Université d'Ottawa
Faculté de génie

École d'ingénierie et de
technologie de l'information

uOttawa
L'Université canadienne
Canada's university

University of Ottawa
Faculty of Engineering

School of Information
Technology and Engineering

CSI2110 (Winter 2021)

Programming Assignment 1 (15%) – 100 points

Prof. WonSook Lee

Please read the document for "Double-ended Priority Queue - Diamond Min-Max-heap Algorithm". It is an extension of a heap you have learned in the class.

Please produce your data structure to allow maximum 1000 input keys. Your program will be tested using several test files and operations.

Input file format

A text file will be given as input. There are a few key words such as construct, insert, removeMin, removeMax

- "construct" comes with a number n which is total number of input keys and following keys in the next line.
  - Initialize your data structure (array)
  - You perform bottomUpConstruction() of a Diamond Min-Max heap with these keys.
  - first n/2 keys are used for min-heap and second n/2 keys are used for max-heap construction
  - if n is odd, the last key is used for buffer
  - **Check the doc "Double-ended Priority Queue - Diamond Min-Max-heap Algorithm"**
    - 6 15 9 10 18 9 1 14 13 11
      - will create the heap shown in first figure in Figure 13
    - 6 15 9 10 18 9 1 14 13 11 3
      - will create the heap shown in first figure in Figure 13 and a buffer containing 3
- "insert" comes with a number which is a key to be inserted to the heap.
  - You perform InsertItem(x)
- "removeMin"
  - You perform RemoveMin()
  -
- "removeMax"
  - You perform RemoveMax()

Output file format

Your program must simply produces as output a textfile. If the input file is called ABC.txt then the output file must be called result_ABC.txt.
It shows the output of removeMin/removeMax and/or content of min heap, max heap and buffer.
Check the example below.

---

e.g. of an input file (see Figure 13 in the doc "Double-ended Priority Queue - Diamond Min-Max-heap Algorithm")

construct 10
6 15 9 10 18 9 1 14 13 11

insert 43

removeMin

removeMax

e.g. of an output file format with the above input file

max-heap 18 15 14 13 11
min-heap 1 6 9 9 10
buffer

max-heap 18 15 14 13 11
min-heap 1 6 9 9 10
buffer 43

min-output 1
max-heap 43 18 14 15 11
min-heap 6 9 9 13 10
buffer

max-output 43
max-heap 18 15 14 13
min-heap 6 9 9 11
buffer 10

Requirement

- You must follow the format given in max-min_heap.txt for coding your class
- You must have four functions called
  - heapConstruction
  - removeMin
  - removeMax
  - insertItem
- For the implementation of these functions, you are free to use the standard Java classes, the implementations given in the book or in the class notes or using your own implementations.
- Your class must have an initialization step that performs all initialization steps as described above.
- Your class must have a read file step that reads the data and operations from the file
- Your class must have an execute() method that perform the algorithm as described in the doc.
- Your class must have a save(filename) method that saves the results in text file as described.
- The main method of this class simply asks for a filename.
- All your Java files must have a header that includes your name and student number.
- Your Java files must be appropriately commented. Include all your Java files in a zip file called projectCSI2110_XXX.zip where XXX is your student number. Include all files required for your program to compile.
- Include also in your zip file the output files showing the solutions for all input files provided.
- Your program will be tested with several new input files by a TA to check the correctness.

Marking Schene

Correctness of solutions provided: 45%
Quality of programming: 15%
Initialize and save method: 10%
Execute method: 20%
Abstract data types used: 10%