a) **Aim:** Write a Java program that correctly implements the producer-consumer problem using the concept of inter thread communication.

**Description:**

- The producer produces the item and the consumer consumes the same. But here, the consumer can not consume until the producer produces the item, and producer can not produce until the consumer consumes the item that already been produced.
- So here, the consumer has to wait until the producer produces the item, and the producer also needs to wait until the consumer consumes the same.

**Program:**

```
import java.util.LinkedList;

public class ProduceConsumer
{
    public static void main(String[] args)   throws InterruptedException
   {
        final PC pc = new PC();      // Object of a class that has both produce()  and consume() methods


         Thread t1 = new Thread(new Runnable()   // Create producer thread
         {
                        public void run()
                        {
                                    try
                                    {
                                          pc.produce();
                                    }
                                    catch (InterruptedException e)
                                    {
                                       e.printStackTrace();
                                    }
                 }
          });
            Thread t2 = new Thread(new Runnable()   // Create consumer thread
            {
                  public void run()
                   {
                              try
                            {
                                        pc.consume();
                              }
                              catch (InterruptedException e)
                            {
                                  e.printStackTrace();
                             }
                     }
      });
    // Start both threads

      t1.start();

      t2.start();
```

```
        t1.join();          // t1 finishes before t2
        t2.join();
    }
    static class PC       // This class has a list, producer (adds items to list    and consumer (removes items)
    {
        LinkedList<Integer> list = new LinkedList<>();
        int capacity = 2;

        public void produce() throws InterruptedException    // Function called by producer thread
        {
                int value = 0;
                while (true)
                {
                    synchronized (this)   // producer thread waits while list    is full
                    {
                        while (list.size() == capacity)
                        wait();

                        System.out.println("Producer produced-"+ value);

                        list.add(value++);          // to insert the jobs in the list
                        notify();                   //notifies the consumer thread that    now it can start consuming
                        Thread.sleep(1000);    // makes the working of program easier  to  understand
                    }
                }
        }
        public void consume() throws InterruptedException     // Function called by consumer thread
        {
            while (true)
            {
                synchronized (this)     // consumer thread waits while list   is empty
                {
                while (list.size() == 0)
                wait();
                int val = list.removeFirst();   // to retrieve the first job in the list
                System.out.println("Consumer consumed-"+ val);
                notify();      // Wake up producer thread   and sleep
                Thread.sleep(1000);
            }
        }
    }
}
```

**Output:**

```
D:\ACEM\II CSE  Bsection>javac ProduceConsumer.java
D:\ACEM\II CSE  Bsection>java ProduceConsumer
Producer produced-0
Producer produced-1
Consumer consumed-0
Consumer consumed-1
Producer produced-2
Producer produced-3
Consumer consumed-2
Consumer consumed-3
Producer produced-4
Producer produced-5
```

