

# Algorithm Homework 3B

학번 / 이름: 2014310407 / 이 준 혁

## 1. 구현

우선 NIL의 색이 존재하기 때문에, 색을 반환하는 함수 getcolor와 색을 설정하는 함수 setcolor를 만들었다. getcolor함수는 NIL인 경우 BLACK을, 아닌 경우 노드의 색을 반환했다. 반면 setcolor함수는 NIL인 경우 아무것도 하지 않았고, NIL이 아닌 경우 노드의 색을 주어진 색으로 설정했다. 그리고 다음 6가지 함수를 구성했다.

1. rotate\_left, 2. rotate\_right, 3. rbt\_create, 4. insert, 5. insert\_rbt, 6. insert\_rbt\_fixup

1. rotate\_left: 입력으로 레드블랙트리와 회전할 노드 x를 받는다. 그리고 y를 x의 오른쪽 자식으로 두고 y의 왼쪽 자식을 x의 오른쪽 자식으로 준다. y의 왼쪽 자식이 NIL이 아닌 경우, y의 왼쪽 자식의 parent를 x로 지정한다.

그리고 y가 x의 부모가 되어야 하므로, 우선 y의 부모를 x의 부모로 지정한다. 만약 x의 부모가 NIL인 경우, x가 root였다는 의미이므로 y를 root로 둔다.

x의 부모가 NIL이 아니었을 때는, x가 원래 왼쪽 자식이었는지, 오른쪽 자식이었는지를 판단해 이에 따라 x의 부모의 왼쪽 / 오른쪽 자식을 y로 둔다. 마지막으로, y의 왼쪽 자식을 x로, x의 부모를 y로 둔다.

2. rotate\_right: 1번에서 방향만 바뀌었으므로, 왼쪽과 오른쪽만 바뀌어서 똑같이 구현했다.

3. rbt\_create: malloc 함수를 이용해 tree만큼의 크기를 할당 받고, tree에 아무 리프도 없으므로, root를 NIL로 초기화했다. 그리고 이렇게 만든 tree를 반환했다.

4. insert: 입력으로 tree와 key를 받는다. malloc 함수를 이용해 node를 만든 후, node의 키 값을 입력 받은 값으로 대입한다. 또, 부모와 자식들을 모두 NIL로 초기화한다. 그리고 binary search tree에서 위치를 찾아서 삽입하는 함수인 insert\_rbt함수를 불렀다.

5. insert\_rbt: 입력으로 tree와 입력할 노드 z를 받는다. x를 트리의 루트로, y를 NIL로 설정한 후, x가 NIL이 될 때까지 왼쪽 혹은 오른쪽으로 이동한다. 이 때 z의 키 값이 x보다 작으면 왼쪽으로, 크면 오른쪽으로 이동한다. 그리고 x의 부모가 항상 y가 되도록 한다. x가 NIL이면, y가 삽입될 위치의 부모가 된다. 이후에 z의 부모를 y로 설정한다. y가 NIL이면, 빈 트리이므로 트리의 루트를 z로 설정한다. 아닌 경우 z의 키가 y의 키보다 작은 경우, z는 y의 left가 되므로 y의 left를 z로 설정한다. 반대의 경우에는 y의 right를 z로 설정한다.

그리고 z의 색깔을 RED로 설정한 후, 색을 규칙에 맞게 변경해야 하므로, insert\_rbt\_fixup함수를 호출한다.

6. insert\_rbt\_fixup:

입력으로 트리와 삽입된 노드 z를 받는다. z의 부모가 z의 부모의 부모의 왼쪽 자식이라고 생각해 보자. (오른쪽인 경우는 대칭이므로 방향만 다를 뿐 같음.) 이 때 z의 부모가 red일 때 문제가 생기는 것이므로,

z의 부모가 red일 동안 계속 반복해서 다음과 같이 세 가지 경우로 나누어 실행한다. 이 때 sibling이라는 노드를 z의 부모의 형제 노드, 즉 z의 부모의 부모의 오른쪽 자식으로 설정한다.

6-1. sibling이 red인 경우, 6-2. sibling이 black이며 z가 오른쪽 자식인 경우. 6-3. sibling이 black이며 z가 왼쪽 자식인 경우

6-1: sibling이 red이면, z의 부모와 sibling을 black으로, 그리고 z의 부모의 부모를 red로 색을 변경한다. 이후 z를 z의 부모의 부모로 바꾼다. 이렇게 하면 z에서 생겼던 색 문제가 z의 부모의 부모로 옮겨 간다. 또한, 이는 어떤 black height도 변경시키지 않는다. 즉, 변경된 z의 색 문제를 제외하고는 어떤 규칙 위반도 없으니, 다시 반복해서 문제를 해결하면 된다.

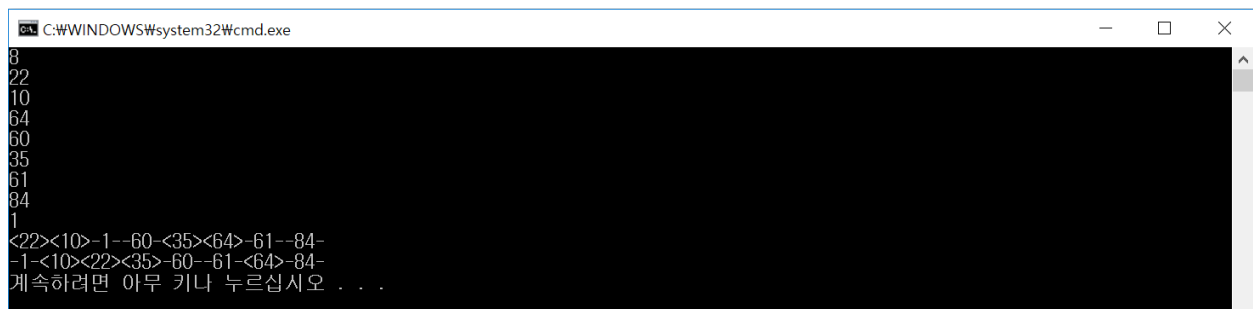
6-2: 이 경우에는, z의 부모로 z를 바꾼 후, z에 대해 왼쪽으로 rotate를 한다. 이렇게 바꾸면 6-3과 완벽히 일치한다.

6-3: 우선 z의 부모를 black으로, z의 부모의 부모를 red로 변경한다. 그리고 z의 부모의 부모에 대해 오른쪽으로 rotate를 하면, black height도 변경되지 않고, 색도 규칙을 따른다. 또, 이 경우에는 z가 가졌던 색 문제도 해결되므로, 이 경우를 만난다면 insertion이 끝나게 된다.

z의 부모가 z의 부모의 부모의 오른쪽인 경우에는 방향만 바꾸어 똑 같은 방식으로 구현했다.

마지막으로, root가 black이어야 한다는 규칙에 의해, z의 root를 black으로 설정했다. 이렇게 fixup 함수를 구현했다.

## 2. 결과



```
C:\WINDOWS\system32\cmd.exe
8
22
10
64
60
35
61
84
1
<22><10>-1--60-<35><64>-61--84-
-1-<10><22><35>-60--61-<64>-84-
계속하려면 아무 키나 누르십시오 . . .
```

tree 가 제대로 구성되었으며, inorder 와 preorder 모두 색에 맞게 잘 출력됨을 확인했다.