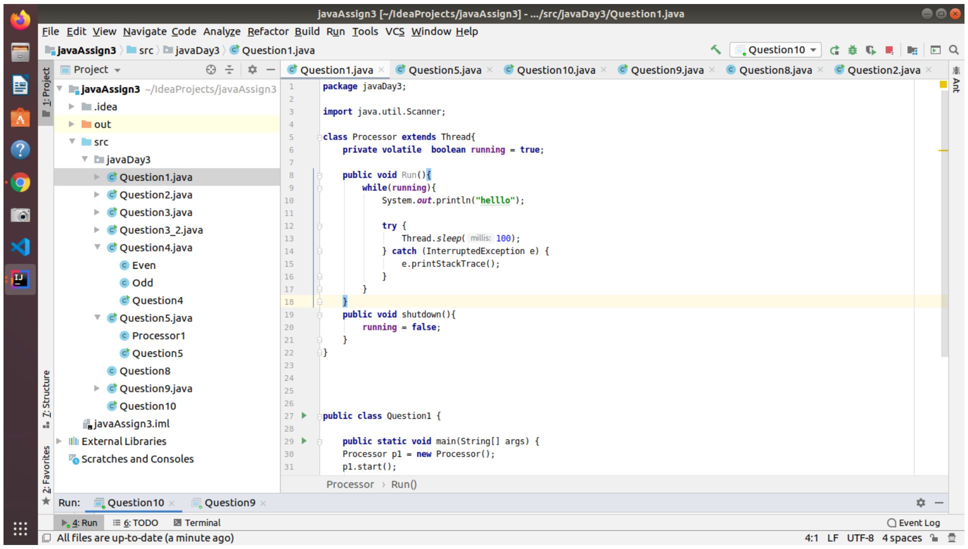
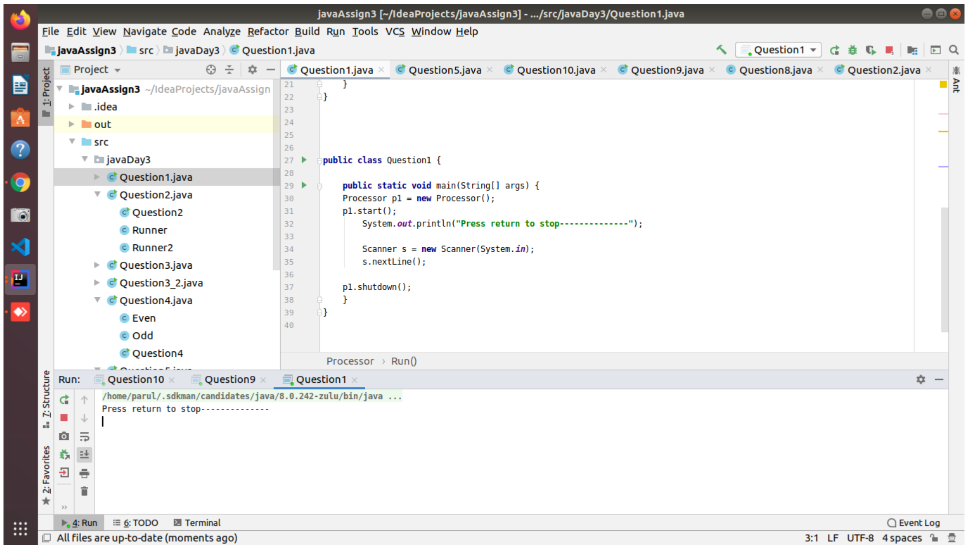
Question: 1:

Write a program to demonstrate the use of volatile keyword.


javaAssign3 [~/IdeaProjects/javaAssign3] - .../src/javaDay3/Question1.java
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
Question1.java
Question5.java
Question10.java
Question9.java
Question8.java
Question2.java
Run: Question10 Question9
All files are up-to-date (a minute ago)

```java
package javaDay3;

import java.util.Scanner;

class Processor extends Thread{
    private volatile  boolean running = true;

    public void Run(){
        while(running){
            System.out.println("helllo");

            try {
                Thread.sleep( millis: 100);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
    public void shutdown(){
        running = false;
    }
}



public class Question1 {

    public static void main(String[] args) {
    Processor p1 = new Processor();
    p1.start();
```

