13077 - Ranking System

Status (/status/?pid=13077) | Limits Submit (/users/submit/13077)

Description

Ranking system is common in everyday life.
Such as shopping site, mobile game, project contest, etc.

You have a struct of `Node` and a Table store `Node*` :
There're 3 kinds of operation:

- `INSERT score, name` : Add `Node*` with (score, name) into Table.
- `DELETE name` : Delete the `Node*` with name in the Table.
- `TOP x` : Return `int` array contains the indices of top x Nodes in Table.
  The rank of `Node*` is defined below:
  - The higher score, the higher rank
  - For those with same score, ranking by their names in lexicological order.

Your task is to complete these 3 operations in ranking system.
Please trace the `main.c` , `function.h` for the detail interface and implementation.

main.c

```c
#include <stdio.h>
#include "function.h"
#include <string.h>
#include <stdlib.h>

#define MAX_SIZE 1000
#define MAX_LEN 100
int N = 0;
Node* Table[MAX_SIZE];


int main(){
    for(int i=0; i<MAX_SIZE; i++)
        Table[i] = NULL;

    int K;
    scanf("%d", &K);

    char op[10];

    while( K-- ){
        // printf("K: %d\n", K);
        scanf("%s", op);
        if( strcmp(op, "INSERT" ) == 0 ){
            int score;
            char name[MAX_LEN+1];
            scanf("%d %s", &score, name );

            Insert(Table, N, score, name );
            N++;
        }
        else if( strcmp(op, "DELETE" ) == 0 ){
            char name[MAX_LEN+1];
            scanf("%s", name);

            Delete(Table, N, name );
            N--;
        }
        else if( strcmp(op, "TOP" ) == 0 ){
            int x;
            scanf("%d", &x);

            int* idxs = Top(Table, N, x);
            printf("Top %d:\n", x);
            for(int i=0; i<x; i++){
                printf("%d %s\n", Table[idxs[i]]->score, Table[idxs[i]]->name );
            }
            free( idxs );
        }
    }
    for(int i=0; i<MAX_SIZE; i++){
        if( Table[i] != NULL ){
            free(Table[i]->name);
            free(Table[i]);
            Table[i] = NULL;
        }
    }

    return 0;
}
```

functin.h

```c
// function.h
#ifndef __FUNCTION_H__
#define __FUNCTION_H__

typedef struct{
    int score;
    char* name;
} Node;

// Node* Table[MAX_SIZE];
// N = number of nodes in Table

void Insert( Node** Table, int N, int score, char* name );
void Delete( Node** Table, int N, char* name );
int* Top( Node** Table, int N, int x);
#endif
```

Input

There's an integer K on the first line.

There's 1 operation on the each of following K lines.

It's guaranteed that:

- The # of elements in Table will not exceed 1000 during the process
- $1 \leq$ The length of all names ≤ 100
- All names are distinct.

Output

Print the top x students in the Table for each `TOP x` operation.

Sample Input

Download (data:text/plain;charset=utf-8,8%0D%0AINSERT%2010%20John%0D%0AINSERT%2033%20Jojo%0D%0ATOP%202%0D%0AINSERT%2020%20Pual%0

```
8
INSERT 10 John
INSERT 33 Jojo
TOP 2
INSERT 20 Pual
DELETE Jojo
INSERT 20 Sasa
TOP 1
TOP 3
```

Sample Output

Download (data:text/plain;charset=utf-8,Top%202%3A%0D%0A33%20Jojo%0D%0A10%20John%0D%0ATop%201%3A%0D%0A20%20Pual%0D%0ATop%203%3A%

```
Top 2:
33 Jojo
10 John
Top 1:
20 Pual
Top 3:
20 Pual
20 Sasa
10 John
```

Partial Judge Code

13077.c (/problem/partial/13077.c/)

Partial Judge Header

13077.h (/problem/partial/13077.h/)

Discuss