[CSE3081(2반)] 알고리즘 설계와 분석

2020학년도 2학기

강의자료

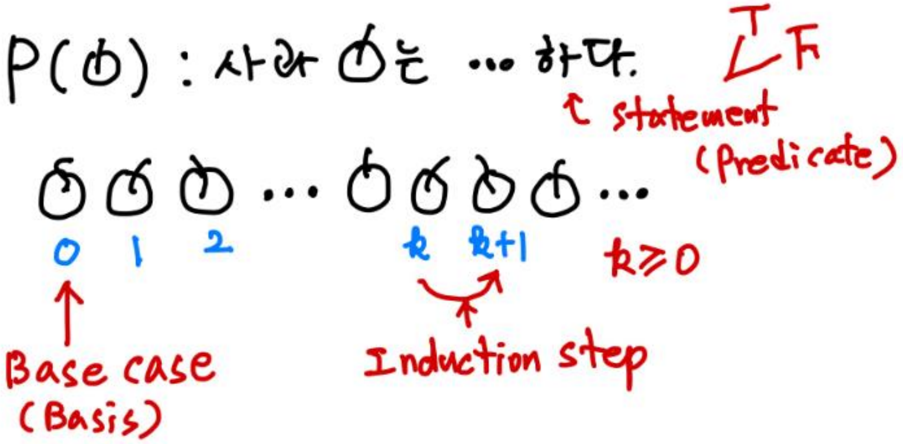
(2020.09.17 목요일)

서강대학교 공과대학 컴퓨터공학과
임 인 성 교수

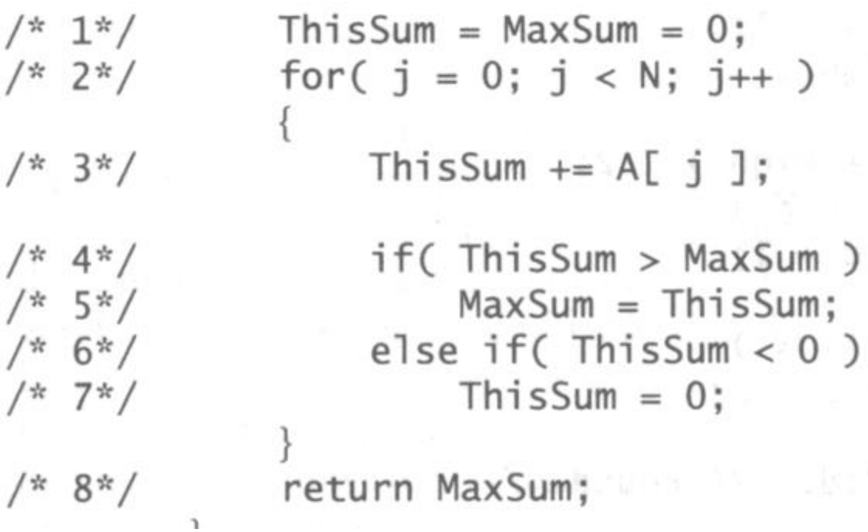
https://en.wikipedia.org/wiki/Mathematical_induction

Mathematical Induction & Proof of Correctness

- **Proof by induction**



- **Proof of correctness: MSS (1D)**



```
/* 1*/        ThisSum = MaxSum = 0;
/* 2*/        for( j = 0; j < N; j++ )
              {
/* 3*/            ThisSum += A[ j ];

/* 4*/            if( ThisSum > MaxSum )
/* 5*/                MaxSum = ThisSum;
/* 6*/            else if( ThisSum < 0 )
/* 7*/                ThisSum = 0;
              }
/* 8*/        return MaxSum;
```

P(j) : for-loop이 j번 수행한 직후에

ThisSum 변수는 (   ) 값을,

MaxSum 변수는 (   ) 값을

가지고 있다.

$k = 0, 1, 2, \cdots, j-1, j, \cdots, N$

$j \geq 1$

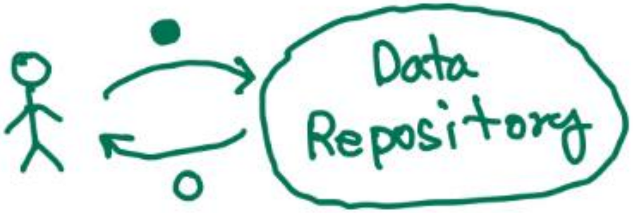
[주제 2]

Heap-based Priority Queues and Heap Sort

(Review)