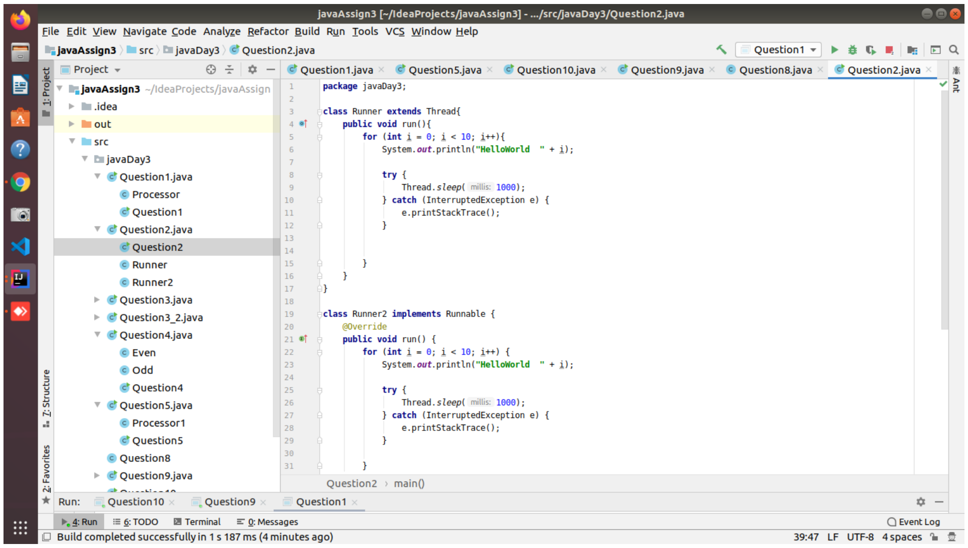

javaAssign3 [~/IdeaProjects/javaAssign3] - .../src/javaDay3/Question1.java
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
Question1.java
Question5.java
Question10.java
Question9.java
Question8.java
Question2.java
Run: Question10 Question9 Question1
All files are up-to-date (moments ago)

```java
    }
}



public class Question1 {

    public static void main(String[] args) {
    Processor p1 = new Processor();
    p1.start();
        System.out.println("Press return to stop-------------");

        Scanner s = new Scanner(System.in);
        s.nextLine();

    p1.shutdown();
    }
}
```

```
/home/parul/.sdkman/candidates/java/8.0.242-zulu/bin/java ...
Press return to stop-------------
```

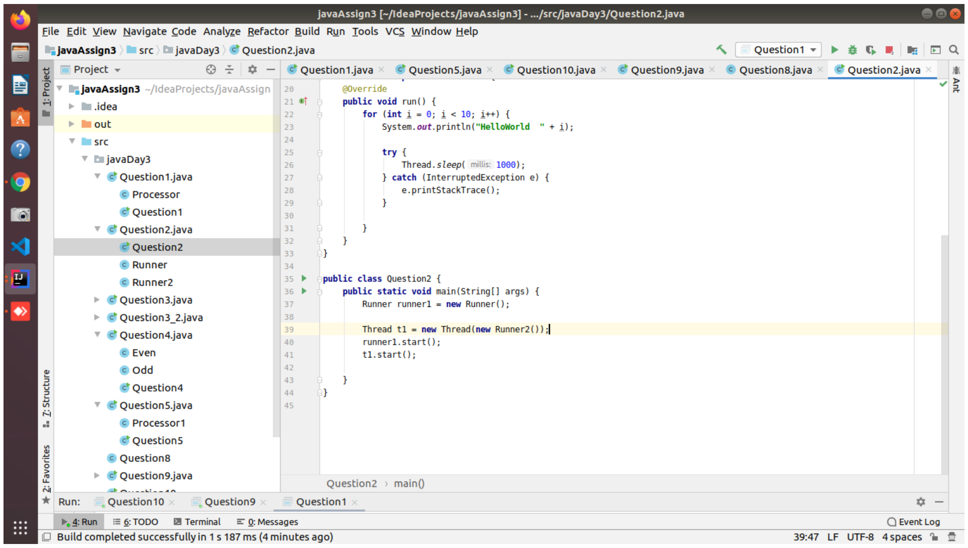
Question: 2:

Write a program to create a thread using Thread class and Runnable interface each.



