National Yang Ming Chiao Tung University
Department of Electrical Engineering
Data Structures, Fall 2021

November 7, 2021
Instructor: Mark P.-H. Lin
TA: Audis Yang

Programming Assignment #4

Trees

1 Problem Description

A max heap is a complete binary tree, in which the key value in each node is larger than the key values in its children, and the highest priority element is always stored at the root. It is one maximally efficient implementation of an abstract data type called a priority queue.

In this programming assignment, you are asked to implement the following operations of a max heap, including *add* and *delete*. In addition to those operations, you are also asked to output the max heap based on the *level-order*, *in-order*, *pre-order*, and *post-order* traversals.

2 Input Format

The input file gives the description of all the aforementioned operations. To simply the problem, each element in the max heap only contains an integer key. The detailed input format of each operation is given below.

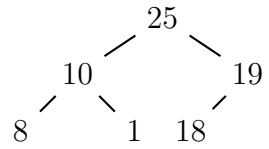
- **Add element** [Operation ID: 0]
  The operation has the format "0 [key]", which indicates that you will need to add a new element to your max heap. An example of adding a sequence of six elements and the resulting max heap is given below.

Sample input:

```
0 8
0 1
0 10
0 19
0 25
0 18
```

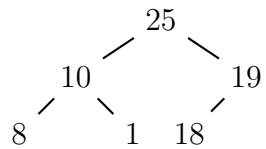
The corresponding array implementation and resulting max heap:

0	1	2	3	4	5	6	← array index
	25	10	19	8	1	18	← key of each element

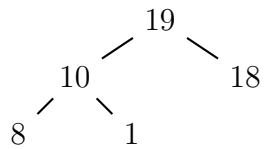


- **Delete element** [Operation ID: 1]
  The operation has the format "1", which indicates that you will need to delete the element with the maximum key from your max heap. The resulting heap must satisfy the max heap definition after the delete operation.

Before delete operation :



After delete operation :



- **Output max heap with level-order traversal** [Operation ID: 2]
  The operation has the format "2", which indicates that you will need to output the max heap at that moment with the level-order traversal.
- **Output max heap with pre-order traversal** [Operation ID: 3]
  The operation has the format "3", which indicates that you will need to output the max heap at that moment with the pre-order traversal.
- **Output max heap with in-order traversal** [Operation ID: 4]
  The operation has the format "4", which indicates that you will need to output the max heap at that moment with the in-order traversal.
- **Output max heap with post-order traversal** [Operation ID: 5]
  The operation has the format "5", which indicates that you will need to output the max heap at that moment with the post-order traversal.

3 Output Format

After performing the sequence the operations given in the input file, "sample.in", you will need to generate the output file, "sample.out", resulting from the operations in "sample.in". For each output operation, you will need to output the keys of all elements in the max heap with a new line in the following format: "key1 key2 key3 ...". An example of the sample input and output is given below.

sample.in

```
0 8
0 1
0 10
0 19
0 25
0 18
2
1
4
```

sample.out

```
25 10 19 8 1 18
8 10 1 19 18
```

It should be noted that the above output might not be unique. You may generate a different output file, which is also correct based on the max heap definition.

4 Command-line Parameter

In order to correctly test your program, you are asked to add the following command-line parameters to your program.

[executable file name] [input file name] [output file name]

(e.g., StudentID.exe sample.in sample.out)

5 Submission Information

Your program must be written in the C/C++ language, and can be compiled on the Linux platform. The source files of your program must be named with "[your student ID].h" and "[your student ID].cpp". The executable file name of your program must be "[your student ID].exe". To submit your program, please archive both executable and source files of your program into a single zip file, named "[your student ID].zip", and upload to E3.

6 Due Date

The zip file must be submitted through E3 before 23:59, November 21, 2021.

7 Grading Policy

The programming assignment will be graded based on the following rules:

- Pass sample input with compilable source code (50%)
- Pass five hidden test cases (50%)

The submitted source codes, which are copied from or copied by others, will NOT be graded. There will be 25% penalty per day for late submission.