A Variety of Priority Queue Implementations

- [Priority Queue 1: Max(Min) Heap]
- [Priority Queue 2: Min-Max Heap]
- [Priority Queue 3: Heap and Hashing]
- [Priority Queue 4: Deap]
- [Priority Queue 5: Leftist Tree]
- [Priority Queue 6: Binomial Heap]
- [Priority Queue 7: Fibonacci Heap]



when retrieved,

FIFO - based,

LIFO - based,

priority - based,

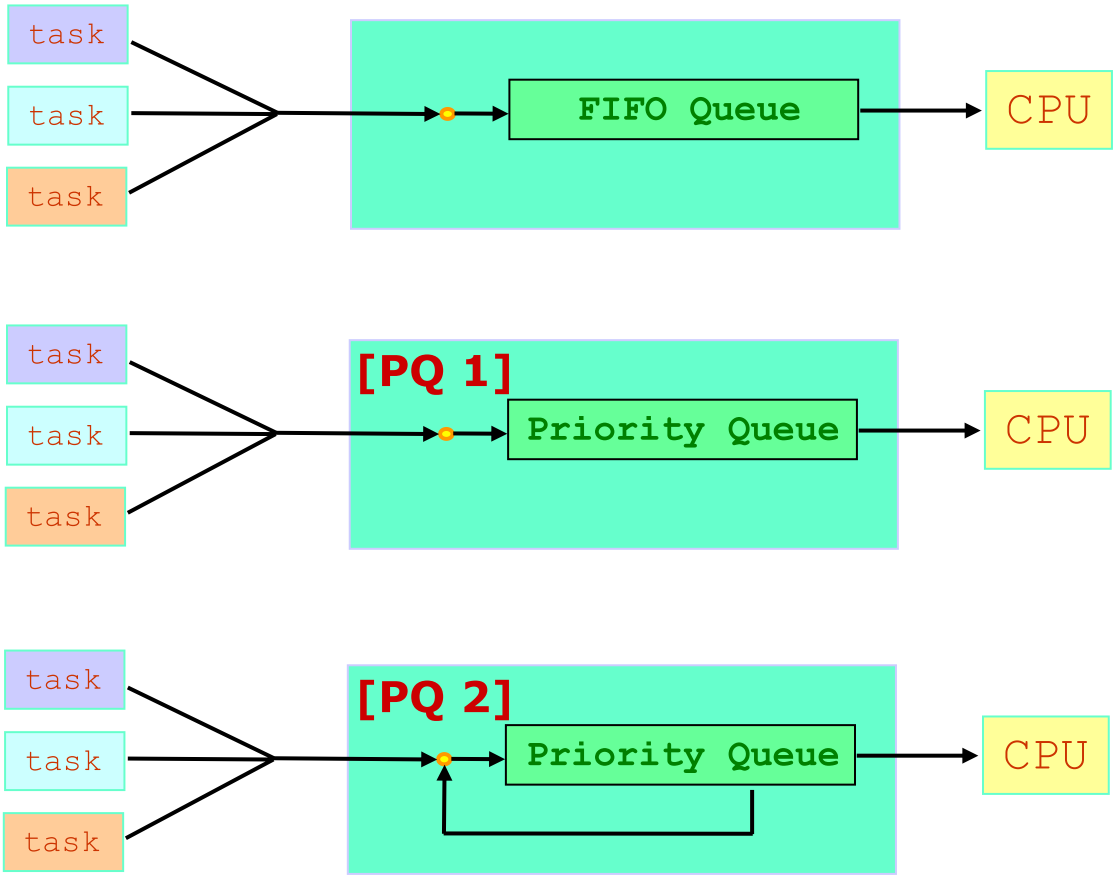
⋮

[Job Scheduling Example: Priority Queue]

- Consider the following sequence of requests in an operating system:

8134	kim	32
8137	kim	15
8138	jpark	24
8142	root	58
8244	ihm	45
8246	jpark	44
8251	root	34
8252	root	58
8255	sylee	21
8256	root	58
8261	jpark	58
8262	kim	32
8264	bhchoi	8
8267	root	58
8268	root	30
8269	root	58
8271	ihm	47

…

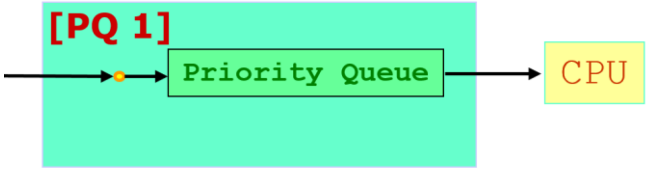


- **Requirement 1**
  - CPU executes the process with **the highest priority** first.