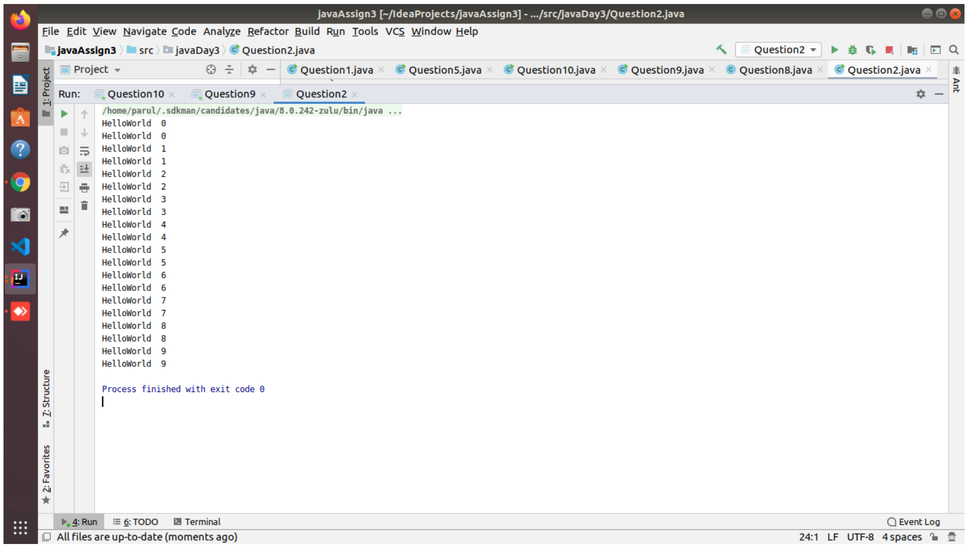
```java
package javaDay3;

class Runner extends Thread{
    public void run(){
        for (int i = 0; i < 10; i++){
            System.out.println("HelloWorld  " + i);

            try {
                Thread.sleep( millis: 1000);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }


        }
    }
}

class Runner2 implements Runnable {
    @Override
    public void run() {
        for (int i = 0; i < 10; i++) {
            System.out.println("HelloWorld  " + i);

            try {
                Thread.sleep( millis: 1000);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }

        }
```

Build completed successfully in 1 s 187 ms (4 minutes ago)



```java
    @Override
    public void run() {
        for (int i = 0; i < 10; i++) {
            System.out.println("HelloWorld  " + i);

            try {
                Thread.sleep( millis: 1000);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }

        }
    }
}

public class Question2 {
    public static void main(String[] args) {
        Runner runner1 = new Runner();

        Thread t1 = new Thread(new Runner2());
        runner1.start();
        t1.start();

    }
}
```

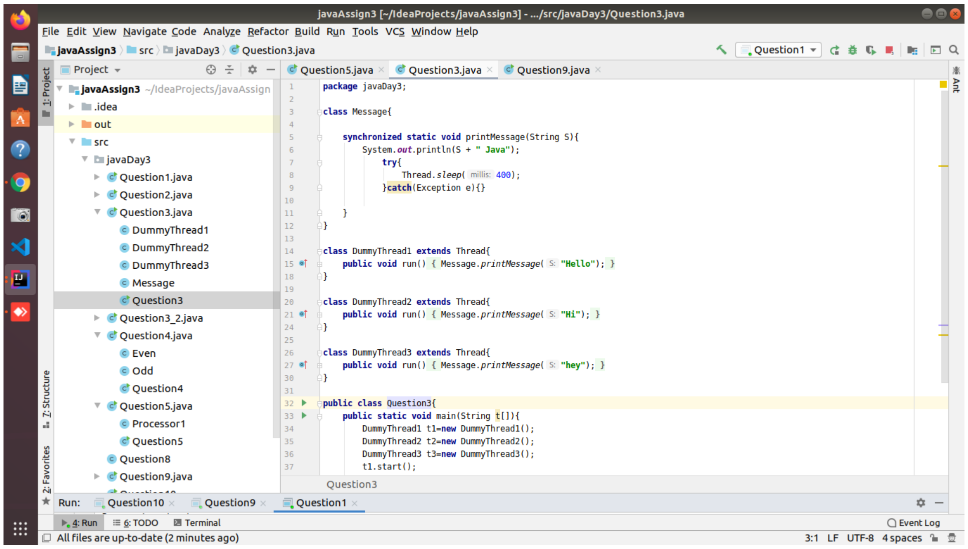
Build completed successfully in 1 s 187 ms (4 minutes ago)


