Java, November 6, Lab Activity 5

Sharadindu Adhikari
19BCE2105

Write a java code to perform the followings using multithreading. Consider the scenario, suppose a producer is producing two different product in a market. There are n customers will consume either single or two products which is available in a market.

(i) Producer first will produce products and distribute to the market.

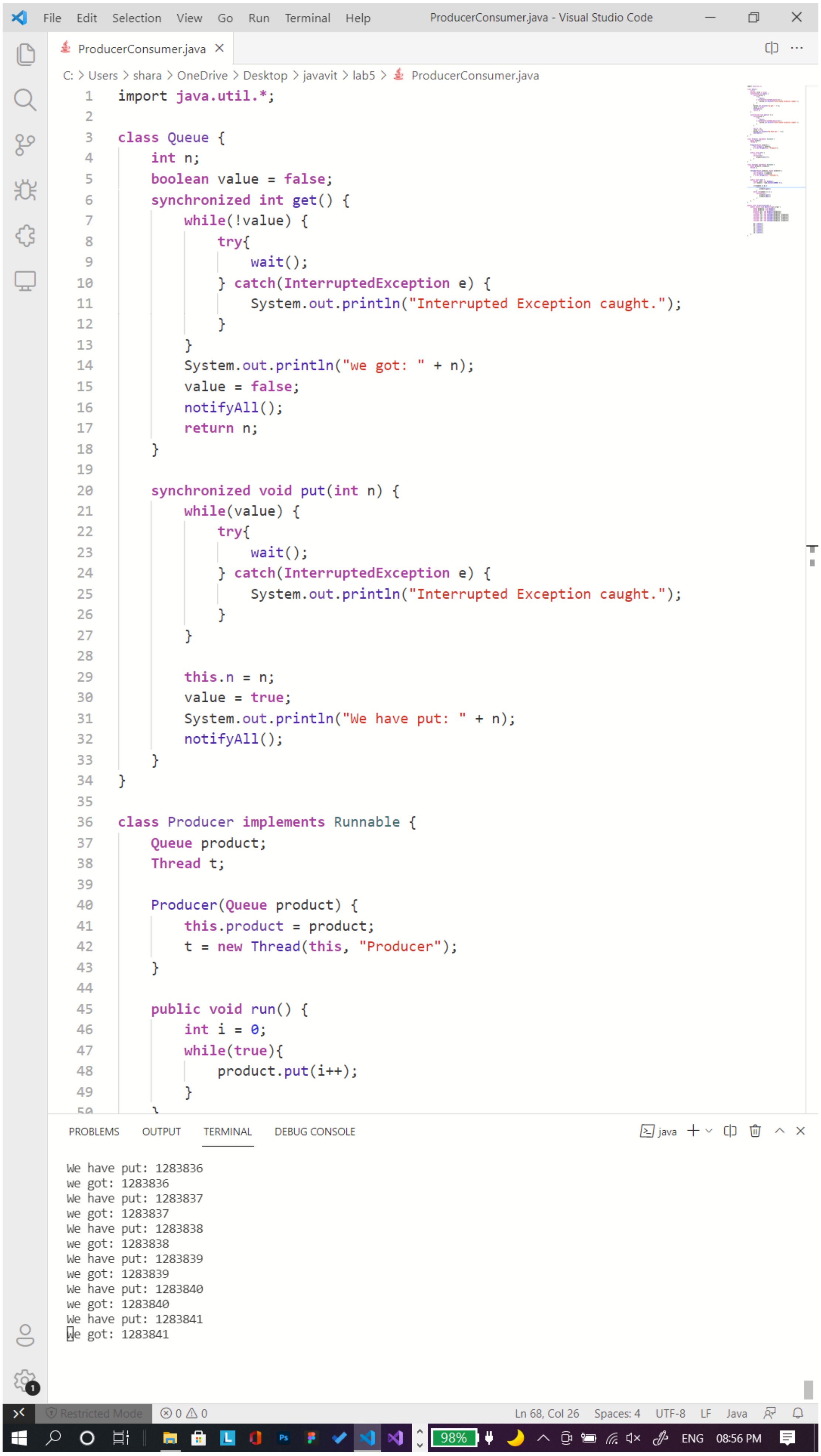
(ii) Whenever consumers are trying to buy one product, if it is available in the market he/she is allowed to buy the product otherwise consumer has to wait until producer produce the same product.

