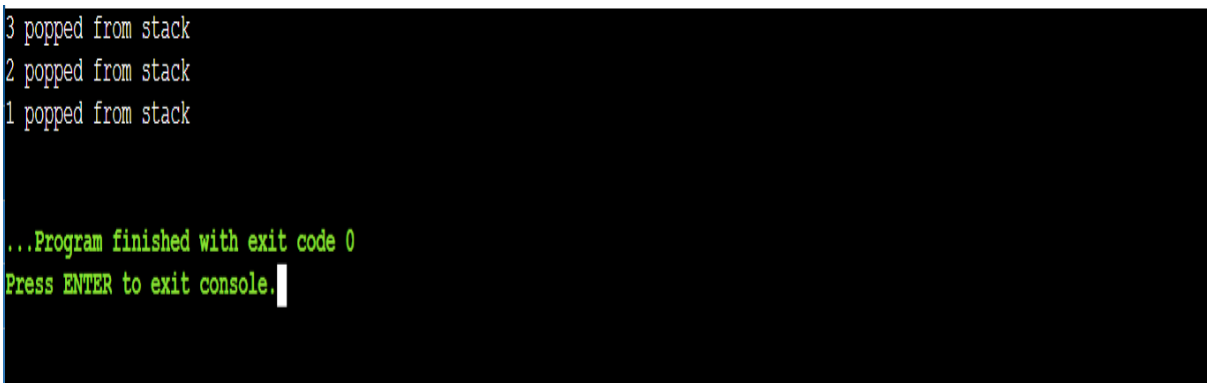
```
#include <stdio.h>
#include <stdlib.h>

// Queue implementation using linked list
struct Node {
    int data;
    struct Node* next;
};

struct Queue {
    struct Node *front, *rear;
};

struct Node* newNode(int data) {
    struct Node* temp = (struct Node*)malloc(sizeof(struct Node));
    temp->data = data;
    temp->next = NULL;
    return temp;
}

struct Queue* createQueue() {
    struct Queue* q = (struct Queue*)malloc(sizeof(struct Queue));
    q->front = q->rear = NULL;
    return q;
}

void enQueue(struct Queue* q, int data) {
    struct Node* temp = newNode(data);

    if (q->rear == NULL) {
        q->front = q->rear = temp;
        return;
    }

    q->rear->next = temp;
    q->rear = temp;
}

int deQueue(struct Queue* q) {
    if (q->front == NULL)
        return -1;

    int data = q->front->data;
    struct Node* temp = q->front;

    q->front = q->front->next;

    if (q->front == NULL)
        q->rear = NULL;

    free(temp);
    return data;
}

// Stack implementation using two queues
struct Stack {
    struct Queue* q1;
    struct Queue* q2;
};

struct Stack* createStack() {
    struct Stack* stack = (struct Stack*)malloc(sizeof(struct Stack));
```

```c
    stack->q1 = createQueue();
    stack->q2 = createQueue();
    return stack;
}

void push(struct Stack* stack, int data) {
    enQueue(stack->q1, data);
}

int pop(struct Stack* stack) {
    if (stack->q1->front == NULL)
        return -1;

    // Move elements from q1 to q2 except the last one
    while (stack->q1->front->next != NULL) {
        enQueue(stack->q2, deQueue(stack->q1));
    }

    // Pop the last element from q1
    int popped = deQueue(stack->q1);

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

    return popped;
}

int main() {
    struct Stack* stack = createStack();
    push(stack, 1);
    push(stack, 2);
    push(stack, 3);

    printf("%d popped from stack\n", pop(stack));
    printf("%d popped from stack\n", pop(stack));
    printf("%d popped from stack\n", pop(stack));

    return 0;
}
```



```
3 popped from stack
2 popped from stack
1 popped from stack


...Program finished with exit code 0
Press ENTER to exit console.
```