javaAssign3 [~/IdeaProjects/javaAssign3] - .../src/javaDay3/Question2.java
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
javaAssign3 › src › javaDay3 › Question2.java
Question2
Project
Question1.java
Question5.java
Question10.java
Question9.java
Question8.java
Question2.java
Run:
Question10
Question9
Question2
4: Run
6: TODO
Terminal
Event Log
24:1 LF UTF-8 4 spaces

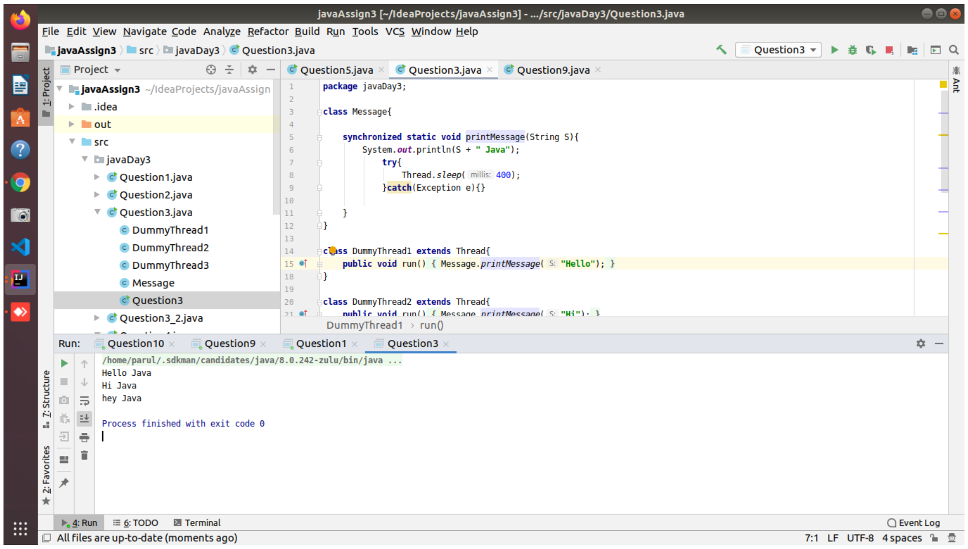
```
/home/parul/.sdkman/candidates/java/8.0.242-zulu/bin/java ...
HelloWorld  0
HelloWorld  0
HelloWorld  1
HelloWorld  1
HelloWorld  2
HelloWorld  2
HelloWorld  3
HelloWorld  3
HelloWorld  4
HelloWorld  4
HelloWorld  5
HelloWorld  5
HelloWorld  6
HelloWorld  6
HelloWorld  7
HelloWorld  7
HelloWorld  8
HelloWorld  8
HelloWorld  9
HelloWorld  9

Process finished with exit code 0
```

All files are up-to-date (moments ago)

Question: 3:
Write a program using synchronization block and synchronization method.