☺ Use a heap structure – a simple max heap.

```
typedef struct _process {
  int proc_id;
  char *owner;
  int priority;
} Process;
static Process *_proc_heap;
static int _proc_heap_size = 0;
static int _proc_heap_ptr = 0;

int PH_create(int n);
int PH_full();
int PH_empty();
int PH_insert(Process item);
int PH_delete(Process *item);
```



- **Requirement 2**
  - The priority of processes **can be modified** after they are placed in the priority queue.

```
int H_change_priority(int proc_id, int new_priority);
```

☹ This function requires locating a particular process in the heap, but the basic heap operations provide no efficient way to do it.

☺ Employ an auxiliary data structure such as a hash table that keeps track of the location of each process in the heap structure.

- Once the two requirements are satisfied, the operating system can process the following basic commands efficiently:

```
INSERT  <proc_id>  <owner>  <priority>
DELETE
CHANGEPR <proc_id> <new_priority>
PRINTHEAP
END
```

Priority Queue 1: Max(Min) Heap

[Horowitz 5.6.2]

[Neapolitan 7.6]

- **Problem**
  - The following operations must be performed as mixed in data processing:
    - Store a record with a key in an arbitrary order.
    - Fetch the record with the current largest key.

- A solution: **Design a data structure** that offers an efficient implementation of the following operations:
  - **Insert an element with an arbitrary key.**
  - **Delete an element with the largest key.**

An Array Implementation

[Sedgewick 9.2]

```
void PQinit();
int  PQempty();
void PQinsert(int);
int PQdelmin();
void PQdec(int)
-----
#include <stdlib.h>
static int *pq;
static int N;
#define MAX_N 10000;
void PQinit() {
   pq = malloc(MAX_N*sizeof(int));
   N = 0;
}

int PQempty() {
   return N == 0;
}

void PQinsert(int v) {
   pq[N++] = v;
}
```

```
int PQdelmin() {
   int j, min = 0;
   for (j = 1; j < N; j++)
      if (less(pq[min], pq[j]))
         min = j;
    exch(pq[min], pq[N-1]);
    return pq[--N];
}

int less(int i, int j) {
   return … ;
}
void exch(int i, int j) {
   …
}
void PQdec(int k) {
   …
}
```

- What will be the worst-case time complexity of each operation?

Max(Min) Heap: Definitions

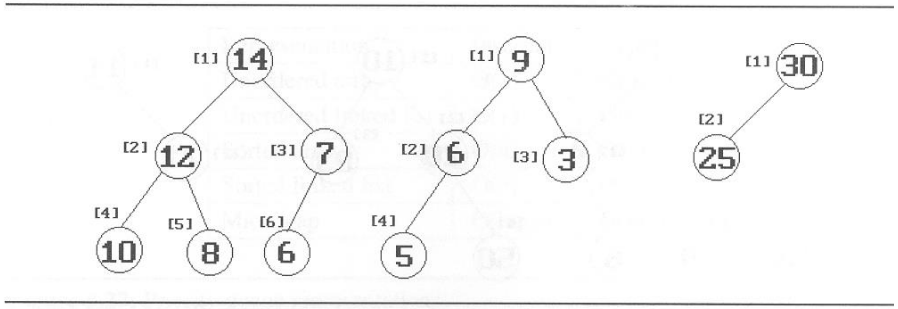
[Horowitz 5.6.2]
[Neapolitan 7.6]

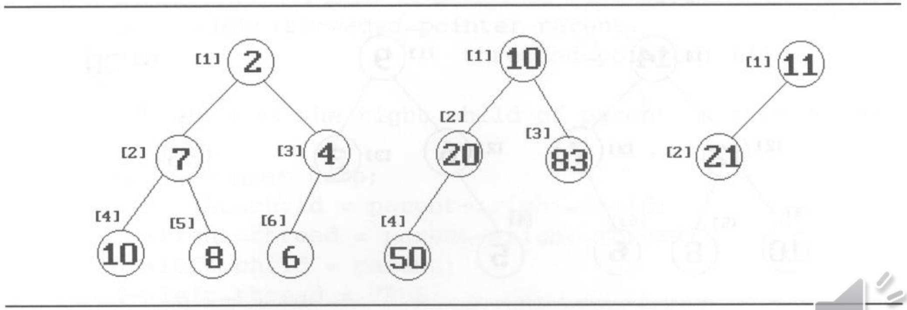
- **Definition 1**
  - A max(min) heap is a complete binary tree where the key value in each internal node is no smaller(larger) than the key values in its children.

