Xi'an Jiaotong-Liverpool University

西交利物浦大学

PAPER CODE	EXAMINER	DEPARTMENT	TEL
CSE104		Computer Science and Software Engineering	

2nd SEMESTER 2017/18 FINAL EXAMINATIONS

BACHELOR DEGREE – Year 2

DATA STRUCTURES AND ALGORITHMS

TIME ALLOWED: 2 Hours

INSTRUCTIONS TO CANDIDATES

1、 **Total marks available are 100. This will count for 80% in the final assessment.**

2、 **Answer all questions.**

3、 **Calculator is not allowed.**

4、 **Answers should be written in the answer booklet(s) provided.**

5、 **Only answers in English are accepted.**

THIS PAPER MUST NOT BE REMOVED FROM THE EXAM HALL.

Part II. (25marks) Answer the following Question.

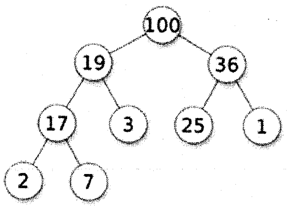
31. There is a tree structure with the following characteristics:

1. The tree structure is a *complete binary tree*.
   A complete binary tree is a binary **tree** in which every level, except possibly the last, is completely filled, and all nodes are as far left as possible.

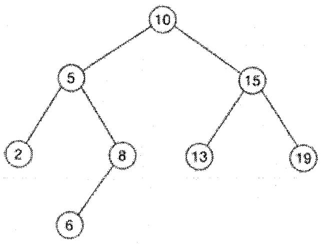
2. The value of a parent node must be bigger or equal to the child nodes.

A tree structure such as the above is called a **Binary Heap**.

See fig.a and fig.b for examples.



a) binary heap



b) not a binary heap

You are assigned to use a binary heap structure to implement **Priority Queue**. The key is the priority level of the element. The element with higher key has higher priority level, and should pop out the queue first.

(a) **(6 marks)** Why is it more convenient to use an array to store the binary heap? Please give two reasons. Please make sure each reason is no longer than two sentences.
(hint: a binary heap is a complete tree)

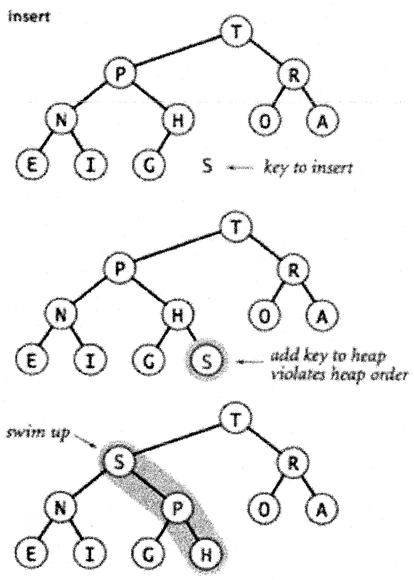
(b) **(2 marks)** Where (what position) should the element with highest priority stored?

(c) **(14 marks)** Please develop a method to delete the element with highest priority from your priority queue. See below for an example for inserting an element to priority queue: use a similar idea to develop your method.

(Your solutions should not exceed 8 lines of pseudo code and/or explanation)

Example

To insert *S* into the priority queue, first we add *S* to the end of the tree which violates the heap order (parent's key >= children's key), and then compare the parent key and swim up *S* to the correct position to maintain the heap order.



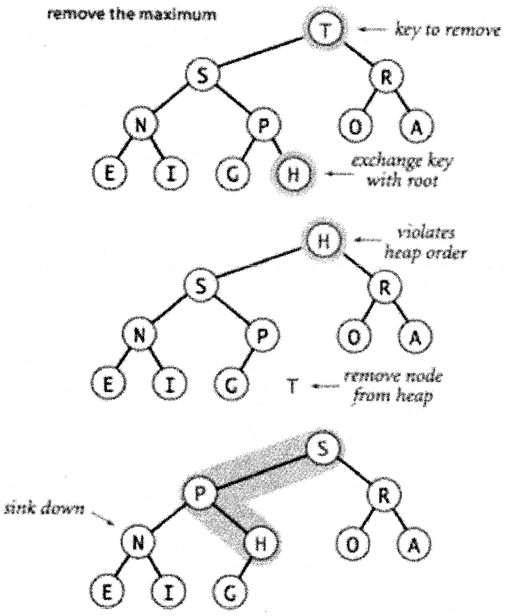
```
insertNewElement {
//insert element to last position which keeps the complete tree structure;
//now the structure may not be heap anymore as it may violate heap order.

        while (inserted element's key > parent's key){
                swap the element with the parent;          //swim up until
to the correct order
        }
}
```

Please complete deleteElement method:

deleteElement{

}



(d) **(3 marks)** What if some elements have the same priority level? How do you improve your structure of the queue? Please answer this question briefly, max 2-3 sentences.

END OF PAPER