(iii) Wherever consumers are trying to buy two products at a same time, if both the products are available he/she is allowed to buy the products otherwise consumers has to wait until producer produce the products.

Solution:

```
import java.util.*;

class Queue {
    int n;
    boolean value = false;
    synchronized int get() {
        while(!value) {
            try{
                wait();
            } catch(InterruptedException e) {
                System.out.println("Interrupted Exception caught.");
            }
        }
        System.out.println("we got: " + n);
        value = false;
        notifyAll();
        return n;
    }

    synchronized void put(int n) {
        while(value) {
            try{
                wait();
            } catch(InterruptedException e) {
                System.out.println("Interrupted Exception caught.");
            }
        }

        this.n = n;
        value = true;
        System.out.println("We have put: " + n);
        notifyAll();
    }
}

class Producer implements Runnable {
    Queue product;
    Thread t;

    Producer(Queue product) {
        this.product = product;
        t = new Thread(this, "Producer");
    }

    public void run() {
        int i = 0;
        while(true){
            product.put(i++);
        }
    }
}

class Consumer implements Runnable {
    Queue product1, product2;
    Thread t;

    Consumer(Queue product1, Queue product2) {
        this.product1 = product1;
        this.product2 = product2;
        t = new Thread(this, "Consumer");
    }

    public void run() {
        Random rand = new Random();
        int randInt = rand.nextInt(10000) % 2;
        if(randInt == 0) {
            while(true) {
                product1.get();
            }
        }else if(randInt == 1) {
            while(true) {
                product1.get();
                product2.get();
            }
        }
    }
}

public class ProducerConsumer {
    public static void main(String[] args) {
        Queue product1 = new Queue();
        Queue product2 = new Queue();
        Producer p1 = new Producer(product1);
        Producer p2 = new Producer(product2);
        Consumer c1 = new Consumer(product1, product2);
        Consumer c2 = new Consumer(product1, product2);
        Consumer c3 = new Consumer(product1, product2);
        Consumer c4 = new Consumer(product1, product2);
        Consumer c5 = new Consumer(product1, product2);

        p1.t.start();
        p2.t.start();
        c1.t.start();
        c2.t.start();
        c3.t.start();
        c4.t.start();
        c5.t.start();
    }
}
```

