**Project 1 (Part D): Testing**

**Before** you start testing your **project part C**, do the following:

1) **Main.cpp**: In the processSelection function there is a statement to format decimal places set to 1; change the format to 2 decimal places.
2) **DonorList:** The output was set with indentation to 4 spaces; change it to 5 spaces.

## PART A – ALPHA TESTING

When correcting, make sure you understand the errors, because there will be questions in the final that will be related to the project.

Name header format:

```
/*
        Team name
        Member name(s) (format as shown in the syllabus, one student per line)
        (blank line)
        CS A250
        (date => format as shown on the syllabus)
        (blank line)
        Project 1: Donor List
*/
```

| **MemberType.h** |
| --- |
| **Order** should be:<br><br>1. Name header (as shown on the syllabus)<br>2. (blank line)<br>3. #ifndef MEMBERTYPE_H<br>4. #define MEMBERTYPE_H<br>5. (blank line) |

6. All other #include statements (iostream and string only)
7. (blank line)
8. using (if including std:: in the code, there is NO need for using statement)
9. (blank line)
10. Class definition

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

1. Default constructor
2. Overloaded constructor
3. Function setMemberInfo
4. Function getFirstName
5. Function getLastName
6. Function getMembershipNo
7. Function printName
8. Function printMemberInfo
9. Destructor

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

Functions are separated by a line OR can be grouped by purpose of the function (all accessor functions together, all print functions together, etc.)

The following functions are **const** functions:

1. Function getFirstName
2. Function getLastName
3. Function getMembershipNo
4. Function printName
5. Function printMemberInfo

Member variables: A **string** storing a first name, a **string** storing a last name and an **int** storing a membership number.
The identifiers are **descriptive** and follow the **camelCase** convention.

**MemberType.cpp**

| |
|---|
| **Order** should be:<br><br>1. Name header (as shown on the syllabus)<br>2. (blank line)<br>3. #include “MemberType.h”<br>4. (blank line)<br>5. Member functions |
| No libraries included (it might include using namespace; IFF not included in h file). |
| **Functions** are in the same **order** shown in the **MemberType.h** file.<br>There is a **blank line** separating each function definition. |
| **Default constructor** initializes membership number to 0.<br>**Default constructor** does **NOT** initialize the first and last name variables to empty strings (**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 **setMemberInfo** initializes all three member variables to the given values passed by the parameters. |
| Function **getFirstName** has one statement only (return). |
| Function **getLastName** has one statement only (return). |
| Function **getMembershipNo** has one statement only (return). |
| Function **printName** prints the member’s name in the following format:<br>`Lastname, Firstname` |
| Function **printMemberInfo** prints the person’s information in the following format:<br>`membership# - Firstname Lastname` |
| **Destructor** is empty. |
| **DonorType.h** |

This class inherits from the **MemberType** class.

**Order** should be:

1. Name header (as shown on the syllabus)
2. (blank line)
3. #ifndef  DONORTYPE_H
4. #define DONORTYPE_H
5. (blank line)
6. #include “MemberType.h”
7. (blank line)
8. Other #include statements (iostream and string only)
9. (blank line)
10. using (if including std:: in the code, there is NO need for using statement)
11. (blank line)
12. Class definition (inherits from MemberType)

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

1. Default constructor
2. Overloaded constructor
3. Function setDonorInfo
4. Function setAmountDonated
5. Function getAmountDonated
6. Function printDonor
7. Function printDonation
8. Destructor

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

| |
|---|
| Functions are separated by a line OR can be grouped by purpose of the function (all accessor functions together, all print functions together, etc.) |
| The following functions are **const** functions:<br><br>1. Function getAmountDonated<br>2. Function printDonor<br>3. Function printDonation |
| **Member variable:** A **double** storing the donation amount.<br>The identifier is **descriptive** and follows the **camelCase** convention. |
| **DonorType.cpp** |
| **Order** should be:<br><br>1. Name header (as shown on the syllabus)<br>2. (blank line)<br>3. #include "DonorType.h"<br>4. (blank line)<br>5. Member functions |
| No libraries included (it might include using namespace; IFF not included in h file). |
| **Functions** are in the same **order** shown in the **DonorType.h** file.<br>There is a **blank line** separating each function definition. |
| **Default constructor** initializes donation to 0**.0** (**NOT** just 0). |
| Overloaded constructor initializes **ONLY** the donation to the amount passed. All other values for the remaining variables are sent to the parent class. How is this accomplished? **Look at lecture 3 (inheritance), slide 17.** |
| Function **setDonorInfo** calls the parent's function **setMemberInfo** (**without** using the class qualifier and scope resolution) to send all values, and then updates the amount donated. |

