ADA LAB-1

(Leet Code Questions)

Name : Kathasagaram Aishwarya
USN : 1BM22CS123
Section : 4C (B1)

Q) Implement a last-in-first-out (LIFO) stack using only two queues. The implemented stack should support all the functions of a normal stack (push, top, pop, and empty).

Link: https://leetcode.com/problems/implement-stack-using-queues/description/

CODE:

```
typedef struct {
    int q1[101];
    int q2[101];
    int f1,r1,f2,r2;
} MyStack;



MyStack* myStackCreate() {
    MyStack *nn = (MyStack *)malloc(sizeof(MyStack));
    nn->f1=nn->r1=nn->f2=nn->r2=-1;
    return nn;
}

void myStackPush(MyStack* obj, int x) {
    if(obj->f1 == -1){
        obj->f1 = 0;
    }
        obj->q1[++obj->r1] = x;
}
```

```
int myStackPop(MyStack* obj) {
    int k;
    if(obj->f1!=-1){
        for(int i=obj->f1;i<obj->r1;i++){
            if(obj->f2==-1){
                obj->f2++;
            }
            obj->q2[++obj->r2] = obj->q1[i];
        }
        k = obj->q1[obj->r1];
        obj->f1=obj->r1=-1;
    }
    else{
        for(int i=obj->f2;i<obj->r2;i++){
            if(obj->f1==-1){
                obj->f1++;
            }
            obj->q1[++obj->r1] = obj->q2[i];
        }
        k = obj->q2[obj->r2];
        obj->f2=obj->r2=-1;
    }

    return k;
}

int myStackTop(MyStack* obj) {
    if(obj->f1 != -1){
        return obj->q1[obj->r1];
    }
    else if(obj->f2 != -1){
        return obj->q2[obj->r2];
```

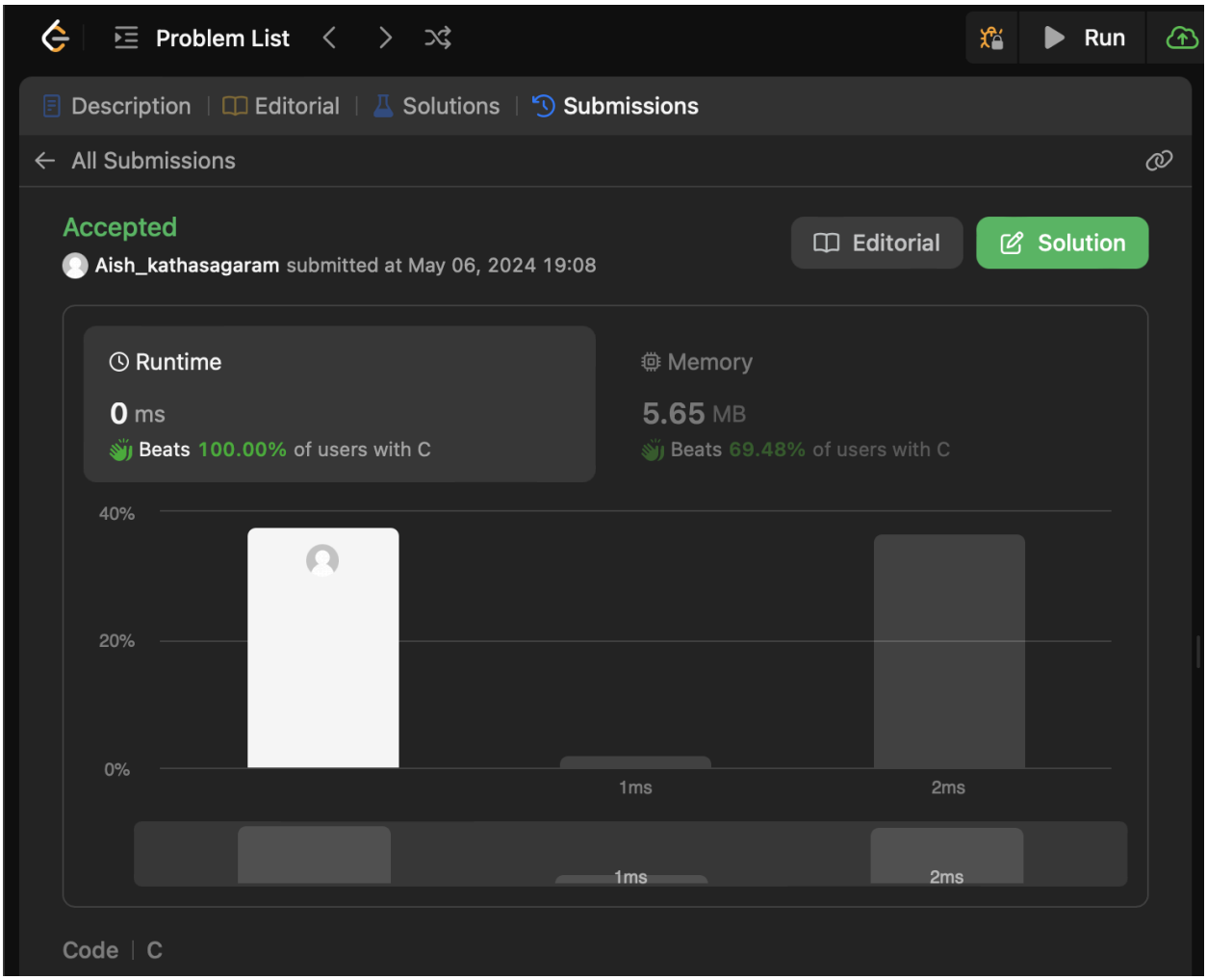
```
    }
    else{
        return -1;
    }
}

bool myStackEmpty(MyStack* obj) {
    return (obj->f1==-1 && obj->f2==-1);
}

void myStackFree(MyStack* obj) {
    free(obj);
}

/**
* Your MyStack struct will be instantiated and called as such:
* MyStack* obj = myStackCreate();
* myStackPush(obj, x);
* int param_2 = myStackPop(obj);
* int param_3 = myStackTop(obj);
* bool param_4 = myStackEmpty(obj);
* myStackFree(obj);
*/
```

OUTPUT:


Problem List
Run
Description
Editorial
Solutions
Submissions
All Submissions
Accepted
Aish_kathasagaram submitted at May 06, 2024 19:08
Editorial
Solution
Runtime
0 ms
Beats 100.00% of users with C
Memory
5.65 MB
Beats 69.48% of users with C
40%
20%
0%
1ms
2ms
1ms
2ms
Code | C


Testcase | Test Result
Case 1
["MyStack","push","push","top","pop","empty"]
[[],[1],[2],[],[],[]]
Source