



컴퓨터 알고리즘과 실습



학번 : 2016110056

학과 : 불교학부

이름 : 박승원

날짜 : 2017년 4월 5일

문제 1

- 연결 리스트(Linked List) 기반의 이진 탐색트리(Binary Search Tree)를 구현하여 보고 탐색, 삽입, 삭제를 수행해 보자.
 - 빈 나무를 생성한 후에 20,6,2,4,16,10,8,12,14,9 를 삽입하여 트리 T1를 생성한 후, 생성된 T1에서 10을 탐색하고 10의 오른쪽과 왼쪽 노드의 키 값을 출력하여 보아라 (프로그램 작성한후 이용하여 나온 결과를 출력할 것).
 - T1에서 6을 삭제한 후, 10을 탐색하고 10의 오른쪽과 왼쪽 노드의 키 값을 출력하여 보아라 (프로그램 작성한후 이용하여 나온 결과를 출력할 것).
 - 빈 나무 생성, 탐색, 삽입, 삭제를 위한 소스코드 및 위 문제를 풀기 위한 코드를 제출하시오.

```
// 2016110056 박승원
#include <stdio.h>
#include <time.h>
#include <stdlib.h>
#include <string.h>
#include <iostream>
using namespace std;

typedef struct Tree {
    int num;
    struct Tree * left , * right ;
} Tree;

Tree* insert (Tree* p, int num) { // 삽입 재귀 함수
    if (!p) {
        p = (Tree*)malloc(sizeof(Tree));
        p->num = num;
        p->left = NULL;
        p->right = NULL;
    } else if (num > p->num) p->right = insert(p->right, num);
    else if (num < p->num) p->left = insert(p->left, num);
    return p;
}

int search (Tree* p, int num) {
    if (!p) return -1;
    if (p->num == num) return 1;
    if (p->num < num) return search(p->right, num);
    if (p->num > num) return search(p->left, num);
    return -1;
}

void print (Tree* p) {
    if (!p) return;
    cout << p->num << endl;
    print(p->left);
    print(p->right);
}
```

```

else p->left = insert (p->left, num);
return p; // 리턴값은 (재귀적 호출시) 항상 부모와 연결할 값이라고 생각하면 된다.
}

```

```

void show(Tree* p) { // 중위 순회 출력 함수
    if (p->left) show(p->left);
    printf ("%d\n", p->num);
    if (p->right) show(p->right);
}

```

```

Tree* del_node(Tree* p) { // p가 가르키는 노드를 삭제한다 .
    if (!p->left && !p->right) { // 자식이 없을 때
        free (p);
        return NULL;
    } else if (!p->right) { // 오른쪽만 널일때
        Tree *r = p->left;
        free (p);
        return r; // 리턴값을 대입하는 형식으로 부모와 연결한다 .
    } else if (!p->left) { // 왼쪽만 널일때
        Tree *r = p->right;
        free (p);
        return r;
    } else { // 둘 다 자식일때
        Tree* prev = p;
        Tree* q = p->right;
        for ( ; q->left; q = q->left) prev = q;
        p->num = q->num;
        if (prev == p) prev->right = del_node(q); // q를 재귀적으로 삭제하고 이어준다 .
        else prev->left = del_node(q);
        return p;
    }
}

```

```

Tree* del(Tree* p, int num) { // num 요소를 삭제한다 .
    if (!p) return NULL;
    if (p->num == num) return del_node(p); // del_node를 호출한다 .
    if (p->num > num) p->left = del(p->left, num);
    if (p->num < num) p->right = del(p->right, num);
    return p;
}

```

```

Tree* find(Tree* p, int num) { // 조회
    if (!p) return NULL;
    if (p->num == num) return p;
    if (p->num > num) return find(p->left, num);
    if (p->num < num) return find(p->right, num);
}

void free_tree (Tree* p) {
    if (p->left) free_tree (p->left);
    if (p->right) free_tree (p->right);
    free (p);
}

int main()
{
    Tree* tree = NULL;
    tree = insert (tree , 20); // 최초 삽입이 외에는
    insert (tree , 6); // tree에서는 삽입시 루트가 바뀌지 않으므로
    insert (tree , 2); // 대입하지 않아도 된다 .
    insert (tree , 4);
    insert (tree , 16);
    insert (tree , 10);
    insert (tree , 8);
    insert (tree , 12);
    insert (tree , 14);
    insert (tree , 9);
    show(tree);
    printf ("=====\\n");
    Tree* p = find (tree , 10); // 조회
    cout << p->left->num << ", " << p->right->num << endl;
    del (tree , 6);
    p = find (tree , 10); // 조회
    cout << p->left->num << ", " << p->right->num << endl;
    free_tree (tree );
}

