Priority Queue

Definition

- **Definition**

  heap property 를 유지하는 data structure

  max-heap → max-priority queue

  min-heap → min-priority queue

- **Data Member**

  A : set of elements (array)

  heap-size[A] : heap size

- **Operations** (max-priority queue 기준)

  HEAP-MAXIMUM(A)

  HEAP-EXTRACT-MAX(A)

  HEAP-INCREASE-KEY(A, i, key)

  MAX-HEAP-INSERT(A, key)

Priority Queue Class

- Class

```
class MaxPriorityQueue
{
public:
    int HEAP-MAXIMUM(int* A);
    int HEAP-EXTRACT-MAX(int* A);
    void HEAP-INCREASE-KEY(int* A, int i, int key);
    void MAX-HEAP-INSERT(int* A, int key);
protected:
    int* A;
    int heap_size;
};
```

Applications

(Ex) 경매 프로그램

최고가 낙찰 : max-priority-queue

(Ex) 입찰 프로그램

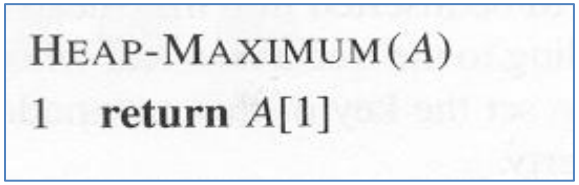
최저가 입찰 : min-priority-queue

(Ex) Job scheduling of machine

Min-priority queue: finish time of all jobs

Operations

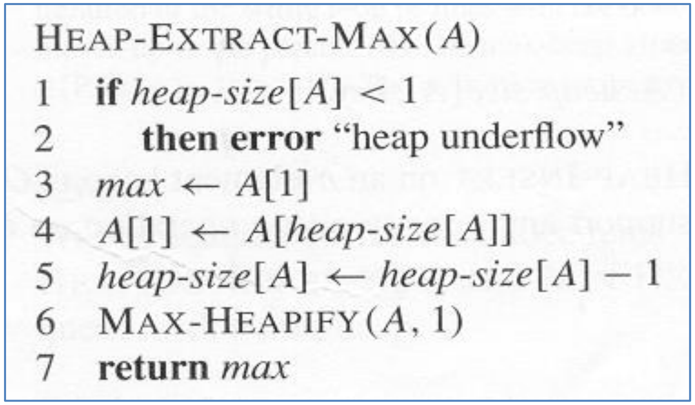
- HEAP-MAXIMUM (A)
  - Operation: 최대값 리턴 (heap size 유지)
  - Algorithm



```
HEAP-MAXIMUM(A)
return A[1]
```

  - Running time: Θ (1)

- HEAP-EXTRACT-MAX(A)
  - Operation: 최대값 리턴 (heap size 감소)
  - Algorithm



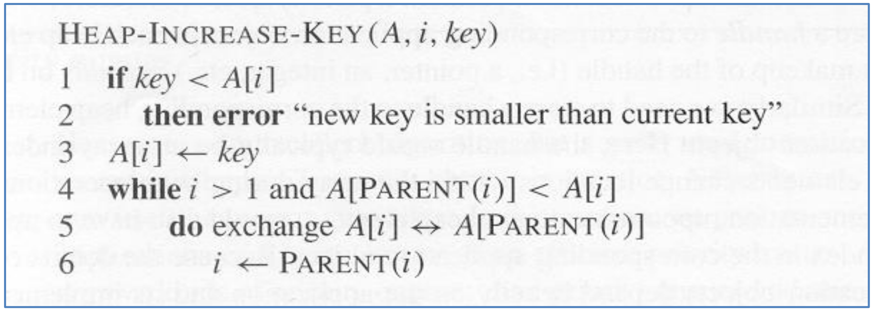
```
HEAP-EXTRACT-MAX(A)
if heap-size[A] < 1
    then error "heap underflow"
max ← A[1]
A[1] ← A[heap-size[A]]
heap-size[A] ← heap-size[A] − 1
MAX-HEAPIFY(A, 1)
return max
```

  - Running time: O (lg n)

