

Contents

Write a JAVA program to illustrate threading using the producer consumer problem. The buffer size is fixed. The producer produces items but not when buffer is full and consumer consumes items but not when buffer is empty.....	2
Solution	2
Code	2
Output	4

Write a JAVA program to illustrate threading using the producer consumer problem. The buffer size is fixed. The producer produces items but not when buffer is full and consumer consumes items but not when buffer is empty.

Solution

The 'Thread' class in *java.lang* package provides various methods to control the behaviour of threads in an application.

To implement the problem we are using the *wait()* and *notify()* methods for the inter-thread communication that are part of the *java.lang.Object* class.

Code

```
package threadConsumerProducer;

import java.util.*;

public class ConsumerProducer extends Thread {
    boolean prod, cons;
    static Buffer b = new Buffer();

    ConsumerProducer(boolean p) {
        prod = p;
        cons = !p;
    }

    public void run() {
        try {
            if (prod)
                b.producer();
            else
                b.consumer();
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }

    public static void main(String[] args) {
        ConsumerProducer producer = new ConsumerProducer(true);
        ConsumerProducer consumer = new ConsumerProducer(false);

        producer.start();
        consumer.start();
    }
}

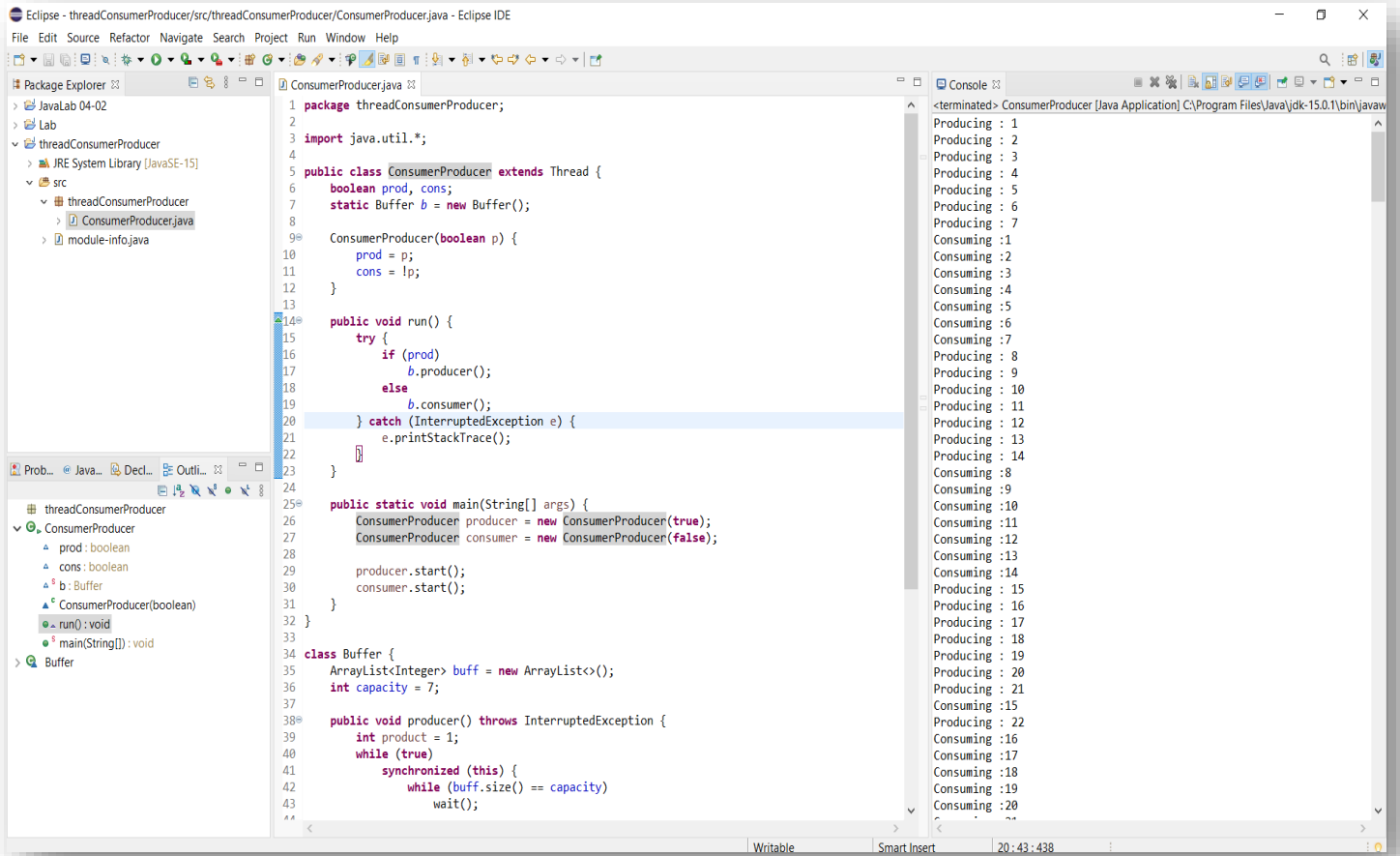
class Buffer {
    ArrayList<Integer> buff = new ArrayList<>();
    int capacity = 7;
}
```

```
public void producer() throws InterruptedException {
    int product = 1;
    while (true)
        synchronized (this) {
            while (buff.size() == capacity)
                wait();

            System.out.println("Producing : " + product);
            buff.add(product++);
            notify();
            Thread.sleep(500);
        }
}

public void consumer() throws InterruptedException {
    while (true)
        synchronized (this) {
            while (buff.size() == 0)
                wait();
            System.out.println("Consuming :" + buff.remove(0));
            notify();
            Thread.sleep(500);
        }
}
}
```

