Code : 1  
package Topic\_15\_Queues;

import java.io.BufferedReader;

import java.io.InputStreamReader;

public class A\_NormalQueue {

public static class CustomQueue {

int[] data;

int front;

int size;

public CustomQueue(int cap) {

data = new int[cap];

front = 0;

size = 0;

}

int size() {

return size;

}

void display() {

for (int i = 0; i < size; i++) {

int idx = (front + i) % data.length;

System.out.print(data[idx] + " ");

}

System.out.println();

}

void add(int val) {

if (size == data.length) {

System.out.println("Queue overflow");

} else {

int idx = (front + size) % data.length;

data[idx] = val;

size++;

}

}

int remove() {

if (size == 0) {

System.out.println("Queue underflow");

return -1;

} else {

int val = data[front];

front = (front + 1) % data.length;

size--;

return val;

}

}

int peek() {

if (size == 0) {

System.out.println("Queue underflow");

return -1;

} else {

int val = data[front];

return val;

}

}

}

public static void main(String[] args) throws Exception {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

int n = Integer.parseInt(br.readLine());

CustomQueue qu = new CustomQueue(n);

String str = br.readLine();

while (str.equals("quit") == false) {

if (str.startsWith("add")) {

int val = Integer.parseInt(str.split(" ")[1]);

qu.add(val);

} else if (str.startsWith("remove")) {

int val = qu.remove();

if (val != -1) {

System.out.println(val);

}

} else if (str.startsWith("peek")) {

int val = qu.peek();

if (val != -1) {

System.out.println(val);

}

} else if (str.startsWith("size")) {

System.out.println(qu.size());

} else if (str.startsWith("display")) {

qu.display();

}

str = br.readLine();

}

}

}

Code : 2  
package Topic\_15\_Queues;

import java.io.BufferedReader;

import java.io.InputStreamReader;

public class B\_DynamicQueue {

public static class CustomQueue {

int[] data;

int front;

int size;

public CustomQueue(int cap) {

data = new int[cap];

front = 0;

size = 0;

}

int size() {

// write ur code here

return size;

}

void display() {

// write ur code here

for (int i = 0; i < size; i++) {

int idx = (front + i) % data.length;

System.out.print(data[idx] + " ");

}

System.out.println();

}

// change this code

void add(int val) {

// write ur code here

if (size == data.length) {

int[] ndata = new int[2 \* data.length];

for (int i = 0; i < size; i++) {

int idx = (front + i) % data.length;

ndata[i] = data[idx];

}

data = ndata;

front = 0;

int idx = (front + size) % data.length;

data[idx] = val;

size++;

} else {

int idx = (front + size) % data.length;

data[idx] = val;

size++;

}

}

int remove() {

// write ur code here

if (size == 0) {

System.out.println("Queue underflow");

return -1;

} else {

int val = data[front];

front = (front + 1) % data.length;

size--;

return val;

}

}

int peek() {

// write ur code here

if (size == 0) {

System.out.println("Queue underflow");

return -1;

} else {

int val = data[front];

return val;

}

}

}

public static void main(String[] args) throws Exception {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

int n = Integer.parseInt(br.readLine());

CustomQueue qu = new CustomQueue(n);

String str = br.readLine();

while (str.equals("quit") == false) {

if (str.startsWith("add")) {

int val = Integer.parseInt(str.split(" ")[1]);

qu.add(val);

} else if (str.startsWith("remove")) {

int val = qu.remove();

if (val != -1) {

System.out.println(val);

}

} else if (str.startsWith("peek")) {

int val = qu.peek();

if (val != -1) {

System.out.println(val);

}

} else if (str.startsWith("size")) {

System.out.println(qu.size());

} else if (str.startsWith("display")) {

qu.display();

}

str = br.readLine();

}

}

}

Code : 3  
package Topic\_15\_Queues;

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.util.ArrayDeque;

import java.util.Queue;

public class C\_QueueToStackAdapterPushEfficient {

public static class QueueToStackAdapter {

Queue<Integer> mainQ;

Queue<Integer> helperQ;

public QueueToStackAdapter() {

mainQ = new ArrayDeque<>();

helperQ = new ArrayDeque<>();

}

int size() {

return mainQ.size();

}

void push(int val) {

mainQ.add(val);

