



# Week7\_과제 (1620)

[통계](#) [수정](#) [삭제](#)

yuyu7123 · 방금 전 · 비공개

0

baekjoon

## Smarcle\_23Algorithm



▼ 목록 보기

7/7



## Baekjoon\_1620 : 나는야 포켓몬 마스터 이다솜

탐색트리로 해결, 삽입과 탐색만 구현

### 풀이 과정

1. 강의 의사코드를 활용하여 삽입, 탐색을 구현
2. 삽입시 인덱스를 이용하여 left, right를 구분하고 노드에 문자열과 인덱스를 저장
3. 탐색 문자열 비교는 strcmp() 함수를 사용
4. M개의 줄에 숫자가 들어오면 숫자를 숫자를 가진 해당 노드의 문자열을 출력, 문자열이 들어오면 문자열을 가진 노드의 숫자를 출력

### 문제 해결

- parent를 사용하지 않아 삽입과정에서 root와 연결이 되지 않음  
탐색하지 않고 루트의 숫자와 넘겨진 숫자를 비교하여 노드를 생성하는 것으로 변경
- M개의 줄에서 문자열과 숫자를 구분할 때 혼란  
ctype.h라이브러리의 isdigit을 사용하여 각 인덱스의 문자가 '1', '2'와 같이 숫자라면 0이 아닌 값이 나오는 것을 사용하여 해결

- 문자열->숫자, 숫자->문자열 입력과 출력의 형식  
탐색함수를 따로 구현하여 문자열 입력시 문자열을 이용하여 비교, 숫자 입력시 숫자를 이용하여 비교하도록 해결
- 숫자->문자열 출력이 (null)  
atoi를 사용하여 문자열을 숫자로 변경하여 호출
- 제출시 시간초과  
인덱스가 0부터 n까지 오른쪽 노드로만 추가되므로 삽입과 탐색과정에서 왼쪽 노드를 제외, gets대신 scanf("%s")사용 변경 -> 시간초과  
구조체만, 배열만 사용해도 시간초과... 해결법 plzπ

## 코드

### 탐색트리 사용1

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

struct Tree {
    char mon[21];
    int num;
    struct Tree* left;
    struct Tree* right;
};

struct Tree* root = NULL;

struct Tree* createNode(int i, char* ch) {
    struct Tree* newNode = (struct Tree*)malloc(sizeof(struct Tree));
    newNode->left = NULL;
    newNode->right = NULL;
    newNode->num = i;
    strcpy(newNode->mon, ch);
    return newNode;
}

void insertMon(int i, char* ch) {
    if (root == NULL)
        root = createNode(i, ch);
    else {
        struct Tree* w = root;
        while (1) {
            if (i < w->num) {
                if (w->left == NULL) {
                    w->left = createNode(i, ch);
                    break;
                }
                w = w->left;
            }
            else {
                if (w->right == NULL) {
```

```

        w->right = createNode(i, ch);
        break;
    }
    w = w->right;
}
}
}

struct Tree* treeSearchMon(struct Tree* w, int n) {
    if (w == NULL) return w;
    if (w->num == n) return w;
    else if (w->num > n) return treeSearchMon(w->left, n);
    else return treeSearchMon(w->right, n);
}

char* findMon(int n) {
    struct Tree* w = treeSearchMon(root, n);
    return w->mon;
}

struct Tree* treeSearchNum(struct Tree* w, char* ch) {
    if (w == NULL) return NULL;
    if (strcmp(w->mon, ch) == 0) return w;
    struct Tree* leftRes = treeSearchNum(w->left, ch);
    if (leftRes != NULL) return leftRes;
    return treeSearchNum(w->right, ch);
}

int findNumber(char* ch) {
    struct Tree* w = treeSearchNum(root, ch);
    return w->num;
}

int main() {

    int n, m;
    char ch[21], ans[21];
    scanf("%d %d", &n, &m);
    getchar();

    for (int i = 0; i < n; i++) {
        gets(ch);
        insertMon(i, ch);
    }

    for (int i = 0; i < m; i++) {
        gets(ans);
        if (ans[0] >= '0' && ans[0] <= '9') {
            int num = atoi(ans);
            printf("%s\n", findMon(num - 1));
        }
        else
            printf("%d\n", findNumber(ans) + 1);
    }
}

```

