CS 201, Fall 2018

Homework 3

DUE: November 30, Friday @23:55

Description: In this assignment, you will write a C++ program that **finds the kth largest number** among a set of N numbers. However, this time, you will use a **heap** structure to implement the program. The details can be found below.

1) The Input and Expected Output

The input and the expected output of the program is the same as described in Homework 1. The program will take the **type of algorithm** to be applied, ***k*** (a number less than or equal to *N*). Then it will take ***N*** followed by a list of ***N* numbers**. As output, it will print out the ***kth* largest number** and the **total elapsed time** for the completion of the algorithm.

As the only difference from Homework 1, **the algorithm type** can be 1, 2, or **3**. You can modify and reuse the test generator program to generate test inputs. You can also just modify the sample test inputs for Homework 1 to test the same set of numbers with algorithm type being 3.

Your program will be tested only with algorithm type 3.

2) The Algorithm

The new algorithm will make use of a **heap** structure to find the kth largest number in *O(NlogN)* time, where *N* is the total amount of numbers. The outline of the algorithm is as follows.

- build a *min heap* with the first *k* numbers (i.e., k times *insert*)
- then, compare the remaining *N-k* numbers with the *root* node (i.e., min. element)
  - if a number is smaller than the *root*, ignore the number
  - else, *deleteMin* (remove the *root*) and *insert* the new number
- return the *root* element