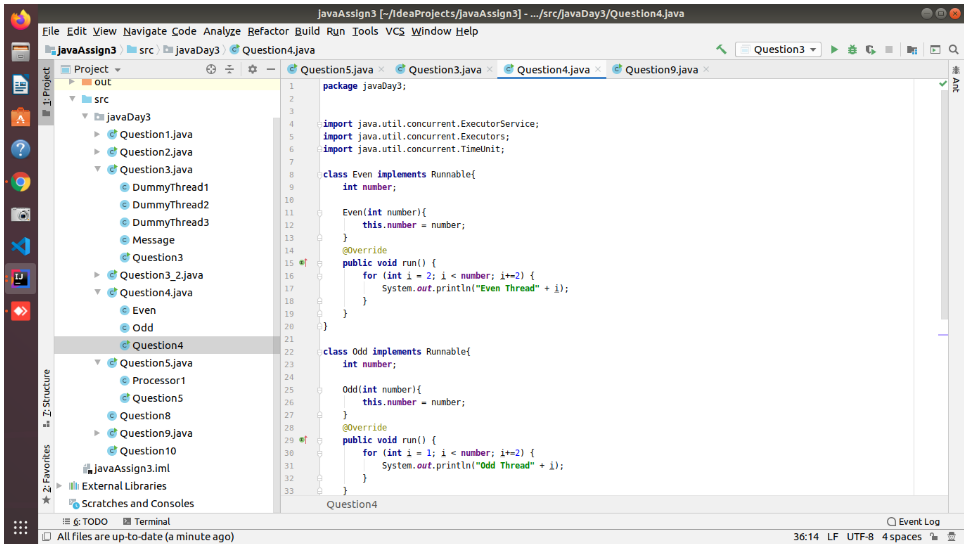


```java
package javaDay3;

class Message{

    synchronized static void printMessage(String S){
        System.out.println(S + " Java");
            try{
                Thread.sleep( millis: 400);
            }catch(Exception e){}

    }
}

class DummyThread1 extends Thread{
    public void run() { Message.printMessage( S: "Hello"); }
}

class DummyThread2 extends Thread{
    public void run() { Message.printMessage( S: "Hi"); }
}

class DummyThread3 extends Thread{
    public void run() { Message.printMessage( S: "hey"); }
}

public class Question3{
    public static void main(String t[]){
        DummyThread1 t1=new DummyThread1();
        DummyThread2 t2=new DummyThread2();
        DummyThread3 t3=new DummyThread3();
        t1.start();
```

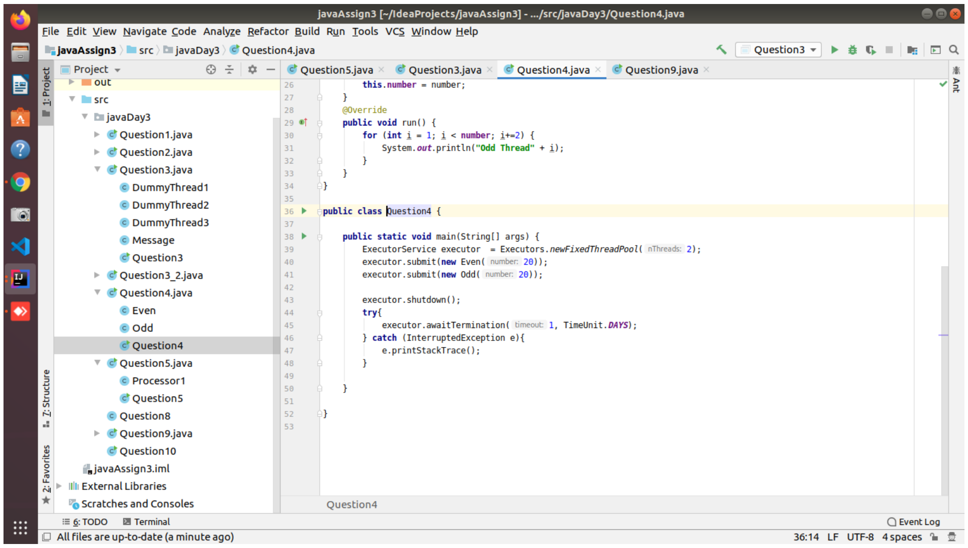


```
/home/parul/.sdkman/candidates/java/8.0.242-zulu/bin/java ...
Hello Java
Hi Java
hey Java

Process finished with exit code 0
```