```
    return 0;
}
```

## 시간초과

### 탐색트리 사용2 (왼쪽 노드 제외)

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

struct Tree {
    char mon[21];
    int num;
    struct Tree* right;
};

struct Tree* root = NULL;

struct Tree* createNode(int i, char* ch) {
    struct Tree* newNode = (struct Tree*)malloc(sizeof(struct Tree));
    newNode->right = NULL;
    newNode->num = i;
    strcpy(newNode->mon, ch);
    return newNode;
}

void insertMon(int i, char* ch) {
    if (root == NULL) {
        root = createNode(i, ch);
        return;
    }

    struct Tree* w = root;
    while (1) {
        if (w->right == NULL) {
            w->right = createNode(i, ch);
            return;
        }
        w = w->right;
    }
}

struct Tree* treeSearchMon(struct Tree* w, int n) {
    if (w->num == n) return w;
    else return treeSearchMon(w->right, n);
}

char* findMon(int n) {
    struct Tree* w = treeSearchMon(root, n);
    return w->mon;
}

struct Tree* treeSearchNum(struct Tree* w, char* ch) {
    if (strcmp(w->mon, ch) == 0) return w;
}
```

```

    return treeSearchNum(w->right, ch);
}

int findNumber(char* ch) {
    struct Tree* w = treeSearchNum(root, ch);
    return w->num;
}

int main() {

    int n, m;
    char ch[21], ans[21];
    scanf("%d %d", &n, &m);
    getchar();

    for (int i = 0; i < n; i++) {
        scanf("%s", ch);
        getchar();
        insertMon(i, ch);
    }

    for (int i = 0; i < m; i++) {
        scanf("%s", ans);
        getchar();
        if (ans[0] >= '0' && ans[0] <= '9') {
            int num = atoi(ans);
            printf("%s\n", findMon(num - 1));
        }
        else
            printf("%d\n", findNumber(ans) + 1);
    }

    return 0;
}

```

## 시간초과

## 구조체만 사용

```

#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

struct pokemon {
    char mon[21];
    int num;
};

void insertMon(struct pokemon* list, int i, char* ch) {
    list->num = i;
    strcpy(list->mon, ch);
}

char* findMon(struct pokemon* list, int i, int n) {
    for (int j = 0; j < n; j++) {

```

```

        if ((list + j)->num == i)
            return (list + j)->mon;
    }
}

int findNum(struct pokemon* list, char* ch, int n) {
    for (int j = 0; j < n; j++) {
        if (strcmp((list + j)->mon, ch) == 0)
            return j;
    }
}

int main() {

    int n, m;
    char ch[21], ans[21];
    scanf("%d %d", &n, &m);

    struct pokemon* list = (struct pokemon*)malloc(n * sizeof(struct pokemon));

    for (int i = 0; i < n; i++) {
        scanf("%s", ch);
        getchar();
        insertMon(list + i, i, ch);
    }

    for (int i = 0; i < m; i++) {
        scanf("%s", ans);
        getchar();
        if (ans[0] >= '0' && ans[0] <= '9') {
            int num = atoi(ans);
            printf("%s\n", (list + num)->mon);
        }
        else
            printf("%d\n", findNum(list, ans, n) + 1);
    }

    return 0;
}

```

시간초과

**배열 사용**

```

#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int main() {

    int n, m;
    scanf("%d %d", &n, &m);
    char** mons = (char**)malloc(n * sizeof(char*));
    for (int i = 0; i < n; i++) {
        mons[i] = (char*)malloc(21 * sizeof(char));
    }
}

```

```
}

for (int i = 0; i < n; i++) {
    scanf("%s", mons[i]);
    getchar();
}

for (int i = 0; i < m; i++) {
    char ans[21];
    scanf("%s", ans);
    getchar();
    if (ans[0] >= '0' && ans[0] <= '9') {
        int num = atoi(ans);
        printf("%s\n", mons[num - 1]);
    }
    else
        for (int j = 0; j < n; j++) {
            if (strcmp(mons[j], ans) == 0) {
                printf("%d\n", j + 1);
                break;
            }
        }
}

free(mons);

return 0;
}
```

시간초과



유정



이전 포스트

**Week5\_과제 (1256)**

0개의 댓글

댓글을 작성하세요

댓글 작성