```

파일(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H)

```
#include <stdlib.h>
#include <string.h> [1,2,1,2,1, 0,0,0,0, 0]
#include<iostream>
using namespace std;

typedef struct Tree {
    int num;
} n = min(n, ar[idx-5])
----- 문제 1번 실행을 시작합니다. -----
./1.x
2
4
6
8
9
print ar
10
12
14
16
20
=====
8 , 12
9 , 12
----- 문제 1번 실행을 종료합니다. -----
```

1.x 12.png 4.c 7.png calc.c report.log
10-14.pdf 12.x 4.cpp 7.x calc.png report.pdf
10-21.pdf 2.c 4.png 8.c calc.x report.synctex.gz
10.png tdc++24cpp 4.x 8.png complex.c report.tex
10.x 2.png 5.c 8.x complex.png sscan.x
11-11.pdf 2.x 5.cpp 9.png complex.x varg
zezeon@ubuntuZ:~/Programming/basicProgramming\$ rm 1.png
zezeon@ubuntuZ:~/Programming/basicProgramming\$ rm 2.png
zezeon@ubuntuZ:~/Programming/basicProgramming\$ make tex

Coin Change Problem

- Problem: Convert some amount of money into given denominations using the smallest number of coins
- Input: An amount of money, M and an array of d denominations $c=(c_1, c_2, \dots, c_d)$, in decreasing order of values ($c_1 > c_2 > \dots > c_d$)
- Output: A list of d integers i_1, i_2, \dots, i_d such that $c_1i_1 + c_2i_2 + \dots + c_di_d = M$ and $i_1 + i_2 + \dots + i_d$ is as small as possible

문제 2

문제 1.2. 동전 교환문제를 Dynamic Programming Algorithm을 이용하여 사용한 동전의 조합과 최소의 동전 갯수를 출력하는 프로그램을 구현하여 보자.

- 유사코드를 적어보아라.
- 알고리즘의 수행시간을 빅오로 표현해 보아라.
- 동전을 조합을 찾는 과정을 그림으로 표현해 보아라.
- 프로그램을 작성하여 보아라.
- 작성한 프로그램을 M=11, c=1,3,5 인 경우에 돌렸을 때 동전갯수를 출력해보시오.

```
M <- 11
c <- [5, 3, 1]
ar <- [1,2,1,2,1, 0,0,0,0,0, 0]
FUNCTION fill_table(idx) :
    m <- min(ar[idx-1], ar[idx-3])
    n <- min(m, ar[idx-5])
    RETURN n+1
ENDFUNCTION

for i in range(5, 11):
    ar[i] <- fill_table (i)
ENDFOR
OUTPUT ar
```

함수에서 2번의 비교와 2번의 대입이 이루어지고 각 요소에 한번씩 함수가 호출되므로, $T(4n) = O(n)$ 이다.

1	2	1	2	1							
1	2	1	2	1	2						
1	2	1	2	1	2	3					
1	2	1	2	1	2	3	2				
1	2	1	2	1	2	3	2	3			
1	2	1	2	1	2	3	2	3	2		
1	2	1	2	1	2	3	2	3	2	3	

```
M = 11
c = [5, 3, 1]
ar = [1,2,1,2,1, 0,0,0,0,0, 0]
```

```
def fill_table (idx) :
    m = min(ar[idx-1], ar[idx-3])
    n = min(m, ar[idx-5])
    return n+1

for i in range(5, 11):
    ar[i] = fill_table (i)

print ar
```

```
zezeon@ubuntuZ:~/Programming/python$ python coin.py
[1, 2, 1, 2, 1, 2, 3, 2, 3, 2, 3]
```

문제 3

돌 문제

- 각 m 개와 n 개의 돌이 들어있는 두개의 상자가 있습니다. 영희와 철수가 어느날 상자를 가지고 재미있는 게임을 제안하였습니다. 게임을 룰은 다음과 같습니다. 영희가 시작으로 각각 돌아 가면서 각자의 순서에 한상자에서 하나의 돌을 꺼내거나 두 상자 각각에서 하나씩 돌을 꺼낼 수 있습니다(한 상자에서 두개의 돌을 꺼낼 수 없음). 꺼낸 돌은 상자에 다시 들어갈 수 없습니다. 마지막에 돌을 꺼내는 사람이 이기는 게임입니다. Dynamic programming algorithm 을 이용하여 이길 수 있는 전략을 세워 보시오.



문제 3

3.1. 게임을 누가 먼저 시작하는지가 게임의 승패를 좌우하나요?

3.2. Dynamic programming 을 사용하여 1~5 범위내의 m 과 n 에 대해서 이기는 전략 테이블을 이용 만들어 보시오. 예를 들어, $m=5$, $n=3$ 일 때 영희와 철수 중 누가 이길지 테이블을 보고 알 수 있어야 합니다. 힌트. 영희가 이길 때는 테이블에 W 를 적고 영희가 질때는 테이블에 L 를 적으시오.

3.3. 이 전략을 유사 코드로 적어보시오.

