#include <stdio.h>

#include <stdlib.h>

#include <limits.h>

struct Stack {

int data;

struct Stack \*next;

};

struct Stack \*CreateStack () {

return NULL;

}

int isEmptyStack(struct Stack \*top) {

return (top == NULL);

}

void Push(struct Stack \*\*top, int data) {

struct Stack \*newNode = (struct Stack\*) malloc(sizeof(struct Stack));

if(!newNode)

return;

newNode->data = data;

newNode->next = \*top;

\*top = newNode;

}

int Pop(struct Stack \*\*top) {

struct Stack \*temp;

int data;

if(isEmptyStack(\*top)) {

printf("Empty Stack.\n");

return INT\_MIN;

}

temp = \*top;

data = (\*top)->data;

\*top = (\*top)->next;

free(temp);

return data;

}

struct Queue {

struct Stack \*S1;

struct Stack \*S2;

};

struct Queue \*CreateQueue() {

return NULL;

}

void EnQueue(struct Queue \*Q, int data) {

Push(Q->S1, data);

}

int DeQueue(struct Queue \*Q) {

if(!isEmptyStack(Q->S2)) {

return Pop(Q->S2);

}

else {

while(!isEmptyStack(Q->S1)) {

Push(Q->S2, Pop(Q->S1));

}

return Pop(Q->S2);

}

}

int main() {

struct Queue \*Q = CreateQueue();

Q->S1 = Q->S2 = NULL;

EnQueue(Q, 1);

EnQueue(Q, 2);

EnQueue(Q, 3);

printf("%d ", DeQueue(Q));

printf("%d ", DeQueue(Q));

printf("%d ", DeQueue(Q));

return 0;

}