```c
typedef struct {
    int queue1[100];
    int queue2[100];
    int front1,rear1,front2,rear2;
} MyStack;


MyStack* myStackCreate() {
    MyStack *stack = (MyStack *)malloc(sizeof(MyStack));
    stack->front1=stack->rear1=stack->front2=stack->rear2=-1;
    return stack;
}

void myStackPush(MyStack* obj, int x) {
    if(obj->front1 == -1){
        obj->front1 = 0;
    }
        obj->queue1[++obj->rear1] = x;

}

int myStackPop(MyStack* obj) {
    int pop_ele;
    if(obj->front1!=-1){
        for(int i=obj->front1;i<obj->rear1;i++){
            if(obj->front2==-1){
                obj->front2++;
            }
            obj->queue2[++obj->rear2] = obj->queue1[i];
        }
        pop_ele = obj->queue1[obj->rear1];
        obj->front1=obj->rear1=-1;
    }
    else{
        for(int i=obj->front2;i<obj->rear2;i++){
            if(obj->front1==-1){
                obj->front1++;
            }
            obj->queue1[++obj->rear1] = obj->queue2[i];
        }
        pop_ele = obj->queue2[obj->rear2];
        obj->front2=obj->rear2=-1;
    }

    return pop_ele;
}

int myStackTop(MyStack* obj) {
    if(obj->front1 != -1){
```

```
        return obj->queue1[obj->rear1];
    }
    else if(obj->front2 != -1){
        return obj->queue2[obj->rear2];
    }
    else{
        return -1;
    }
}

bool myStackEmpty(MyStack* obj) {
    return (obj->front1==-1 && obj->front2==-1);
}

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