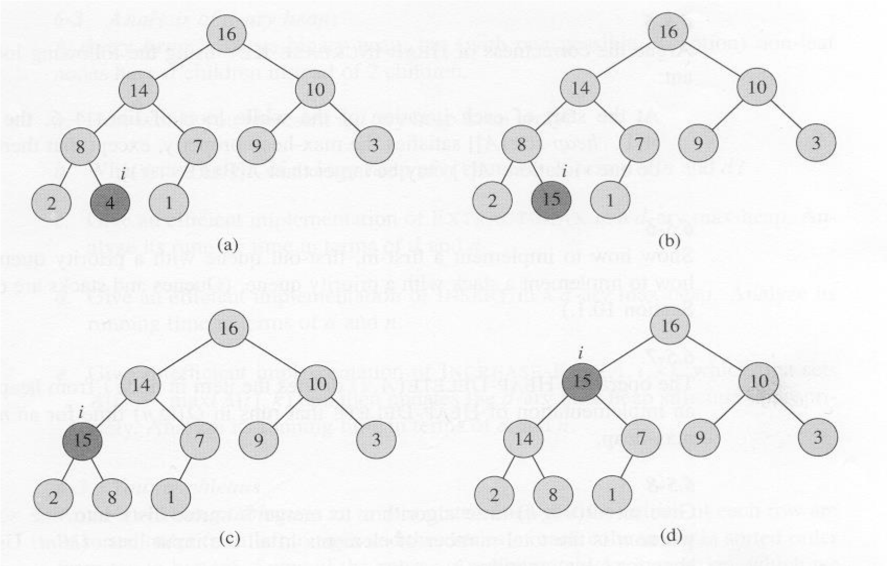
Operations

- HEAP-INCREASE-KEY (A, i, key)
  - Operation:

    heap A 의 i 번째 노드 값을 key 로 증가시키고, 전체를 heap 으로 재구성
  - Algorithm
  - Running time: O (lg n)



```
HEAP-INCREASE-KEY(A, i, key)
if key < A[i]
  then error "new key is smaller than current key"
A[i] ← key
while i > 1 and A[PARENT(i)] < A[i]
    do exchange A[i] ↔ A[PARENT(i)]
       i ← PARENT(i)
```



(a) (b) (c) (d)

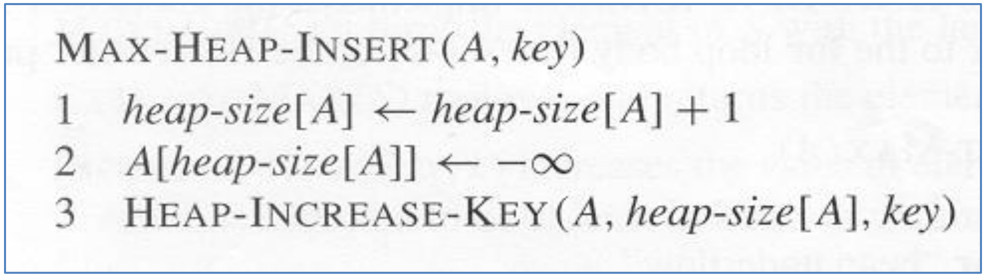
Operations

- MAX-HEAP-INSERT(A, key)
  - Operation

    Heap A 에 key 값을 갖는 노드 추가.
    추가 후에도 A 는 heap property 유지

  - Algorithm

  - Running time: O(lg n)

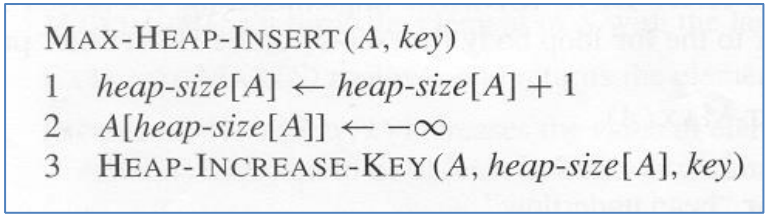


```
MAX-HEAP-INSERT(A, key)
heap-size[A] ← heap-size[A] + 1
A[heap-size[A]] ← −∞
HEAP-INCREASE-KEY(A, heap-size[A], key)
```

Operations

(Q) A=<15, 13, 9, 5, 12, 8, 7, 4, 0, 6, 2, 1>, MAX-HEAP-INSERT(A,10)

15	13	9	5	12	8	7	4	0	6	2	1



```
MAX-HEAP-INSERT(A, key)
heap-size[A] ← heap-size[A] + 1
A[heap-size[A]] ← $-\infty$
HEAP-INCREASE-KEY(A, heap-size[A], key)
```

Operations

(Q) MAX-HEAP-DELETE(A, i) , delete node i from heap A