**b) Aim:** Develop a Java application for stack operation using Buttons and JOptionPane input and Message dialog box.

**Description:**

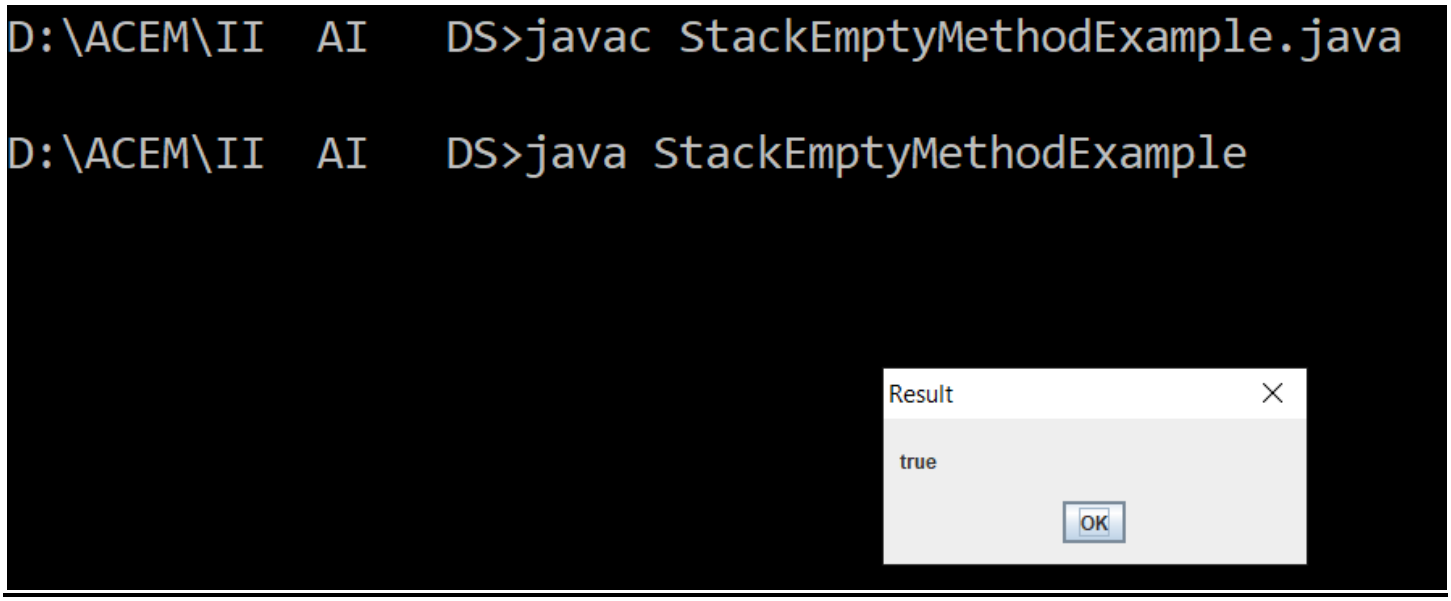
- This dialogbox defined in JOptionPane class in Java is used when you wish to get some input from the user.
- Like if you want user to enter his bank account number to display his transaction details you can use showInputDialog to prompt user.
- After entering the value in the textbox when user presses OK button the input gets stored in a string variable on left hand variable of the assignment operator.

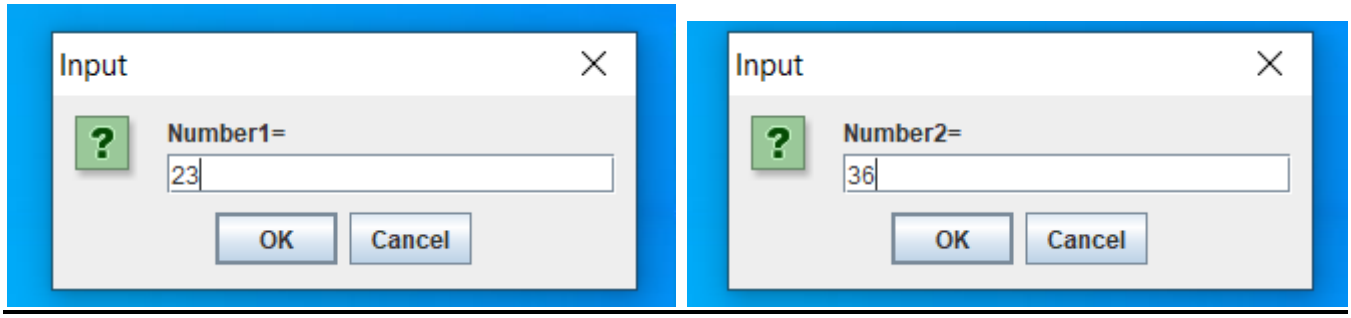
**Program:**

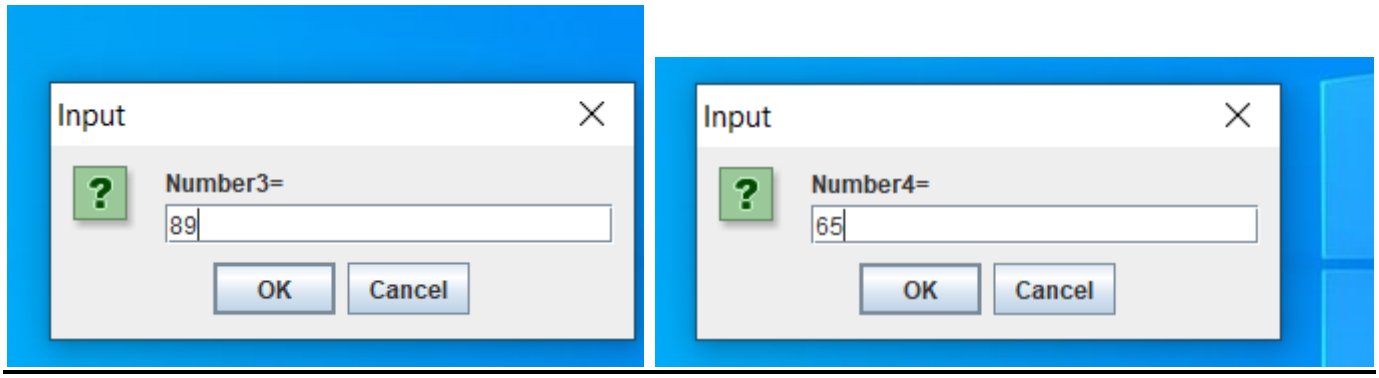
```
import java.util.Stack;
import javax.swing.JOptionPane;
public class StackEmptyMethodExample
{
        public static void main(String[] args)
        {
                Stack<Integer> stk= new Stack<>();
                boolean result = stk.empty();      // checking stack is empty or not
                JOptionPane.showMessageDialog(null,result,"Result", JOptionPane.PLAIN_MESSAGE);
```