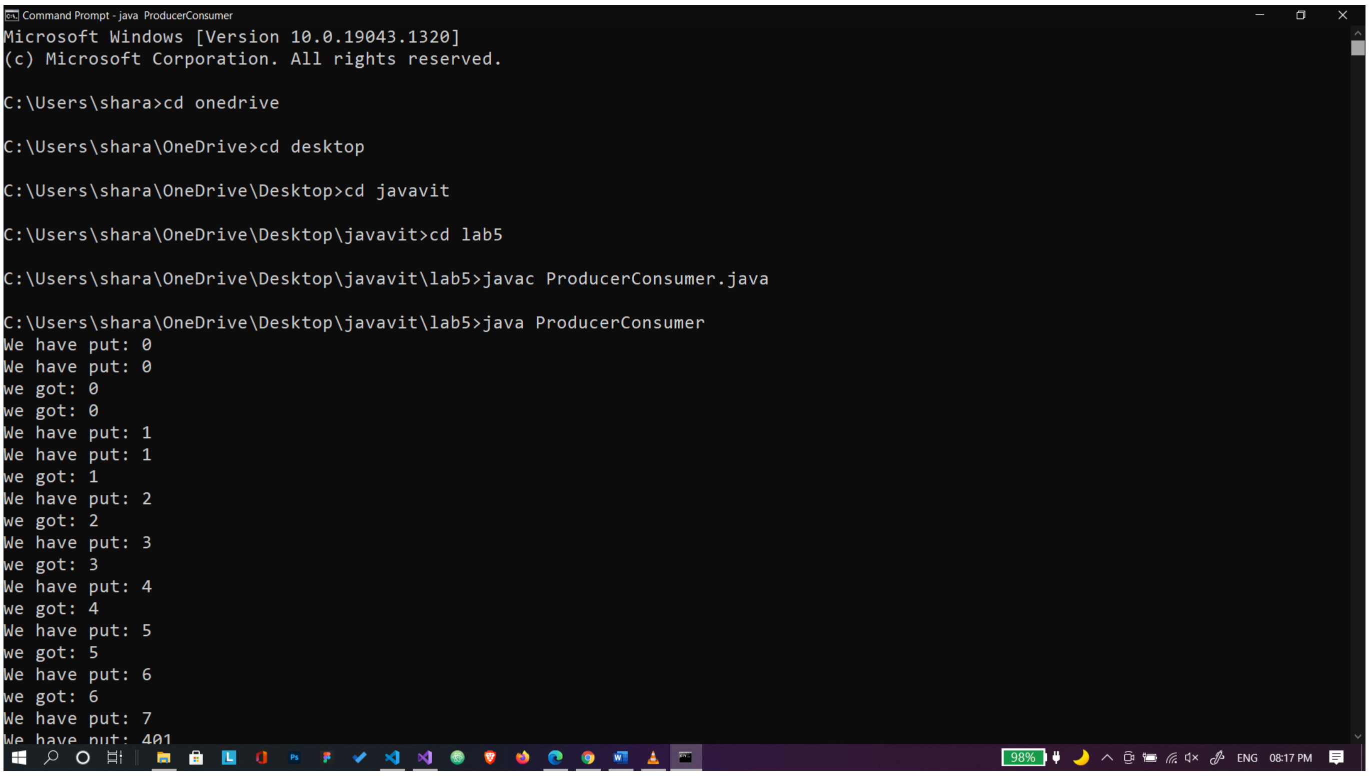
© Sharadindu Adhikari, 19BCE2105 sharadindu.adhikari2019@vitstudent.ac.in



File Edit Selection View Go Run Terminal Help
ProducerConsumer.java - Visual Studio Code
ProducerConsumer.java
C: > Users > shara > OneDrive > Desktop > javavit > lab5 > ProducerConsumer.java

```java
import java.util.*;

class Queue {
    int n;
    boolean value = false;
    synchronized int get() {
        while(!value) {
            try{
                wait();
            } catch(InterruptedException e) {
                System.out.println("Interrupted Exception caught.");
            }
        }
        System.out.println("we got: " + n);
        value = false;
        notifyAll();
        return n;
    }

    synchronized void put(int n) {
        while(value) {
            try{
                wait();
            } catch(InterruptedException e) {
                System.out.println("Interrupted Exception caught.");
            }
        }

        this.n = n;
        value = true;
        System.out.println("We have put: " + n);
        notifyAll();
    }
}

class Producer implements Runnable {
    Queue product;
    Thread t;

    Producer(Queue product) {
        this.product = product;
        t = new Thread(this, "Producer");
    }

    public void run() {
        int i = 0;
        while(true){
            product.put(i++);
        }
```

