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.

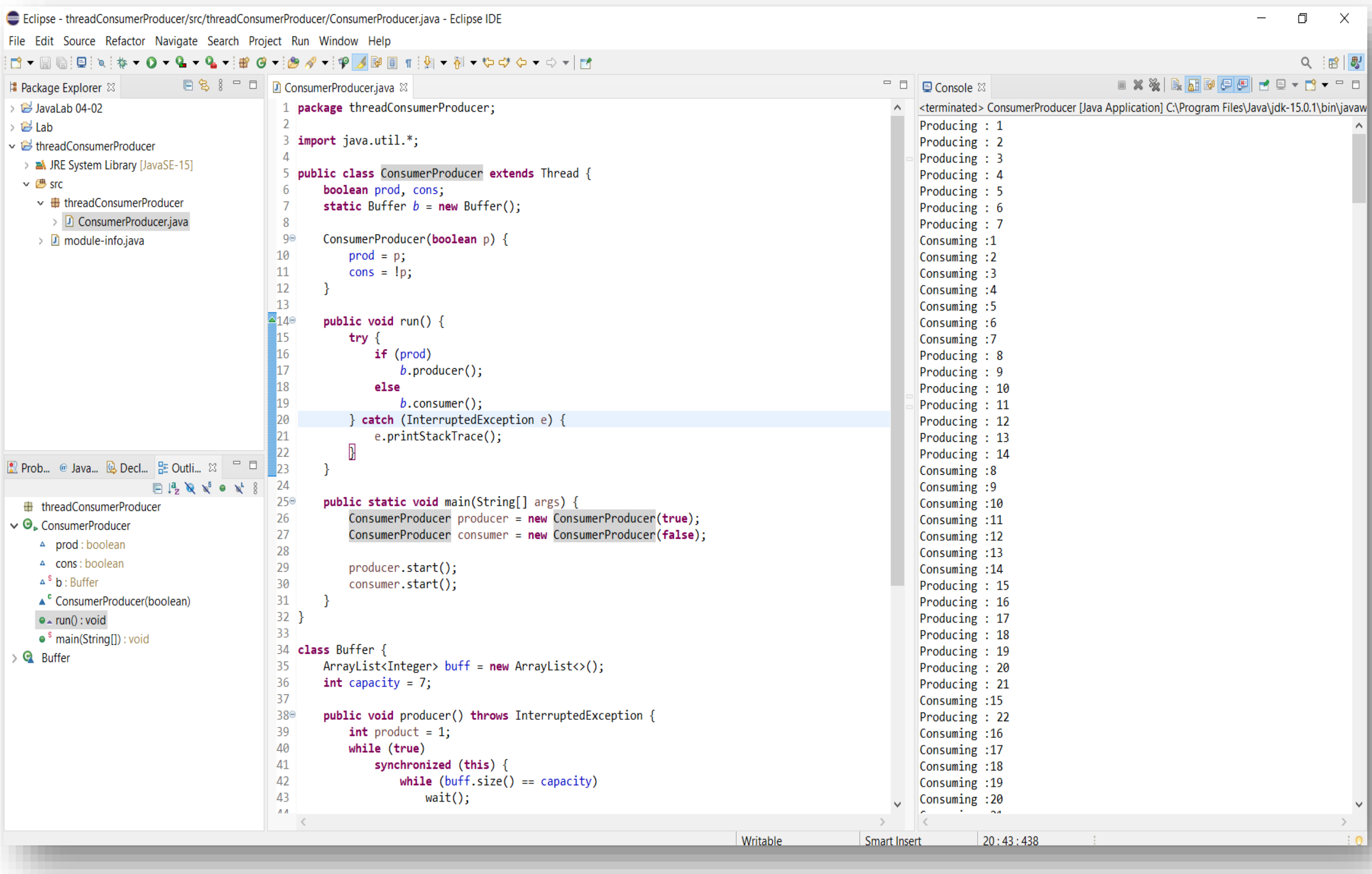
Solution

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


Eclipse - threadConsumerProducer/src/threadConsumerProducer/ConsumerProducer.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
JavaLab 04-02
Lab
threadConsumerProducer
JRE System Library [JavaSE-15]
src
threadConsumerProducer
ConsumerProducer.java
module-info.java
Prob... Java... Decl... Outli...
threadConsumerProducer
ConsumerProducer
prod : boolean
cons : boolean
b : Buffer
ConsumerProducer(boolean)
run() : void
main(String[]) : void
Buffer
ConsumerProducer.java
Console
<terminated> ConsumerProducer [Java Application] C:\Program Files\Java\jdk-15.0.1\bin\javaw
Writable
Smart Insert
20 : 43 : 438

```java
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();
```

```
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
Producing : 22
Consuming :16
Consuming :17
Consuming :18
Consuming :19
Consuming :20
```