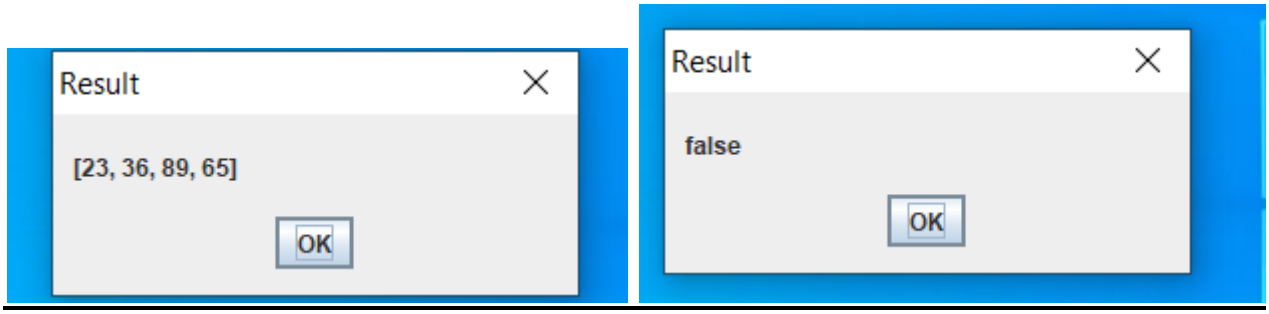
```
// pushing elements into stack
                stk.push(Integer.parseInt(JOptionPane.showInputDialog("Number1=")));
                stk.push(Integer.parseInt(JOptionPane.showInputDialog("Number2=")));
                stk.push(Integer.parseInt(JOptionPane.showInputDialog("Number3=")));
                stk.push(Integer.parseInt(JOptionPane.showInputDialog("Number4=")));
//prints elements of the stack
                JOptionPane.showMessageDialog(null,stk,"Result", JOptionPane.PLAIN_MESSAGE);
                result = stk.empty();
                JOptionPane.showMessageDialog(null,result,"Result", JOptionPane.PLAIN_MESSAGE);
        }
}
```

**Output:**






**c) Aim:** Develop a Java application to perform Addition, Division, Multiplication and subtraction usingthe JOptionPane dialog Box and Textfields

**Description:**

- This dialogbox defined in JOptionPane class in Java is used when you wish to get some input from the user.
- Like if you want user to enter his bank account number to display his transaction details you can use showInputDialog to prompt user.
- After entering the value in the textbox when user presses OK button the input gets stored in a string variable on left hand variable of the assignment operator.
- We can perform mathematical operations using operators and use showMessageDialog to display output

**Program:**

```
import javax.swing.JOptionPane;
public class Calculator
{
        public static void main(String args[])
        {
                int num1;
                int num2;
                int sum;
                int sub;
                int mul;
                int div;
                String Res;
```

```
        num1=Integer.parseInt(JOptionPane.showInputDialog("Number1="));
        num2=Integer.parseInt(JOptionPane.showInputDialog("Number2="));

       sum=num1+num2;
       sub=num1-num2;
       mul=num1*num2;
       div=num1/num2;

    Res="The Sum of "+num1+" and "+num2+" is " +sum ;
    Res=Res +"\n The DIfference of "+num1+" and "+num2+" is " +sub;
    Res=Res +"\n The Product of "+num1+" and "+num2+" is " +mul;
    Res=Res +"\n The Division of "+num1+" by "+num2+" is " +div;
    JOptionPane.showMessageDialog(null,Res,"Result", JOptionPane.PLAIN_MESSAGE);
}
}
```

**Output:**