**Implement Stack using Queues**

typedef struct { int\* q1; int\* q2; int f1, f2, r1, r2;} MyStack;

MyStack\* myStackCreate() { MyStack\* st = (MyStack\*)malloc(sizeof(MyStack)); st->q1 = (int\*)calloc(10, sizeof(int)); st->q2 = (int\*)calloc(10, sizeof(int)); st->f1 = -1; st->f2 = -1; st->r1 = -1; st->r2 = -1; return st;}

void myStackPush(MyStack\* obj, int x) { if (obj->f1 == -1 && obj->r1 == -1) { obj->f1 = 0; obj->r1 = 0; } else { obj->r1++; } printf("%d\n", x); obj->q1[obj->r1] = x;}int myStackPop(MyStack\* obj) { if (obj->f1 == -1) { return -1; } int k1 = obj->f1; int l1 = obj->r1; int k2 = obj->f2; int l2 = obj->r2; int ch; while (k1 < l1) { if (k2 == -1) { k2 = 0; l2 = 0; } else { l2++; } obj->q2[l2] = obj->q1[k1]; k1++; } ch = obj->q1[k1]; k1=-1; l1=-1; int\* temp = obj->q1; obj->q1 = obj->q2; obj->q2 = temp; obj->f1 = k2; obj->f2 = k1; obj->r1 = l2; obj->r2 = l1; if(obj->r1<obj->f1){ obj->r1=-1; obj->f1=-1; } return ch;}int myStackTop(MyStack\* obj) { if (obj->f1 == -1) { return -1; } int k1 = obj->f1; int l1 = obj->r1; int k2 = obj->f2; int l2 = obj->r2; int ch; while (k1 <= l1) { if (k2 == -1) { k2 = 0; l2 = 0; } else { l2++; } ch = obj->q1[k1]; obj->q2[l2] = obj->q1[k1]; k1++; } int\* temp = obj->q1; obj->q1 = obj->q2; obj->q2 = temp; return ch;}bool myStackEmpty(MyStack\* obj) { return (obj->f1 == -1);}void myStackFree(MyStack\* obj) { free(obj->q1); free(obj->q2); free(obj);}



