Aim:

Create multiple threads to access the contents of a stack. Synchronize thread to prevent simultaneous access to push and pop operations.

Note: Please don't change the package name.

Source Code:

q29795/StackThreads.java

```java
package q29795;
import java.util.*;
class Stack {
    int tos;
    int stck[];
    int size;
    Stack(int size) {
        this.size=size;
        tos=-1;
        stck=new int[this.size];
    }
    synchronized void push(int item) {
        if(tos==stck.length-1) {
            // use length member
            System.out.println("Stack is full");
        }
        else {
            stck[++tos] = item;
        }
    }
    // Pop an item from the stack
    synchronized int pop() {
        if(tos < 0) {

            System.out.println("Stack underflow");
            return 0;
        }
        else
            return stck[tos--];
    }
}

class PushThread extends Thread {
    Stack s;
    PushThread(Stack s) {
        this.s=s;
    }
    public void run() {
        for(int i=1;i<=s.size;i++) {
            s.push(i);
            try {
```

```java
                Thread.sleep(100);
            }
            catch(Exception e) {
                System.out.println(e);
            }
        }
    }
}
class PopThread extends Thread {

    Stack s;
    PopThread(Stack s){
        this.s=s;
    }
    public void run() {
        for(int i=1;i<=s.size;i++) {
            System.out.println(s.pop());
            try {
                Thread.sleep(100);
            }
            catch(Exception e) {
                System.out.println(e);
            }
        }
    }
}

public class StackThreads {
    public static void main(String args[]) {
        int size;
        Scanner sc =new Scanner(System.in);
        System.out.println("Enter the size of the stack");
        size=sc.nextInt();
        Stack s = new Stack(size);//only one object
        PushThread t1=new PushThread(s);
        PopThread t2=new PopThread(s);
        t1.start();
        t2.start();
        t2.setPriority(9);

    }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter the size of the stack 4
1
2
3
4

Test Case - 2
User Output
Enter the size of the stack 9
1
2
3
4
5
6
7
8
9