Eidgenössische
Technische Hochschule
Zürich

Ecole polytechnique fédérale de Zurich
Politecnico federale di Zurigo
Federal Institute of Technology at Zurich

Markus Püschel
Peter Widmayer
Thomas Tschager
Tobias Pröger
Tomáš Gavenčiak

Department Informatik
11th January 2017

Algorithmen & Datenstrukturen Trial Examination AS 16

This is the first trial exam task. The results of the trial exam have no weight on the final grade. The final exam will allow more time and contain two practical tasks (plus theory questions). You can find the second trial exam task as PDF in the `documentation` *folder or online afterwards. We recommend to try it at home (possibly with a timer).*

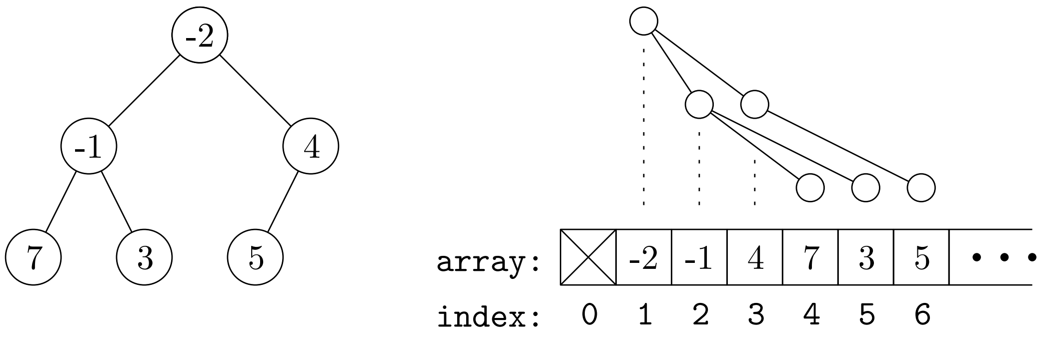
Minheap implementation

Your task is to complete an implementation of a binary *min-heap* using an array. The following operations should be supported:

- `Insert(`v`)` inserts the element v into the heap and restores the heap property.
- `ExtractMin` extracts the minimum from the heap and returns it, restoring the heap property.
- `QueryLast` returns the element that is stored in the last position of the part of the array that stores the heap.

If, e.g., the numbers $4, 7, -1, 3, -2, 5$ are inserted (in this order) into an initially empty heap, then we obtain the following heap and its array representation. The figure also shows the mapping from the heap tree to the array elements.

The heap array in the given template uses indexes starting from 1 to make calculating indexes easier, so element 0 is unused, and the heap may not fill the array until the end. For example, the `QueryLast` operation returns 5 for the heap below.



Given n integers $v_1, v_2, \ldots, v_n$, the program should start with an empty heap and perform operations `Insert(`v_i`)`, each followed by a `QueryLast` for all $i = 1, \ldots, n$, outputting the result of every `QueryLast`. Afterwards, the program should perform the operation `ExtractMin` n times, outputting every result.

The entire sequence of operations is therefore: `Insert(`v_1`), QueryLast, Insert(`v_2`), QueryLast, ..., Insert(`v_n`), QueryLast, ExtractMin, ..., ExtractMin` until the heap is empty.

Note that most of the code is already provided in the template – your task is to complete the missing parts of `Insert` and `ExtractMin`.

Input The first line of the input contains only the number of test cases.

Each test is specified on a single line and consists of one integer $1 \leq n \leq 1000$, the number of inserts, followed by n integers $v_1, v_2, \ldots, v_n$ to be inserted, each $-1000 \leq v_i \leq 1000$. Note that the numbers may repeat and multiple identical values must be all preserved by the heap.

Output For every test instance, output two lines: The first line contains the output of the n `QueryLast` operations, separated by spaces. The second line contains the output of the n `ExtractMin` operations, separated by spaces.

Example

Input (the first case same as above):

```
2
6 4 7 -1 3 -2 5
5 5 7 3 4 2
```

Output:

```
4 7 4 7 3 5
-2 -1 3 4 5 7
5 7 5 7 4
2 3 4 5 7
```

Grading You may get up to 100 judge points. Your program should implement every operation in time $\mathcal{O}(\log n)$ to get full points.

Submit your `Main.java` at `https://judge.inf.ethz.ch/team/websubmit.php?cid=18986&problem=DNA14_5`, enroll password is "`testitwell`". The judge will be open for submissions of this task at least until the final exam, so you may continue solving it later.

Notes For this exercise, we provide a program template as an Eclipse project in your workspace. This program already implements most of the functionality. Your task is to complete the code of the functions `MinHeap.insert()` and `MinHeap.extractMin()`.

As usual, the project also contains data for your local testing and `Judge.java` program that runs your `Main.java` on all the local tests – just open and run `Judge.java` in the project. The local test data are of course different than the data that are used in the online judge.

In this exam, you *must not* `import` any Java libraries except for `java.util.Scanner`. This is not checked on submission but will be checked afterwards and penalized if your code `imports` other libraries.