Output



The screenshot shows the Eclipse IDE with the following components:

- Package Explorer:** Shows the project structure with 'threadConsumerProducer' and its sub-packages 'src' and 'module-info.java'.
- Source Editor:** Displays the code for 'ConsumerProducer.java'. The code includes a package declaration, imports, a class definition for 'ConsumerProducer' that extends 'Thread', and a 'Buffer' class. The 'run()' method contains a loop that alternates between producing and consuming items from a buffer.
- Console:** Shows the output of the program, which is a sequence of 'Producing' and 'Consuming' messages, indicating the execution of the producer-consumer algorithm.

```
1 package threadConsumerProducer;
2
3 import java.util.*;
4
5 public class ConsumerProducer extends Thread {
6     boolean prod, cons;
7     static Buffer b = new Buffer();
8
9     ConsumerProducer(boolean p) {
10         prod = p;
11         cons = !p;
12     }
13
14     public void run() {
15         try {
16             if (prod)
17                 b.producer();
18             else
19                 b.consumer();
20         } catch (InterruptedException e) {
21             e.printStackTrace();
22         }
23     }
24
25     public static void main(String[] args) {
26         ConsumerProducer producer = new ConsumerProducer(true);
27         ConsumerProducer consumer = new ConsumerProducer(false);
28
29         producer.start();
30         consumer.start();
31     }
32 }
33
34 class Buffer {
35     ArrayList<Integer> buff = new ArrayList<>();
36     int capacity = 7;
37
38     public void producer() throws InterruptedException {
39         int product = 1;
40         while (true)
41             synchronized (this) {
42                 while (buff.size() == capacity)
43                     wait();
```

Console Output:

```
<terminated> ConsumerProducer [Java Application] C:\Program Files\Java\jdk-15.0.1\bin\javaw
Producing : 1
Producing : 2
Producing : 3
Producing : 4
Producing : 5
Producing : 6
Producing : 7
Consuming : 1
Consuming : 2
Consuming : 3
Consuming : 4
Consuming : 5
Consuming : 6
Consuming : 7
Producing : 8
Producing : 9
Producing : 10
Producing : 11
Producing : 12
Producing : 13
Producing : 14
Consuming : 8
Consuming : 9
Consuming : 10
Consuming : 11
Consuming : 12
Consuming : 13
Consuming : 14
Producing : 15
Producing : 16
Producing : 17
Producing : 18
Producing : 19
Producing : 20
Producing : 21
Consuming : 15
Consuming : 16
Consuming : 17
Consuming : 18
Consuming : 19
Consuming : 20
```