

HW5

문제 1:

Write insertion and deletion functions for a max heap represented as a linked binary tree. Assume that each node has a parent field as well as the usual left child, right child, and data fields.

예제	
입력 (input1.txt)	출력 (output1.txt)
i 4	Insert 4
i 4	Exist number
i 5	Insert 5
d	Delete 5
d	Delete 4
d	The heap is empty
i 3	Insert 3
q	

- 입력:
 - i k** – heap 에 key 값이 k 인 node 를 insert 한다.
 - d** – heap 에서 가장 큰 key 를 가진 node 를 delete 한다.
 - q** – 프로그램을 종료한다.
- 출력:
 - i k** 가 성공적으로 동작했을 경우 – **Insert k**
 - i k** 가 실패한 경우 (이미 존재하는 key 값일 경우) – **Exist number**
 - d** 가 성공적으로 동작했을 경우 – **Delete k**
 - d** 가 실패한 경우 (heap 이 비어있을 경우) – **The heap is empty**

- 제약 조건:

- Linked representation 을 사용할 것
- 주어진 input 형식 외에, 예외 input 은 들어오지 않는다고 가정
- 아래의 자료구조를 선언하여 사용할 것:

```
typedef struct node *treePointer;
typedef struct node {
    int key;
    treePointer parent;
    treePointer leftChild, rightChild;
};
```

- 문제 2:

Write a program to construct binary search tree from given preorder traversal, and perform inorder and postorder traversal on it.

예제	
입력 (input2.txt)	출력 (output2.txt)
6 30 5 2 40 35 80	Inorder: 2 5 30 35 40 80 Postorder: 2 5 35 80 40 30
6 30 5 5 40 35 80	cannot construct BST

- 입력:
 - 첫 번째 줄 – tree node 의 개수 n
 - 두 번째 줄 – n 개의 key value
- 출력:
 - 한 줄에 하나씩, inorder traversal 과 postorder traversal 에 대한 결과
 - 중복되는 key 가 있을 경우, “cannot construct BST” 출력
- 제약 조건:
 - 반드시 tree 를 만든 후, inorder 와 postorder traversal 을 진행할 것
 - tree 를 만들지 않은 경우, 0 점 처리

▪ 문제 3:

Write a program for a max priority queue that represents the priority queue as a binary search tree.
Your codes for top, pop and push should have complexity $O(h)$, where h is the height of the search tree.

예제	
입력 (input3.txt)	출력 (output3.txt)
push 3	Push 3
push 3	Exist number
top	The top is 3
push 5	Push 5
top	The top is 5

pop	Pop 5
push 3	Exist number
pop	Pop 3
pop	The queue is empty
q	

- 입력:
 - **push k** – queue 에 key 값이 k 인 node 를 insert 한다.
 - **top** – heap 에서 가장 큰 key 를 가진 node 를 출력한다.
 - **pop** – heap 에서 가장 큰 key 를 가진 node 를 delete 한다.
 - **q** – 프로그램을 종료한다.
- 출력:
 - push k 가 성공적으로 동작했을 경우 – **Push k**
 - push k 가 실패한 경우 (이미 존재하는 key 값일 경우) – **Exist number**
 - top 이 성공적으로 동작했을 경우 – **The top is k**
 - pop 이 성공적으로 동작했을 경우 – **Pop k**
 - top/pop 이 실패한 경우 (queue 가 비어있을 경우) – **The queue is empty**
- 제약 조건:
 - Binary search tree 를 사용할 것
 - 주어진 input 형식 외에, 예외 input 은 들어오지 않는다고 가정

- **제출 주의사항**

- 이름: **HW5_학번.zip**
 - ex) HW5_20240000.zip
- 압축을 풀면 아래의 파일들이 있어야 함:
 - HW5_학번_1.c
 - HW5_학번_2.c
 - Document.pdf
- 컴파일 에러가 발생할 경우 0 점 처리
- 무한 루프/세그멘테이션 오류는 해당 testcase 0 점 처리
- 프로그램이 일정시간 안에 답을 출력 안하는 경우 틀린 출력이라고 간주함.
- **입출력 양식이 틀릴 경우 감점**
- 입력과 출력은 파일 입출력으로 하며, txt 파일 이름은 주어진 예제에 있는 것으로 하면 됨.
- 과제 채점은 cspiro 의 gcc compiler 기준 (.c 일경우) 또는 g++기준 (.cpp 일경우)
- Copy 검사