| |
|---|
| Function **setAmountDonated** has one statement only. |
| Function **getAmountDonated** has one statement only (return). |
| Function **printDonor** has one statement only, a call to the parent's function **printMemberInfo** (**without** using the class qualifier and scope resolution). |
| Function **printDonation** should either call<br>• parent's functions **getLastName and getFirstName** (**without** using the class qualifier and scope resolution)<br>**OR**<br>• Parent's function **printName** (**without** using the class qualifier and scope resolution).<br><br>The function does **NOT** format the decimal. |
| **Destructor** is empty. |
| **DonorList.h** |
| **Order** should be:<br><br>1. Name header (as shown on the syllabus)<br>2. (blank line)<br>3. #ifndef DONORLIST_H<br>4. #define DONORLIST_H<br>5. (blank line)<br>6. #include "DonorType.h"<br>7. (blank line)<br>8. Other #include statements (iostream and string only)<br>9. (blank line)<br>10. using (if including std:: in the code, there is NO need for using statement)<br>11. (blank line)<br>12. A constant that stores the capacity of 20 (identifier should be all capital letters).<br>13. (blank line)<br>14. Definition class DonorList |

| **Default Constructor** |
| --- |
| Overloaded constructor<br>- parameter: int (a new capacity) |
| Function **addDonor**<br>- is named correctly<br>- is a void function<br>- parameters: const string&, const string&, int, double |
| Function **getNumberOfDonors**<br>- is named correctly<br>- returns an int<br>- is a **const** function |
| Function **getTotalDonations**<br>- is named correctly<br>- returns a double<br>- is a **const** function |
| Function **getHighestDonation**<br>- is named correctly<br>- returns a double<br>- is a **const** function |
| Function **isEmpty**<br>- is named correctly<br>- returns a Boolean<br>- is a **const** function |
| Function **searchID**<br>- is named correctly<br>- parameter: int<br>- returns a Boolean |

| |
|---|
| - is a **const** function |
| Function **searchName**<br>- is named correctly<br>- parameter: const string&<br>- returns a Boolean<br>- is a **const** function |
| Function **deleteDonor**<br>- is named correctly<br>- is a void function<br>- parameter: int |
| Function **emptyList**<br>- is a void function |
| Function **printAllDonors**<br>- is named correctly<br>- is a void function<br>- is a **const** function |
| Function **printDonorByName**<br>- is named correctly<br>- is a void function<br>- is a **const** function<br>- parameter: const string& |
| Function **printDonor**<br>- is named correctly<br>- is a void function<br>- is a **const** function<br>- parameter: int |

| |
|---|
| Function **printDonation**<br>- is named correctly<br>- is a void function<br>- is a **const** function<br>- parameter: int |
| Function **printTotalDonations**<br>- is named correctly<br>- is a void function<br>- is a **const** function |
| Function **printHighestDonation**<br>- is named correctly<br>- is a void function<br>- is a **const** function |
| **Destructor** |
| Function **resizeList**<br>- is a **private** function<br>- is named correctly<br>- is a void function |
| **Member variables:** A pointer of type DonorList and two ints (as given). |
| All variable naming follows standards (camelCase) |
| **DonorList.cpp** |
| **Order** should be:<br><br>1. Name header (as shown on the syllabus)<br>2. (blank line)<br>3. #include “DonorList.h” |

| |
|---|
| 4. (blank line)<br>5. Member functions |
| No libraries included (*it might include using namespace; IFF not included in h file*). |
| **Default Constructor**<br>**-** sets numOfElem to zero<br>- sets capacity to CAP<br>- creates a new dynamic array of type **DonorType**, using pointer list. |
| **Overloaded Constructor**<br>**-** sets numOfElem to zero<br>- sets capacity to the given capacity passed by the parameter<br>- creates a new dynamic array of type **DonorType**, using pointer list. |
| Function **addDonor**<br>- creates a new object of **DonorType**, using the overloaded constructor of the **DonorType**.<br>- checks if the list is full, in which case calls the function **resizeList**.<br>- uses a WHILE statement to find the index where the object should be inserted (all objects are inserted in ascending order by membership) and STOPS when the index is found; shifting the objects can be done either with a WHILE loop or a FOR loop.<br>- increments the number of elements. |
| Function **getNumberOfDonors**<br>- returns the number of elements (one statement only). |
| Function **getTotalDonations**<br>- uses a loop to compute the total donations and then returns the amount.<br>- does NOT format the decimals. |
| Function **getHighestDonation**<br>- uses a loop to find the highest donation and then returns the amount.<br>- while loop contains if/else statement<br>- does NOT format the decimals. |

