.) |
| Choice: 4 | Election winner: Cricket, Jiminy<br>Total votes: 278 |
| Choice: 5 | FINAL RESULTS<br>-------------<br>17  278 Cricket, Jiminy<br>16  233 Pelekai, Lilo<br>15  221 Mouse, Mickey<br>14  173 Duck, Donald<br>13  167 Cow, Clarabelle<br>12  166 Pistoles, Panchito<br>11  159 Mary, Fairy<br>10  135 Horsecollar, Horace<br>9  132 Elephant, Elmer<br>8  130 Khan, Shere<br>7  108 Rabitt, Roger<br>6  105 Bear, Baloo<br>5  101 Lollygagger, Gus<br>4  99  Robinson, Wilbur<br>3  94  Wanderquack, Webby<br>2  89  Bunny, Bugs<br>1  86  Pan, Peter<br><br>(Numbers to the left should be indented as shown in the output.exe provided previously.) |
| Choice: 6 | (exits program) |