Instructions

1- Students will form teams of **2** students **from the same group.**
2- Deadline of submission is **Friday Dec.24th at 11:55 pm.**
3- Submission will be on Blackboard.
4- No late submission is allowed.
5- No submission through e-mails.
6- You will write a cpp file with the name GroupNum_firstStudentID_SecondStudentID.cpp . **No folders or zip files are allowed**. Zip submissions will receive a **zero**. Students who attend the same lab slot can form a group like S7 and S8. In this case name the file with any of the 2 groups (S7 or S8 for ex.)

   The group number CAN ONLY BE from S1 to S40.
   For example if you have an ID 20202020 and in group S1 and your partner has the ID 19191919 and in group S2, then your cpp file will have the name S1_20202020_19191919.cpp . The character separating the group name and the ids is an Underscore character. Do not use any other characters like a space, a dash, or a bracket,…etc.

7- **In case of Cheating you will get a negative grade whether you give the code to someone, take the code from someone/internet, or even send it to someone for any reason.**
8- You have to write clean code and follow a good coding style including choosing meaningful variable names.

Task

Declare and implement 5 classes: FloatArray, SortedArray, FrontArray, PositiveArray & NegativeArray.
1- The FloatArray class stores a dynamic array of floats and its size. It has:

- A parameterized constructor that takes the array size.
- An add method that adds a float at the end of the array.
- Overloading for the insertion operator << to write the array to a

file (ofstream)
- Overloading for the extraction operator >> to read the array elements from the file (ifstream) and add them to the array.
- A destructor to deallocate the array

2- The SortedArray inherits from FloatArray. It has:

- A parameterized constructor that takes the array size.
- An add method that adds a float at the right place in the array such that the array remains sorted with every add. Don't add to the array then sort but rather add in the right place.

3- The FrontArray inherits from FloatArray. It has:

- A parameterized constructor that takes the array size.
- An add method that adds a float at the front of the array.

4- The PositiveArray that inherits from SortedArray. It has:

- A parameterized constructor that takes the array size.
- An add method that adds a float to the array only if it's a positive number. It then uses the add method of SortedArray.

5- The NegativeArray that inherits from SortedArray. It has:

- A parameterized constructor that takes the array size.
- An add method that adds a float to the array only if it's a negative number. It then uses the add method of SortedArray.

The only input to your program is the names of the input txt file and output txt file name. The input text file will have exactly 10 lines. Each line will have:

1- **One of 5 words: Array, Front, Sorted, Positive, or Negative. This indicates the type of array to create.**

2- **An integer which indicates the number of elements in the array.**

3- **The elements to be entered into the array.**

The following is an example content of the file:

Sorted 10	8.4	-4	2.3	11	80	7	77	95	12	100
Array 7	3.4	2	0	9	4.7	3	9			
Front 5	8	4	7.9	0.44	1					
Array 6	45	23	8.5	3.98	4	2.5				
Sorted 4	90	6	4	111						
Sorted 8	7	3	4	0	1.1	3.2	88	9		
Array 6	13.5	7.6	9	33	1	0				
Front 3	7	5	1.5							
Positive 7	7.9	-1	-44	9.7	4.3	-1	0			
Negative 8	88	-1.5	-40	-9.7	4.3	13	0	-11		

You should use polymorphism in your code by creating an array of FloatArray* in main.

Read from the txt file and allocate your objects according to the type of array and fill the arrays using the extraction operator >>.
After finishing reading the file, you should write your arrays to another text file using the insertion operator<<.

The output file for the above text file will be:

10|	-4	2.3	7	8.4	11	12	77	80	95	100
7|	3.4	2	0	9	4.7	3	9			
5|	1	0.44	7.9	4	8					
6|	45	23	8.5	3.98	4	2.5				
4|	4	6	90	111						
8|	0	1.1	3	3.2	4	7	9	88		
6|	13.5	7.6	9	33	1	0				
3|	1.5	5	7							
3|	4.3	7.9	9.7							
4|	-40	-11	-9.7	-1.5						

Where each line begins with the number of elements followed by the character '|', a tab, then the array elements separated by tabs.

Don't forgot to deallocate by deleting your objects in main after finishing.

A sample run is found below:

```
Enter the name of the input file:
in.txt
Enter the name of the output file:
MyOut.txt

Process returned 0 (0x0)   execution time : 16.048 s
Press any key to continue.
```

Don't take any other inputs from the user. Otherwise you will lose grades.