CS2400 Project 4

Total points: 100

Purpose:

1. Implement a binary heap using array representation.
2. Understand the time complexity of heap operations through experiments.

Task Description:

In this project, you are going to build a **max-heap** using array representation. In particular, your program should:

- **Implement** two methods of building a max-heap.
  - **Using sequential insertions** (its time complexity: $O(nlogn)$, by successively applying the regular add method).
  - **Using the optimal method** (its time complexity: $O(n)$, the "smart" way we learned in class).

  For both methods, your implementations need to keep track of how many swaps (swapping parent and child) are required to build a heap.

- **Implement** the remove method of a max-heap.

- **Read** a sequence of integers **from an input file**.
  - "data.txt": This file contains 100 integers (no duplicates, and positive numbers). Each line is an integer.

- **Perform heap operations** and **Write** the results **into an output file**.
  - Create a max-heap using the **sequential insertions**, for those 100 integers.
  - Output the first 10 integers of your array, into the output file
  - Output the number of swaps performed, into the output file
  - Perform 10 removals on the heap
  - Output the first 10 integers of the resulting array, into the output file

  - Create a max-heap using **the optimal method**, for those 100 integers
  - Output the first 10 integers of your array, into the output file
  - Output the number of swaps performed, into the output file
  - Perform 10 removals on the heap
  - Output the first 10 integers in the resulting array, into the output file

The output file should use the format as shown below:

==========

Heap built using sequential insertions: 100,94,99,77,93,98,61,68,76,84,...
Number of swaps in the heap creation: 480
Heap after 10 removals: 90,89,62,77,88,53,61,68,76,84,...

Heap built using optimal method: 100,95,99,79,94,98,63,71,78,87,...
Number of swaps in the heap creation: 96
Heap after 10 removals: 90,89,63,79,88,55,62,71,78,87,...
==========

This project will be graded based on the quality of your program. **Java interface and generic data types are NOT required** in this project, but bonus points will be considered for those who use Java interface and generic data type.

What to Submit?

1. Source code
2. Input file (just the given "data.txt")
3. Output file
4. README.md describes anything that may help our grader to test your code (Full names/contributions of your teammates). If you used Java interface and generic data types or implemented extra features, please mention it in this file.
5. Submit a link to your GitHub repo, which belong to your group's GitHub organization, of which "2404s21" and "arahman7552" are members.