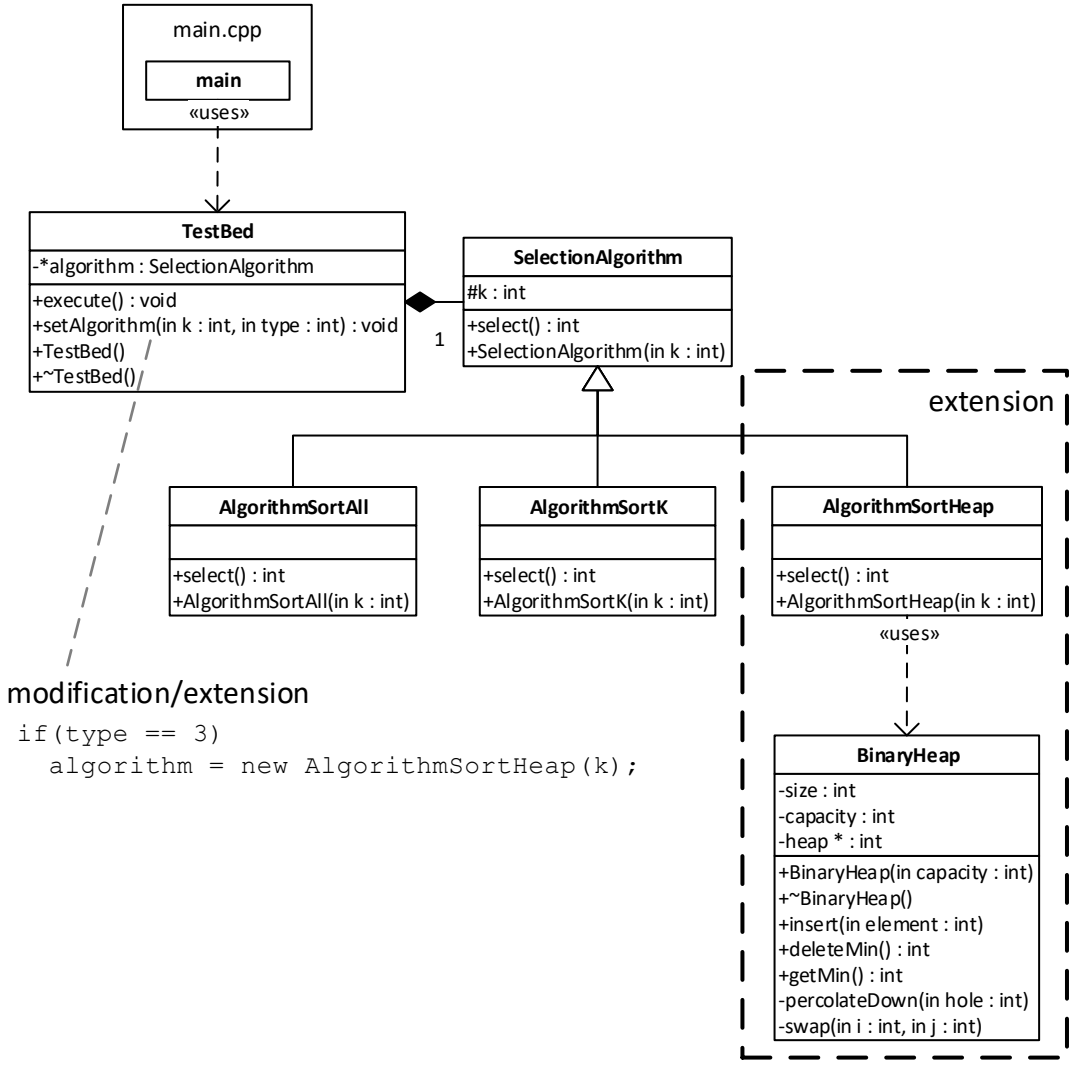
3) Heap Implementation

Before implementing your algorithm, you will need to implement a **binary min heap** that will be used by the algorithm. You will be provided with 3 files: *i)* "BinaryHeap.h", *ii)* "BinaryHeap.cpp", *iii)* "heaptest.cpp". The first file includes a header file, which lists the member attributes and functions of the heap. The second file should be completed by you to provide the implementation of the listed functions. In this homework, we will employ an **array-based** binary heap implementation with a fixed capacity. The third file contains the main method, which tests the implementation of the class.

As the first step, you need to complete the implementation in the "BinaryHeap.cpp" file, compile the 3 files together, execute the program, and observe that all the tests pass. Once you make sure that your heap implementation is correct, you can copy the files "BinaryHeap.h" and "BinaryHeap.cpp" to your project created for Homework 1 and extend it as described below.

4) The (Extended) Design

You can reuse the design of Homework 1. You just need to add two extra classes: *i) AlgorithmSortHeap*, which extends from the *SelectionAlgorithm* class and overwrites the *select* method to implement the new algorithm, and *ii) BinaryHeap* class, which implements a heap structure. The *select* method within the *AlgorithmSortHeap* class should make use of the *BinaryHeap* to create and manipulate (insert/delete items on) a heap object. The overall design is depicted below.



You might need to make further small modifications in function *main* and the *TestBed* class to accept 3 as the algorithm type. This is because, previously the algorithm type was assumed to be either 1 or 2. For instance, you should make a modification/extension within the *setAlgorithm* method of the *TestBed* class. If the *type* argument is 3, an object of type *AlgorithmSortHeap* should be assigned to the *algorithm* member variable. The rest of the design and implementation can be reused as is.

If you have not submitted Homework 1, then you have to implement at least the *main* method, the *TestBed* class and the *SelectionAlgorithm* class as described in the assignment description of Homework 1.

5) Submission

You will submit this homework via the LMS system. You should follow the file-naming conventions and guidelines below.

- You should submit your source files as a **ZIP** archive file (**NOT** RAR or other formats). The name of the file should be in format "**<USER-ID>_hw<HOMEWORK-NR>.zip**". For example, if your username is vy1043, then the name of the submitted file should be "vy1043_hw4.zip". Pay attention that all the letters are in lower-case. ZIP archive is supposed to contain **just the source files**, no folders are allowed by any means.
- The contents of the ZIP file should be as follows:
  - **main.cpp** (includes the *main* function)
  - **TestBed.h** (TestBed class definition)
  - **TestBed.cpp** (TestBed class implementation)
  - **SelectionAlgorithm.h** (SelectionAlgorithm class definition)
  - **SelectionAlgorithm.cpp** (SelectionAlgorithm class implementation)
  - **AlgorithmSortHeap.h** (AlgorithmSortHeap class definition)
  - **AlgorithmSortHeap.cpp** (AlgorithmSortHeap class implementation)
  - **BinaryHeap.h** (BinaryHeap class definition)
  - **BinaryHeap.cpp** (BinaryHeap class implementation)
  - **AlgorithmSortK.h** (AlgorithmSortK class definition)
  - **AlgorithmSortK.cpp** (AlgorithmSortK class implementation)
  - **AlgorithmSortAll.h** (AlgorithmSortAll class definition)
  - **AlgorithmSortAll.cpp** (AlgorithmSortAll class implementation)
- Late submissions and C++ files that do not compile are **not** accepted.
- You can resubmit your homework (until the deadline) if you need to.
- Make sure that your program does **not** include commands specific to a development environment, e.g., `system("pause")` or `#pragma once` in Visual Studio.

*** Optional files:** The last 4 files are to be submitted if implemented as part of Homework 1. These algorithms will not be tested for evaluating Homework 3. The algorithm type will always be set as 3 in the test cases.