| |
|---|
| Function **isEmpty**<br>- returns true if the list is empty, false otherwise (one statement only). |
| Function **searchID**<br>- uses a **WHILE** loop to search the membership number and **STOPS** when the number is found.<br>- while loop contains if/else statement<br>- Check **WARNINGS**: Do all paths return a value? If you are not sure how to do this, then check this page:<br>https://msdn.microsoft.com/en-us/library/ms182026(v=vs.80).aspx |
| Function **searchName**<br>- uses a loop to a donor with the given last name and **STOPS** when it finds the first occurrence.<br>- while loop contains if/else statement<br>- Check **WARNINGS**: Do all paths return a value? If you are not sure how to do this, then check this page:<br>https://msdn.microsoft.com/en-us/library/ms182026(v=vs.80).aspx |
| Function **deleteDonor**<br>- uses a **WHILE** loop to find the index where the object is and STOPS when the index is found; shifting the objects can be done either with a WHILE loop or a FOR loop.<br>- decrements the number of elements. |
| Function **emptyList**<br>- empties the list (one statement only) |
| Function **printAllDonors**<br>- loops to print all donors by calling the **printDonor** function. |
| Function **printDonorByName**<br>- uses a loop to print one of more donors with the given last name.<br>- if the donor is not found, outputs the error message, "There are no donors with this last name."<br>- uses **cout** to print out the error message. |

| |
|---|
| Function **printDonor**<br>- uses a **WHILE** loop to find the membership number and **STOPS** when the number is found to call the function **printDonor**. |
| Function **printDonation**<br>- uses a **WHILE** loop to find the membership number and **STOPS** when the number is found to call the function **printDonation**. |
| Function **printTotalDonations**<br>- prints the total donations by calling the **getTotalDonations** function (one statement only). |
| Function **printHighestDonation**<br>- prints the highest donation by calling the **getHighestDonation** function (one statement only). |
| **Destructor**<br>- deletes the dynamic array<br>- resets the pointer list to NULL |
| Function **resizeList**<br>- creates a new pointer to point to a new dynamic array of type **DonorType** with double the existing capacity. You should **not** use the const CAP, but you should use the member variable **capacity**. Why?<br>- using a loop, copies all the objects from the array pointed by pointer list into the new dynamic array.<br>- resets the member variable capacity to the new capacity<br>- deletes the original array<br>- resets pointer list to point to the newly created array. |
| **OTHER** |
| The name header must be in the .h files, formatted as shown on the **syllabus** (check the syllabus if not sure). |
| There is **NO horizontal scrolling**. All statements are short enough. |
| There is a **space** before and after **operators** (i.e., a = b + c; instead of a=b+c;). |
| All **variables** have a **descriptive** identifier that uses the **camelCase** convention. |

| |
|---|
| Nowhere in the project there are two or more consecutive blank lines. |
| Proper indentation and spacing. |

## PART B – BETA TESTING

Test the program using the test cases given below and comparing the output with the given output, **final_output.exe.**

**PAY ATTENTION TO OUTPUT FORMATTING.** If you have errors you will need to fix **YOUR** code.

The project does **not** handle **all** exceptions, only a few. We will assume the user is paying attention and is typing what is required. **Please note**: This project will be graded also on correct and exact output, which means that spaces, lines, upper- and lower-cases need to match the given output.

| |
|---|
| **Selection 1** – Add the following donors (make sure you print all after adding to check if the donor was added):<br>• Niklaus Wirth 12121212 10000<br>• Jason Fried 98989898 20000<br>• Rasmus Lerdorf 11221122 30000<br>• John Resig 99889988 40000<br>• Brian Kernighan 44556677 50000 |
| **Selection 2** – Delete the following donors (make sure you print all after deleting to check if the donor was added):<br>• Rasmus Lerdorf 11221122<br>• John Resig 99889988<br>• Brian Kernighan 44556677<br>• (does not exist) 33443344 |
| **Selection 3** – Search these donors:<br>• Wirth<br>• Fried |

| |
|---|
| • Bohr<br>• Curie<br>• Resig |
| **Selection 4** – Search these IDs:<br>• 12121212<br>• 12345678<br>• 98989898<br>• 45454545<br>• 45674567 |
| **Selection 5** – Prints all donors. |
| **Selection 6** – Print donations from these donors:<br>• 12121212<br>• 12345678<br>• 98989898<br>• 45454545<br>• 45674567 |
| **Selection 7** – Prints total donations. |
| **Selection 8** – Print highest donations. |
| **Selection 11** – Not in the menu. |
| **Selection 9** – Exit with greeting. |