```
#include <stdbool.h>
#include <stdlib.h>
#include <stdio.h>
typedef struct
{
   int *data;
   int front;
   int rear;
   int size;
} Queue;

typedef struct
{
   Queue *q1;
   Queue *q2;
} MyStack;

Queue *createQueue(int size)
{
   Queue *queue = (Queue *)malloc(sizeof(Queue));
   queue->data = (int *)malloc(size * sizeof(int));
   queue->front = queue->rear = -1;
   queue->size = size;
   return queue;
```

```
}

void enqueue(Queue *queue, int value)
{
   if (queue->rear == -1)
   {
      queue->front = queue->rear = 0;
   }
   else
   {
      queue->rear = (queue->rear + 1) % queue->size;
   }
   queue->data[queue->rear] = value;
}

int dequeue(Queue *queue)
{
   int value = queue->data[queue->front];
   if (queue->front == queue->rear)
   {
      queue->front = queue->rear = -1;
   }
   else
   {
```

```
        queue->front = (queue->front + 1) % queue->size;
    }
    return value;
}

bool isEmpty(Queue *queue)
{
    return queue->front == -1;
}

MyStack *myStackCreate()
{
    MyStack *stack = (MyStack *)malloc(sizeof(MyStack));
    stack->q1 = createQueue(1000); // Adjust the size as needed
    stack->q2 = createQueue(1000);
    return stack;
}

void myStackPush(MyStack *obj, int x)
{
    enqueue(obj->q1, x);
}

int myStackPop(MyStack *obj)
```

```
{
    if (isEmpty(obj->q1))
    {
        return -1; // Stack is empty
    }

    while (obj->q1->front != obj->q1->rear)
    {
        enqueue(obj->q2, dequeue(obj->q1));
    }

    int poppedValue = dequeue(obj->q1);

    // Swap q1 and q2
    Queue *temp = obj->q1;
    obj->q1 = obj->q2;
    obj->q2 = temp;

    return poppedValue;
}

int myStackTop(MyStack *obj)
{
    if (isEmpty(obj->q1))
```

```
    {
        return -1; // Stack is empty
    }

    while (obj->q1->front != obj->q1->rear)
    {
        enqueue(obj->q2, dequeue(obj->q1));
    }

    int topValue = dequeue(obj->q1);
    enqueue(obj->q2, topValue);

    // Swap q1 and q2
    Queue *temp = obj->q1;
    obj->q1 = obj->q2;
    obj->q2 = temp;

    return topValue;
}

bool myStackEmpty(MyStack *obj)
{
    return isEmpty(obj->q1);
}
```

```
void myStackFree(MyStack *obj)
{
    free(obj->q1->data);
    free(obj->q1);
    free(obj->q2->data);
    free(obj->q2);
    free(obj);
}

void main()
{
    MyStack *obj = myStackCreate();
    myStackPush(obj, 1);

    int a = myStackPop(obj);

    int b = myStackTop(obj);

    bool c = myStackEmpty(obj);

    myStackFree(obj);

    printf("Popped element:%d\nTop element:%d\nIsempty?:%d\t", a, b, c);
```

}

```
C:\Users\shrey\VSCODE\Python>gcc qtos.c

C:\Users\shrey\VSCODE\Python>a.exe
Popped element:1
Top element:-1
Isempty?:1
```