3.4. 이 전략을 프로그램으로 작성해 보고, $m=10$, $n=7$ 일때 누가 이길 지 출력해보시오.

누가 먼저 시작하는지에 따라 승패가 갈린다.

테이블 상에서 좌측 혹은 위쪽 혹은 좌상에 L이 있으면 승리할 수 있다. 좌에서 하나를 빼거나 우에서 하나를 빼거나 양쪽에서 하나를 빼는 경우가 상대의 패가 되는 경우이므로.

	0	1	2	3	4	5
0	L	W	L	W	L	W
1	W	W	W	W	W	W
2	L	W	L	W	L	W
3	W	W	W	W	W	W
4	L	W	L	W	L	W
5	W	W	W	W	W	W

```
m <- [['L']*11 for i in range(11)]  
ENDFOR  
for i in range(5):  
    m[2*i+1][0] <- 'W'  
    m[0][2*i+1]='W'  
ENDFOR  
for x in range (1,11) :  
    for y in range (1,11) :  
        IF m[x][y-1] is 'L' OR m[x-1][y] is 'L' OR m[x-1][y-1] is 'L':  
            m[x][y] <- 'W'  
        ENDIF  
    ENDFOR  
ENDFOR  
for a in m:  
    OUTPUT a;
```

```
m = [['L']*11 for i in range(11)]  
for i in range(5):  
    m[2*i+1][0] = 'W'  
    m[0][2*i+1]='W'  
for x in range(1,11) :  
    for y in range(1,11) :  
        if m[x][y-1] is 'L' or m[x-1][y] is 'L' or m[x-1][y-1] is 'L':  
            m[x][y] = 'W'  
  
for a in m:  
    print a;
```

```
zezeon@ubuntuZ:~/Programming/basicProgramming$ python game.py
['L', 'W', 'L', 'W', 'L', 'W', 'L', 'W', 'L', 'W', 'L']
['W', 'W', 'W', 'W', 'W', 'W', 'W', 'W', 'W', 'W', 'W']
['L', 'W', 'L', 'W', 'L', 'W', 'L', 'W', 'L', 'W', 'L']
['W', 'W', 'W', 'W', 'W', 'W', 'W', 'W', 'W', 'W', 'W']
['L', 'W', 'L', 'W', 'L', 'W', 'L', 'W', 'L', 'W', 'L']
['W', 'W', 'W', 'W', 'W', 'W', 'W', 'W', 'W', 'W', 'W']
['L', 'W', 'L', 'W', 'L', 'W', 'L', 'W', 'L', 'W', 'L']
['W', 'W', 'W', 'W', 'W', 'W', 'W', 'W', 'W', 'W', 'W']
['L', 'W', 'L', 'W', 'L', 'W', 'L', 'W', 'L', 'W', 'L']
['W', 'W', 'W', 'W', 'W', 'W', 'W', 'W', 'W', 'W', 'W']
['L', 'W', 'L', 'W', 'L', 'W', 'L', 'W', 'L', 'W', 'L']
['W', 'W', 'W', 'W', 'W', 'W', 'W', 'W', 'W', 'W', 'W']
['L', 'W', 'L', 'W', 'L', 'W', 'L', 'W', 'L', 'W', 'L']
['W', 'W', 'W', 'W', 'W', 'W', 'W', 'W', 'W', 'W', 'W']
['L', 'W', 'L', 'W', 'L', 'W', 'L', 'W', 'L', 'W', 'L']
```

위의 도표 상에서 보면 승리한다.

문제 4

박테리아 VS 바이러스 문제:

- 음식 접시위에 n 개의 박테리아와 1개의 바이러스가 있습니다. 일분마다 각바이러스가 하나씩의 박테리아를 죽이고 남은 박테리아들과 바이러스들은 복제를 하게 됩니다. 이 과정이 계속 지나면 마지막에 음식접시 위에는 박테리아가 남을까요 바이러스가 남을까요? 만약 모든 박테리아가 죽는다면, 모든 박테리아가 죽는데 걸리는 시간은 얼마일까요?

문제 4

박테리아 VS 바이러스 문제:

- 박테리아는 1분마다 복제된다.
- 바이러스는 1분마다 복제된다.
- 바이러스는 박테리아를 1분마다 1개의 박테리아를 죽인다.

V는 바이러스의 수, m은 분, B는 박테리아의 수.

$$V_m = 2^m$$

$$B_{m+1} = (B_m - 2^m) \times 2$$

$$B_{m+1} = 2B_m - 2^{m+1}$$

$$\frac{B_{m+1}}{2^{m+1}} = \frac{B_m}{2^m} - 1$$

$$b_m = \frac{B_m}{2^m}$$

$$b_0 = n, b_m = n - m$$

$\therefore B_m = 2^m(n - m) = 0$ 이므로 n분 이후에 박테리아의 수는 0이 된다.