

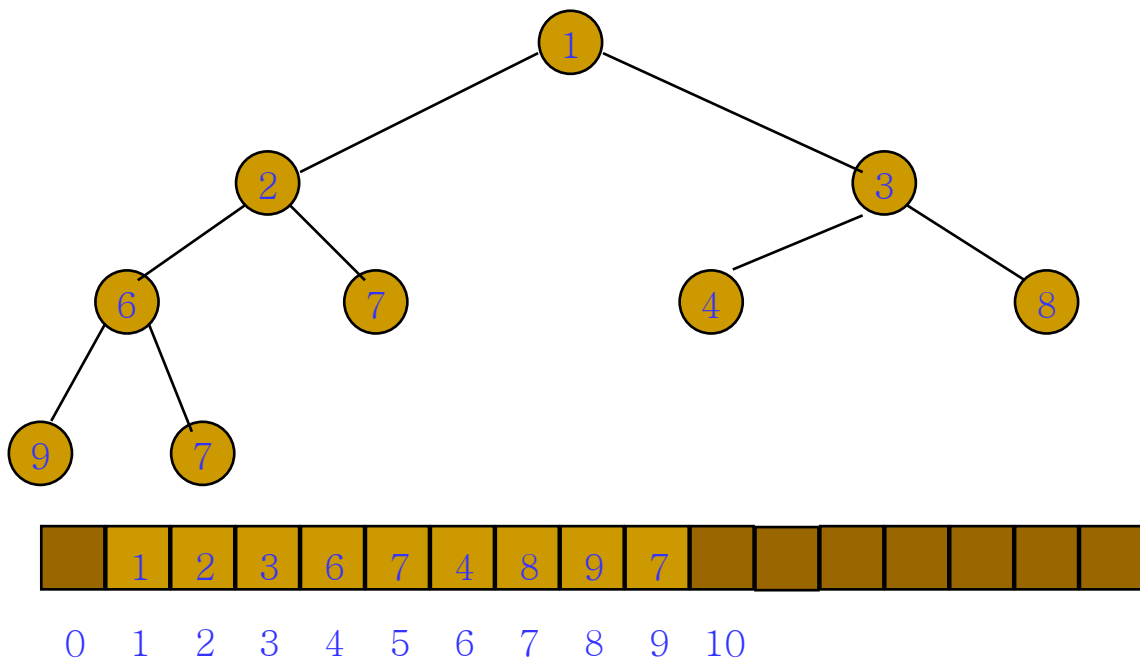
자료 구조 Lab 006 :

Lab18006.zip : LabTest.java, lab006.java, lab.in, lab.out, lab006.pdf

제출

lab006.java 를 학번.java 로 변경하여 이 파일 한 개만 제출할 것.

다음은 배열을 이용하여 min heap을 구현하는 내용이다. 클래스 MinHeap은 아래 그림과 같이 complete binary tree의 형태를 가지고 있는데, complete binary tree이기 때문에 **배열**로 표현하는 것이 적절하다. Min Heap의 조건은 각 노드를 기준으로 자신을 root로 하는 서브 트리의 모든 원소보다도 작아야 한다. 아래 그림은 그 한 예이다.



이러한 자료구조를 이용하여 간단한 프로그램을 작성한다.
수행 예는 다음과 같다.

```
선택 sanghwan@PC: ~/dbox/classes201/ds/lab20/lab20006
sanghwan@PC:~/dbox/classes201/ds/lab20/lab20006$ java LabTest
MeanHeap >
ins 10
CMD : ins 10
Min Heap : - 10

MeanHeap >
ins 5
CMD : ins 5
Min Heap : - 5 10

MeanHeap >
ins 15
CMD : ins 15
Min Heap : - 5 10 15

MeanHeap >
ins 19
CMD : ins 19
Min Heap : - 5 10 15 19

MeanHeap >
ins 4
CMD : ins 4
Min Heap : - 4 5 15 19 10

MeanHeap >
post 2
CMD : post 2 : 19 10 5
Min Heap : - 4 5 15 19 10

MeanHeap >
```

사용자가 사용하는 명령어의 syntax는 다음과 같다. Main() 함수에 정의되어 있다.

