1971044 정희수
HW08_tech_01
구조체 / 함수 / main 함수 / 실행화면

#구조체

```
typedef struct {
    int heap[MAX_ELEMENT];
    int heap_size;
} HeapType;
```

HeapType 구조체를 정의한다.
int heap[MAX_ELEMENT] // 힙에 들어가는 노드의 값을 저장하는 배열
int heap_size // 힙에 들어가는 노드 개수

#함수

```
void init(HeapType* h) {
    h->heap_size = 0;
}
```

h가 가리키는 HeapType을 초기화 한다.
HeapType* h //h가 가리키는 HeapType
h -> heap_size =0 // h가 가리키는 힙에 들어있는 노드 개수가 0이 된다.

```
void insert_min_heap(HeapType* h, int key) {
    int i;
    i = ++(h->heap_size);
    while ((i != 1) && (key < h->heap[i / 2])) {
        h->heap[i] = h->heap[i / 2];
        i /= 2;
    }
    h->heap[i] = key;
}
```

h가 가리키는 HeapType에 key값을 가지는 노드를 삽입한다.
while문 // I가 1이 아니고, 삽입할 key값이 heap[i/2]보다 작을 때까지

```
void Decrease_key_min_heap(HeapType* h, int i, int key) {
    int temp;
    if (key >= h->heap[i]) {
        printf("error new key is not smaller than current key ");
        return;
    }
    h->heap[i] = key;
    while ((i > 1) && (h->heap[i / 2] > h->heap[i])) {
        temp = h->heap[i / 2];
        h->heap[i / 2] = h->heap[i];
        h->heap[i] = temp;
        i /= 2;
    }
}
```

heap의 노드안의 값보다 작은 값을 대입하여 min heap을 만든다
HeapType* h //h가 가리키는 HeapType
int i // 힙의 i번째 노드 (==heap[i])
int key // 바꿀 노드 안의 값
if문 // key값이 I번째 노드보다 크면 error 메시지를 출력하고 끝낸다.
원래 I번째 노드보다 작은 key값을 대입하면
min_heap이기 때문에 I번째 노드의 부모노드들만 비교해주면 된다.

```
void increase_key_min_heap(HeapType* h, int i, int key) {
    int temp;
    if (key <= h->heap[i])
        printf("error : new key is not bigger than current key ");
    h->heap[i] = key;
    while ((i<h->heap_size) && (h->heap[i]>h->heap[i * 2]) ) {
        temp = h->heap[i * 2];
        h->heap[i * 2] = h->heap[i];
        h->heap[i] = temp;
        i *= 2;
    }
    i /= 2;
    h->heap[i] = key;

}
```

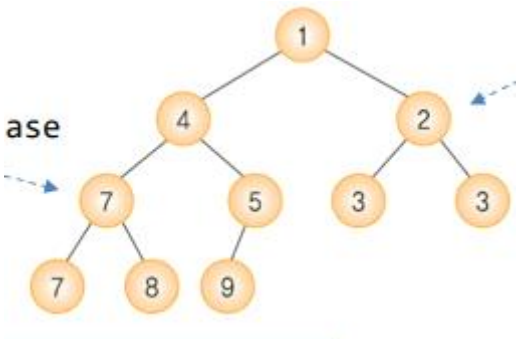
heap의 노드안의 값보다 큰 값을 대입하여 min heap을 만든다
HeapType* h //h가 가리키는 HeapType
int i // 힙의 i번째 노드 (==heap[i])
int key // 바꿀 노드 안의 값
if문 // key값이 I번째 노드보다 작으면 error 메시지를 출력하고 끝낸다.
원래 I번째 노드보다 큰 key값을 대입하면
min_heap이기 때문에 I번째 노드의 자식노드들만 비교해주면 된다.

```
void print(HeapType* h) {
    for (int i = 1;i < h->heap_size + 1;i++)
        printf("%d₩n", h->heap[i]);
    printf("----------₩n");
}
```

힙의 노드들을 순서대로 출력해준다.
HeapType* h //h가 가리키는 HeapType

#main함수

```
void main(void) {
    HeapType h;
    init(&h);

    insert_min_heap(&h, 1);
    insert_min_heap(&h, 4);
    insert_min_heap(&h, 2);
    insert_min_heap(&h, 7);
    insert_min_heap(&h, 5);
    insert_min_heap(&h, 3);
    insert_min_heap(&h, 3);
    insert_min_heap(&h, 7);
    insert_min_heap(&h, 8);
    insert_min_heap(&h, 9);

    print(&h);

    Decrease_key_min_heap(&h, 4, 3);

    print(&h);

    increase_key_min_heap(&h, 3, 10);

    print(&h);

}
```

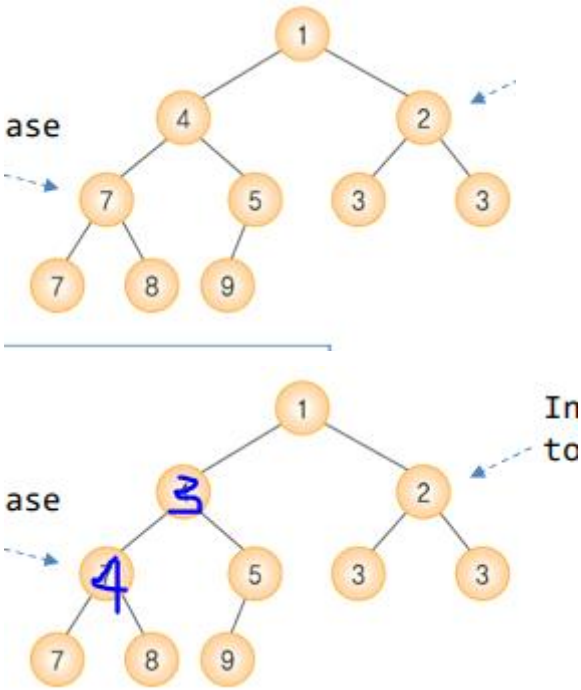


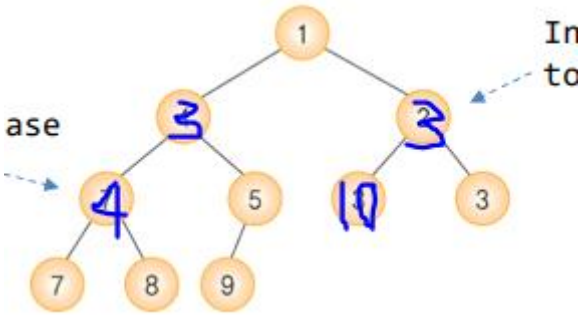
위 그림처럼 힙에 노드 값들을 차례대로 삽입한다.
힙의 노드들을 출력한다.
힙의 4번째 노드의 값(7)을 3으로 바꾼다.

힙의 노드들을 출력한다.
힙의 3번째 노드의 값(2)을 10으로 바꾼다.
힙의 노드들을 출력한다.

#실행화면

아래 그림 순서대로 힙의 노드 값이 바뀐다.




In
to
ase
1
3
2
4
5
10
3
7
8
9