Project 1 (Part C): Donor List

For this part of the project, you will complete the class **DonorList** that creates an object containing a **pointer** that points to a **dynamic array** of type **DonorType** object (this is similar to the **DArray** class).

You will start by adding the following files to your **project part B:**

- **DonorList.h**
    - The class **DonorList** has **three (3) member variables**: a **pointer** to a **dynamic array** of **type DonorType**, an **int** storing the **capacity** of the array, and an **int** storing the **number of elements** in the array.
    - You will need to complete the **DonorList** class **definition** by writing the declarations for the functions listed below.

- **DonorList.cpp**
    - In this file, you will implement all the member functions needed for the **DonorList** class (instructions are below).

- **donors_data.txt**
    - This text file goes inside the **Resource Files** of your solution.
    - The file contains a list of donors that will be added to the list. Each line contains a first name, a last name, a membership number and the amount of the donation:

```
Maria Curie 12345678 10000.0
```

- **InputHandler.h**
    - Function **readDonorData**
        - It opens the text file and checks if the file is available and if the data can be read. If the file can be read, it will call function **createDonorList**; if the file cannot be read, the program will terminate (statement: **exit(1)**). Once all entries are read, the file will close.
    - Function **createDonorList**
        - It reads each item from the text file and **calls** the **function DonorList::addDonor** to add a new donor to the list.
    - Although the implementation of this file is complete, **do NOT dismiss it!** Pay careful attention to the functions that are implemented to understand how they work.

- **Main.cpp**
    - Function **displayMenu**
        - It displays a menu to interact with the user.
    - Function **processSelection**
        - Determines the user's selection and proceeds accordingly.
    - Do **NOT** modify the code in this file.

IMPORTANT:

- Your implementation **needs to agree with *my* implementation** (**not** the other way around).

- When implementing the functions below for the **DonorList** class, if possible, make a good use of functions that already exist in the **DonorType** and/or **MemberType** file.

- Do **NOT** include error message that are **NOT** listed below.

- Assume **IDs** are **unique** (you should **always** assume that there is a unique key in every database).

These are the functions you need to implement (write the functions **IN THE ORDER SHOWN BELOW**):

- **Default constructor**
    - Initializes the member variables of the class; it uses the **constant capacity** already declared.

- **Overloaded constructor**
    - **Parameter:** An **int** storing the **capacity** of the array.
    - Initializes the member variables of the class; it uses the **capacity** passed by the **parameter**.

- Function **addDonor**
    - **Parameters (in this order):** a **string** storing a first name, a **string** storing a last name, an **int** storing a membership number and a **double** storing the amount donated.
    - Creates an object of type **DonorType** and inserts it into the array **in ascending order by membership number**.
    - If the array is full, the function calls the **resizeList** function (see below).

- Function **getNumberOfDonors**
    - Returns the number of donors in the list.

- Function **getTotalDonations**
    - Returns the total amount of donations. (Do **not** format the amount; formatting will be handled in the Main.cpp file.)
    - **NO** need to check if the list is empty.

- **Function getHighestDonation**
    - Returns the highest donation. (Do **not** format the amount; formatting will be handled in the Main.cpp file.)
    - **NO** need to check if the list is empty.

- Function **isEmpty**
    - Returns true if the list is empty, false otherwise.

- Function **searchID**

    - **Parameters:** An **int** storing a membership number.
    - Traverses the list and returns **true** if the donor is in the list, **false** otherwise.
    - Use a **while** loop and make sure you **stop the search as soon as you find the donor**.
    - **NO** need to check if the list is empty.

- Function **searchName**
    - **Parameters:** A **string** storing a last name.
    - Traverses the list and returns **true** if the donor is in the list, **false** otherwise.
    - **NO** need to check if the list is empty.
-
- Function **deleteDonor**
    - **Parameters:** An **int** storing a membership number.
    - Deletes the donor with the given membership from the list.
    - Use a **while** loop and make sure you **stop the search as soon as you find the donor**.
    - **NO** need to check if the list is empty.

- Function **emptyList**
    - Re-sets the list to an empty list.

- Function **printAllDonors**
    - Calls the function **printDonor** to print all the donors in the list.
    - For the expected format, check the **output.exe** file given.

- Function **printDonorByName**
    - **Parameters:** A **string** storing a last name.
    - Searches for the donor and uses the function **printDonor** to print.
    - If the donor is not found, outputs the message "There are no donors with this last name."
    - For the expected format, check the **output.exe** file given.

- Function **printDonor**
    - **Parameters:** An **int** storing a membership number.
    - Searches for the donor and uses the function **printDonor** to print.
    - Use a **while** loop and make sure you **stop the search as soon as you find the donor**.
    - For the expected format, check the **output.exe** file given.

- Function **printDonation**
    - **Parameters:** An **int** storing a membership number.
    - Searches for the donor and uses the function **printDonation** to print.
    - Use a **while** loop and make sure you **stop the search as soon as you find the donor**.
    - For the expected format, check the **output.exe** file given.

- Function **printTotalDonations**
    - Calls function **getTotalDonations** to print.

    - For the expected format, check the **output.exe** file given.

- Function **printHighestDonation**
    - Calls function **getHighestDonation** to print.
    - For the expected format, check the **output.exe** file given.

- **Destructor**
    - Deletes all dynamic data.

- Function **resizeList**
    - This is a **private** function.
    - Re-creates a new array of twice the capacity and copies all the objects into the new array. Make sure you handle memory correctly.

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.

Testing cases to try:

- **Selection 1** – Add the following donors (make sure you print all after adding to check if the donor was added):
    - Niklaus Wirth 12121212 10000
    - Jason Fried 98989898 20000
    - Rasmus Lerdorf 11221122 30000
    - John Resig 99889988 40000
    - Brian Kernighan 44556677 50000
- **Selection 2** – Delete the following donors (make sure you print all after deleting to check if the donor was added):
    - Rasmus Lerdorf 11221122
    - John Resig 99889988
    - Brian Kernighan 44556677
    - (does not exist) 33443344
- **Selection 3** – Search these donors:
    - Wirth
    - Fried
    - Bohr
    - Curie
    - Resig
- **Selection 4** – Search these IDs:
    - 12121212
    - 12345678
    - 98989898

    - 45454545
    - 45674567
- **Selection 5** – Prints all donors.
- **Selection 6** – Print donations from these donors:
    - 12121212
    - 12345678
    - 98989898
    - 45454545
    - 45674567
- **Selection 7** – Prints total donations.
- **Selection 8** – Print highest donations.
- **Selection 11** – Not in the menu.
- **Selection 9** – Exit with greeting.