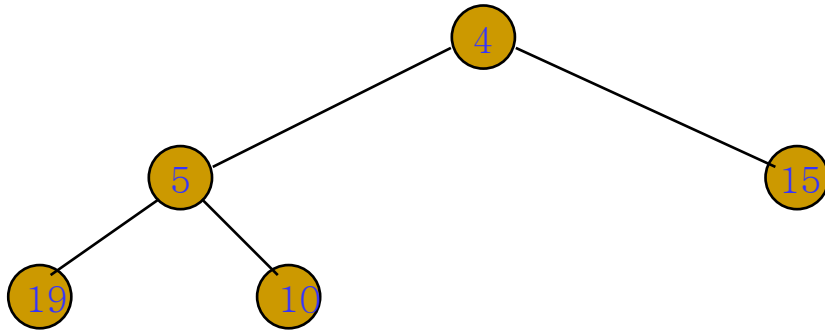
- ins integervalue

정수인 integervalue 값을 가지는 노드를 MinHeap에 삽입한다.

- post index

index가 가리키는 노드를 root로 하는 subtree를 postorder 로 방문(traverse) 한다. 방문해서 하는 작업은 방문한 노드의 값을 출력하는 것이다.

Min Heap : 으로 시작하는 라인은 현재 배열의 상태를 보여준다. 배열의 0번 원소는 사용하지 않기 때문에 -로 표현하였고, 배열 1번의 원소가 가장 작은 값을 확인한다. 예를 들면 위 예제의 Min Heap : - 4 5 15 19 10 은 다음과 같이 트리 형태로 표현 가능하다. 이미 구현되어 있으니 따로 구현할 필요는 없다.



따라서 post 2 명령에 의해 노드 5를 root로 하는 서브트리를 post order 로 방문하면 19 10 5 가 출력되어야 한다.

이 내용을 구현하기 위해 다음 두 개의 함수를 구현해야 한다.

- `void Insert (T e);`

MinHeap에 새로운 값 e를 삽입한다. 수업시간에 배운 MaxHeap 알고리즘을 상기하여 구현한다.

한 가지 주의할 점은 T 타입의 원소를 비교할 때 `>`, `<`, `==` 등을 사용할 수 없다.

이 경우에는 Comparable Interface에서 제공하는 `compareTo()`를 사용하면 된다.

예를 들면 `if(heapArray[1] > e)` 와 같은 비교를 하고 싶다면,

`if(heapArray[1].compareTo(e) > 0)`와 같이 쓰면 된다.

MaxHeap의 C++ 코드가 교과서에 구현되어 있기 때문에 이를 잘 응용하면 쉽게 구현할 수 있다.

- `void PostOrder (final int idx);`

idx 가 Heap의 한 원소의 index 인데 이 노드로부터 시작하는 subtree를 post order로 방문 (traverse) 하는 재귀 함수이다. 방문한 노드의 내용은 `System.out.print()`로 출력한다.

프로그램 테스트

컴파일

```
$ javac lab006.java LabTest.java
```

실행

```
$ java LabTest
```

주어진 **input**으로 실행

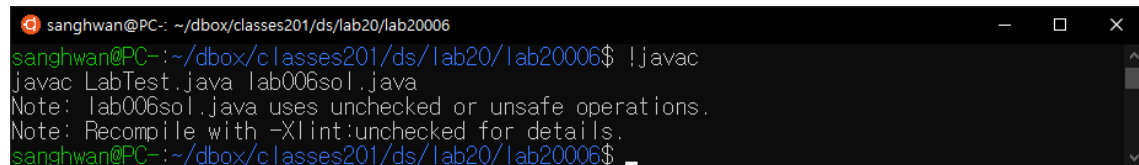
```
$ java LabTest < lab.in
```

주어진 **output**과 비교

```
$ java LabTest < lab.in > abc
```

```
$ diff abc lab.out
```

컴파일시 나오는 다음과 같은 warning은 무시하기 바람.

A terminal window with a dark background and light-colored text. The window title bar shows 'sanghwan@PC: ~/dbox/classes201/ds/lab20/lab20006'. The terminal content shows the user running 'javac LabTest.java lab006sol.java'. The output includes two 'Note' messages: 'Note: lab006sol.java uses unchecked or unsafe operations.' and 'Note: Recompile with -Xlint:unchecked for details.' The prompt returns to the user's shell.

```
sanghwan@PC: ~/dbox/classes201/ds/lab20/lab20006
sanghwan@PC:~/dbox/classes201/ds/lab20/lab20006$ !javac
javac LabTest.java lab006sol.java
Note: lab006sol.java uses unchecked or unsafe operations.
Note: Recompile with -Xlint:unchecked for details.
sanghwan@PC:~/dbox/classes201/ds/lab20/lab20006$
```