

class Q5

int n;

boolean valueSet = false;

synchronized int get() {

while (!valueSet)

try {

System.out.println("In Consumer waiting\n");

wait();

}

catch (InterruptedException e) {

System.out.println("InterruptedException Caught");

}

System.out.println("Got:" + n);

valueSet = false;

System.out.println("In Enter the Producer\n");

notify();

return n;

}

synchronized void put(int n) {

while (valueSet)

try {

System.out.println("In producer waiting\n");

wait();

}

catch (InterruptedException e)

{ System.out.println("Interr Exception Caught"); }

```
this.n = n;
```

```
valueSet = true;
```

```
System.out.println("\n Entimate Consumer In");
```

```
notify();
```

```
}
```

```
}
```

```
class Producer implements Runnable {
```

```
Q q;
```

```
Producer(Q q) {
```

```
    this.q = q;
```

```
    new Thread(this, "Producer").start();
```

```
}
```

```
public void run() {
```

```
    int i = 0;
```

```
    while (i < 10) {
```

```
        q.put(i++);
```

```
    }
```

```
}
```

```
}
```

```
class Consumer implements Runnable {
```

```
Q q;
```

```
Consumer(Q q) {
```

```
    this.q = q;
```

```
    new Thread(this, "Consumer").start();
```

```
}
```

```
public void run() {
```

```
    int i = 0;
```



```

while (i < 5) {
    int r = q.get();
    System.out.println("Consumed: " + r);
    i++;
}
}
}

```

```

class PCFixed {
    public static void main(String args[]) {
        Q q = new Q();
        new Producer(q);
        new Consumer(q);
        System.out.println("Press Control-C to stop.");
    }
}

```

Output :-

Put : 0

Intimate Consumer

Producer waiting.

press Control-C to stop

Got : 0

~~Intimate Producer~~

Put : 1

Intimate Consumer

producer waiting.

Consumed : 0

Got : 1

Intimate Producer

consumed: 1

put: 2

Intimate Consumer

Producer waiting

Got: 2

Intimate Producer

consumed: 2

put: 3

Intimate Consumer

Producer waiting

Got: 3

Intimate Producer

Consumed: 3

put: 4

Intimate Consumer

Producer waiting

Got: 4

Intimate Producer

Consumed: 4

put: 5

Intimate Consumer

Producer waiting

Got: 5

Intimate Producer

consumed: 5

put: 6

Deadlock

Class A {

synchronized void foo (B b) {

String name = Thread.currentThread().getName();

System.out.println(name + " entered A.foo");

try {

Thread.sleep(1000);

} catch (Exception e) {

System.out.println("A Interrupted");

}

System.out.println(name + " trying to call B.last()");

b.last();

}

void last() {

System.out.println("Inside A.last()");

}

}

Class B {

synchronized void bar (A a) {

String name = Thread.currentThread().getName();

System.out.println(name + " entered B.bar");

try {

Thread.sleep(1000);

} catch (Exception e) {

System.out.println("B Interrupted");

}

```
System.out.println (name + " trying to call A.Last()");  
a.last();
```

⚡

```
void last() {
```

```
System.out.println ("Inside A.Last()");
```

}

}

```
class Deadlock implements Runnable
```

```
{
```

```
A a = new A();
```

```
B b = new B();
```

```
Deadlock() {
```

```
Thread.currentThread().setName("MainThread");
```

```
Thread t = new Thread(this, "RunningThread");
```

```
t.start();
```

```
a.foo(b); // get locks on a in this thread.
```

```
System.out.println("Back in main thread");
```

}

```
public void run() {
```

```
b.bar(a); // get locks on b in other thread.
```

```
System.out.println("Breaks in other thread");
```

```
}
```

```
public static void main (String args[]) {
```

```
new Deadlock();
```

⚡

Output :

MainThread entered A.foo Running Thread entered entered

B.bar Main Thread trying to call B.last()

Inside A.last

Back in main thread

RunningThread trying to call A.last()

Inside A.last

Back in ~~other~~ thread.

13/2/2024