}

int pop() {

if (size() == 0) {

System.out.println("Stack underflow");

return -1;

} else {

while (mainQ.size() > 1) {

helperQ.add(mainQ.remove());

}

int val = mainQ.remove();

while (helperQ.size() > 0) {

mainQ.add(helperQ.remove());

}

return val;

}

}

int top() {

if (size() == 0) {

System.out.println("Stack underflow");

return -1;

} else {

while (mainQ.size() > 1) {

helperQ.add(mainQ.remove());

}

int val = mainQ.remove();

helperQ.add(val);

while (helperQ.size() > 0) {

mainQ.add(helperQ.remove());

}

return val;

}

}

}

public static void main(String[] args) throws Exception {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

QueueToStackAdapter st = new QueueToStackAdapter();

String str = br.readLine();

while (str.equals("quit") == false) {

if (str.startsWith("push")) {

int val = Integer.parseInt(str.split(" ")[1]);

st.push(val);

} else if (str.startsWith("pop")) {

int val = st.pop();

if (val != -1) {

System.out.println(val);

}

} else if (str.startsWith("top")) {

int val = st.top();

if (val != -1) {

System.out.println(val);

}

} else if (str.startsWith("size")) {

System.out.println(st.size());

}

str = br.readLine();

}

}

}

Code : 4  
package Topic\_15\_Queues;

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.util.ArrayDeque;

import java.util.Queue;

public class D\_QueueToStackAdapterPopEfficient {

public static class QueueToStackAdapter {

Queue<Integer> mainQ;

Queue<Integer> helperQ;

public QueueToStackAdapter() {

mainQ = new ArrayDeque<>();

helperQ = new ArrayDeque<>();

}

int size() {

return mainQ.size();

}

void push(int val) {

while (mainQ.size() > 0) {

helperQ.add(mainQ.remove());

}

mainQ.add(val);

while (helperQ.size() > 0) {

mainQ.add(helperQ.remove());

}

}

int pop() {

if (size() == 0) {

System.out.println("Stack underflow");

return -1;

} else {

return mainQ.remove();

}

}

int top() {

if (size() == 0) {

System.out.println("Stack underflow");

return -1;

} else {

return mainQ.peek();

}

}

}

public static void main(String[] args) throws Exception {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

QueueToStackAdapter st = new QueueToStackAdapter();

String str = br.readLine();

while (str.equals("quit") == false) {

if (str.startsWith("push")) {

int val = Integer.parseInt(str.split(" ")[1]);

st.push(val);

} else if (str.startsWith("pop")) {

int val = st.pop();

if (val != -1) {

System.out.println(val);

}

} else if (str.startsWith("top")) {

int val = st.top();

if (val != -1) {

System.out.println(val);

}

} else if (str.startsWith("size")) {

System.out.println(st.size());

}

str = br.readLine();

}

}

}

Code : 5  
package Topic\_15\_Queues;

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.util.Stack;

public class E\_StackToQueueAdapterAddEfficient {

public static class StackToQueueAdapter {

Stack<Integer> mainS;

Stack<Integer> helperS;

public StackToQueueAdapter() {

mainS = new Stack<>();

helperS = new Stack<>();

}

int size() {

return mainS.size();

}

void add(int val) {

mainS.push(val);

}

int remove() {

if (size() == 0) {

System.out.println("Queue underflow");

return -1;

} else {

while (mainS.size() > 1) {

helperS.push(mainS.pop());

}

int val = mainS.pop();

while (helperS.size() > 0) {

mainS.push(helperS.pop());

}

return val;

}

}

int peek() {

if (size() == 0) {

System.out.println("Queue underflow");

return -1;

} else {

while (mainS.size() > 1) {

helperS.push(mainS.pop());

}

int val = mainS.pop();

helperS.push(val);

while (helperS.size() > 0) {

mainS.push(helperS.pop());

}

return val;

}

}

}

public static void main(String[] args) throws Exception {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

StackToQueueAdapter qu = new StackToQueueAdapter();

String str = br.readLine();

while (str.equals("quit") == false) {

if (str.startsWith("add")) {

int val = Integer.parseInt(str.split(" ")[1]);

qu.add(val);

} else if (str.startsWith("remove")) {

int val = qu.remove();

if (val != -1) {

System.out.println(val);