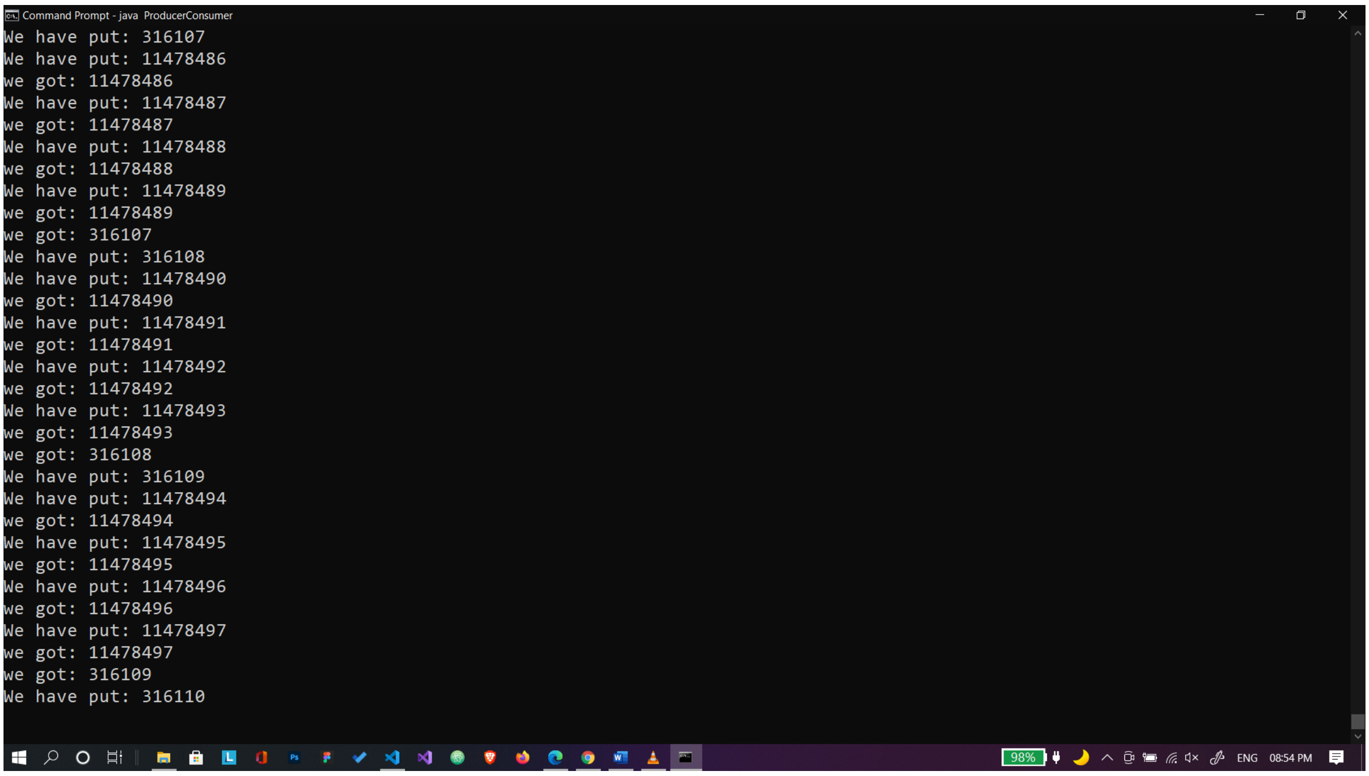
```
We have put: 1283836
we got: 1283836
We have put: 1283837
we got: 1283837
We have put: 1283838
we got: 1283838
We have put: 1283839
we got: 1283839
We have put: 1283840
we got: 1283840
We have put: 1283841
we got: 1283841
```



```
Microsoft Windows [Version 10.0.19043.1320]
(c) Microsoft Corporation. All rights reserved.

C:\Users\shara>cd onedrive

C:\Users\shara\OneDrive>cd desktop

C:\Users\shara\OneDrive\Desktop>cd javavit

C:\Users\shara\OneDrive\Desktop\javavit>cd lab5

C:\Users\shara\OneDrive\Desktop\javavit\lab5>javac ProducerConsumer.java

C:\Users\shara\OneDrive\Desktop\javavit\lab5>java ProducerConsumer
We have put: 0
We have put: 0
we got: 0
we got: 0
We have put: 1
We have put: 1
we got: 1
We have put: 2
we got: 2
We have put: 3
we got: 3
We have put: 4
we got: 4
We have put: 5
we got: 5
We have put: 6
we got: 6
We have put: 7
We have put: 401
```

•

•



```
We have put: 316107
We have put: 11478486
we got: 11478486
We have put: 11478487
we got: 11478487
We have put: 11478488
we got: 11478488
We have put: 11478489
we got: 11478489
we got: 316107
We have put: 316108
We have put: 11478490
we got: 11478490
We have put: 11478491
we got: 11478491
We have put: 11478492
we got: 11478492
We have put: 11478493
we got: 11478493
we got: 316108
We have put: 316109
We have put: 11478494
we got: 11478494
We have put: 11478495
we got: 11478495
We have put: 11478496
we got: 11478496
We have put: 11478497
we got: 11478497
we got: 316109
We have put: 316110
```

• •