- **Definition 2**
  - A binary tree has the **max(min) heap property** if and only if

    ① The number of nodes of the tree is either 0 or 1, *or*

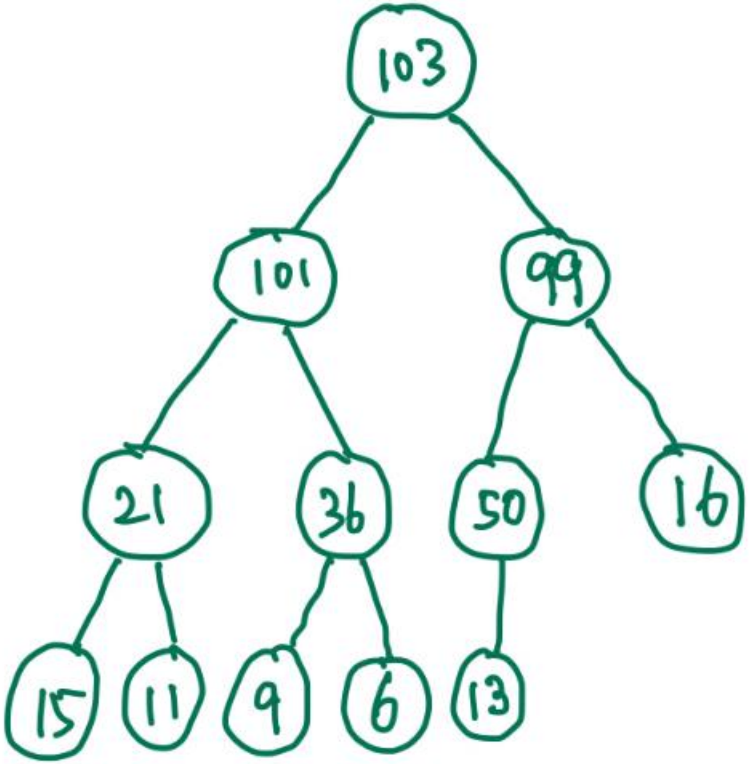
    ② For the tree that has at least two nodes, the key in the root is no smaller(larger) than that in each child and the subtree rooted at the child has the **max(min) heap property**.
  - A max(min) heap is a complete binary tree that has the max(min) heap property.



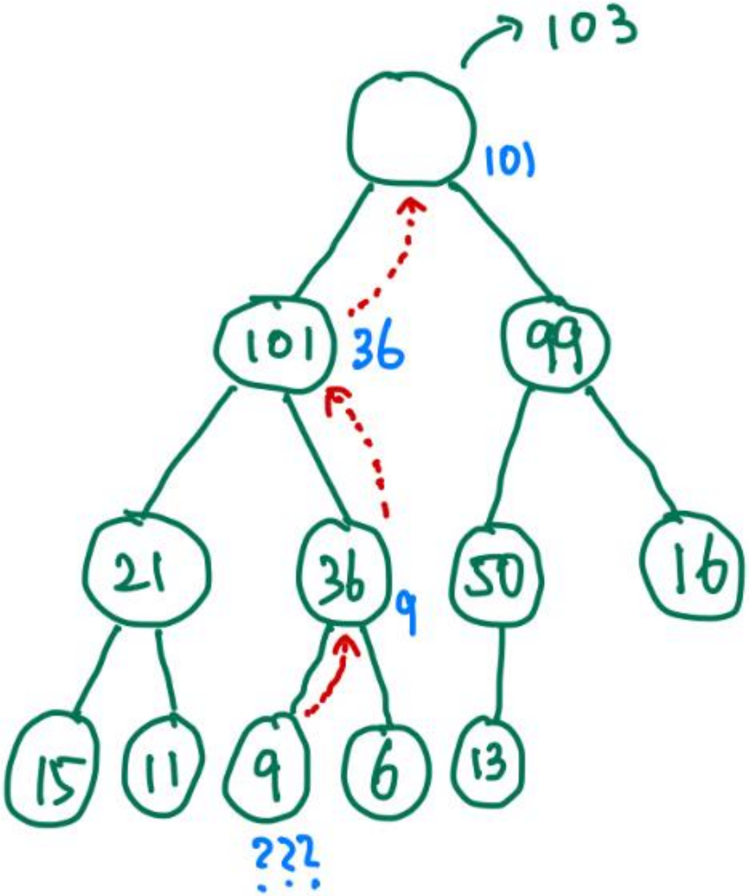


Brainstorming on Max Heap Operations

Max Heap Example



Deletion Example 1


103
101
101
36
99
21
36
9
50
16
15
11
9
6
13
???