}

} else if (str.startsWith("peek")) {

int val = qu.peek();

if (val != -1) {

System.out.println(val);

}

} else if (str.startsWith("size")) {

System.out.println(qu.size());

}

str = br.readLine();

}

}

}

Code : 6  
package Topic\_15\_Queues;

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.util.Stack;

public class F\_StackToQueueAdapterRemoveEfficient {

public static class StackToQueueAdapter {

Stack<Integer> mainS;

Stack<Integer> helperS;

public StackToQueueAdapter() {

mainS = new Stack<>();

helperS = new Stack<>();

}

int size() {

return mainS.size();

}

void add(int val) {

while (mainS.size() > 0) {

helperS.push(mainS.pop());

}

mainS.push(val);

while (helperS.size() > 0) {

mainS.push(helperS.pop());

}

}

int remove() {

if (size() == 0) {

System.out.println("Queue underflow");

return -1;

} else {

return mainS.pop();

}

}

int peek() {

if (size() == 0) {

System.out.println("Queue underflow");

return -1;

} else {

return mainS.peek();

}

}

}

public static void main(String[] args) throws Exception {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

StackToQueueAdapter qu = new StackToQueueAdapter();

String str = br.readLine();

while (str.equals("quit") == false) {

if (str.startsWith("add")) {

int val = Integer.parseInt(str.split(" ")[1]);

qu.add(val);

} else if (str.startsWith("remove")) {

int val = qu.remove();

if (val != -1) {

System.out.println(val);

}

} else if (str.startsWith("peek")) {

int val = qu.peek();

if (val != -1) {

System.out.println(val);

}

} else if (str.startsWith("size")) {

System.out.println(qu.size());

}

str = br.readLine();

}

}

}

Code : 7  
package Topic\_15\_Queues;

import java.io.BufferedReader;

import java.io.InputStreamReader;

public class G\_TwoStacksInAnArray {

public static class TwoStack {

int[] data;

int tos1;

int tos2;

public TwoStack(int cap) {

data = new int[cap];

tos1 = -1;

tos2 = data.length;

}

int size1() {

return tos1 + 1;

}

int size2() {

return data.length - tos2;

}

void push1(int val) {

if (tos2 == tos1 + 1) {

System.out.println("Stack overflow");

} else {

tos1++;

data[tos1] = val;

}

}

void push2(int val) {

if (tos2 == tos1 + 1) {

System.out.println("Stack overflow");

} else {

tos2--;

data[tos2] = val;

}

}

int pop1() {

if (size1() == 0) {

System.out.println("Stack underflow");

return -1;

} else {

int val = data[tos1];

tos1--;

return val;

}

}

int pop2() {

if (size2() == 0) {

System.out.println("Stack underflow");

return -1;

} else {

int val = data[tos2];

tos2++;

return val;

}

}

int top1() {

if (size1() == 0) {

System.out.println("Stack underflow");

return -1;

} else {

int val = data[tos1];

return val;

}

}

int top2() {

if (size2() == 0) {

System.out.println("Stack underflow");

return -1;

} else {

int val = data[tos2];

return val;

}

}

}

public static void main(String[] args) throws Exception {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

int n = Integer.parseInt(br.readLine());

TwoStack st = new TwoStack(n);

String str = br.readLine();

while (str.equals("quit") == false) {

if (str.startsWith("push1")) {

int val = Integer.parseInt(str.split(" ")[1]);

st.push1(val);

} else if (str.startsWith("pop1")) {

int val = st.pop1();

if (val != -1) {

System.out.println(val);

}

} else if (str.startsWith("top1")) {

int val = st.top1();

if (val != -1) {

System.out.println(val);

}

} else if (str.startsWith("size1")) {

System.out.println(st.size1());

} else if (str.startsWith("push2")) {

int val = Integer.parseInt(str.split(" ")[1]);

st.push2(val);

} else if (str.startsWith("pop2")) {

int val = st.pop2();

if (val != -1) {

System.out.println(val);

}

} else if (str.startsWith("top2")) {

int val = st.top2();

if (val != -1) {

System.out.println(val);

}

} else if (str.startsWith("size2")) {

System.out.println(st.size2());

}

str = br.readLine();

}

}

}