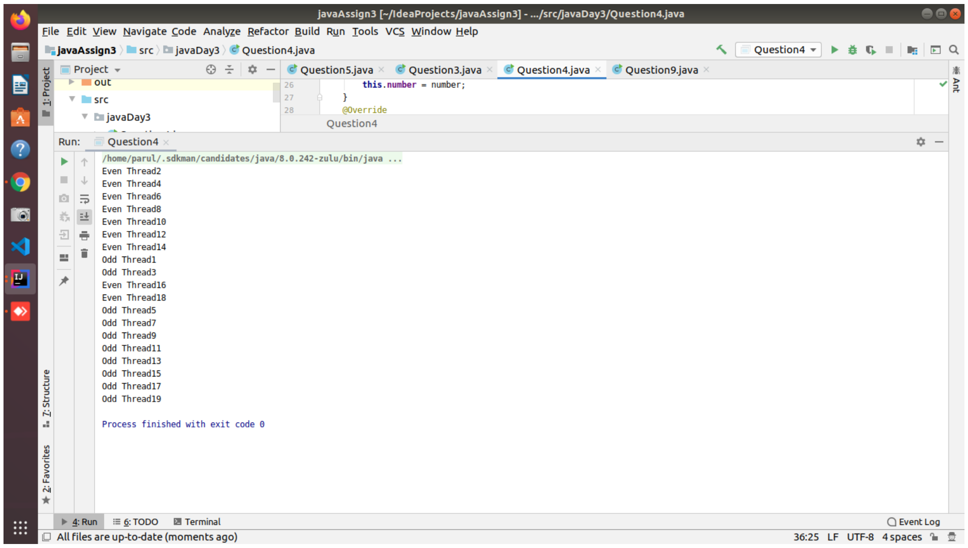
Question: 4:
Write a program to create a Thread pool of 2 threads where one Thread will print even numbers and other will print odd numbers.



```
package javaDay3;


import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;
import java.util.concurrent.TimeUnit;

class Even implements Runnable{
    int number;

    Even(int number){
        this.number = number;
    }
    @Override
    public void run() {
        for (int i = 2; i < number; i+=2) {
            System.out.println("Even Thread" + i);
        }
    }
}

class Odd implements Runnable{
    int number;

    Odd(int number){
        this.number = number;
    }
    @Override
    public void run() {
        for (int i = 1; i < number; i+=2) {
            System.out.println("Odd Thread" + i);
        }
    }
```

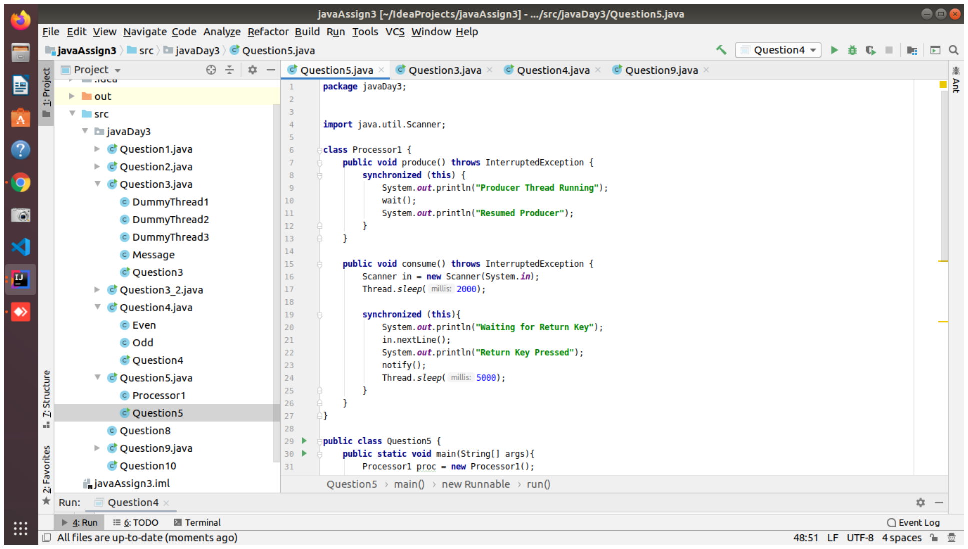

javaAssign3 [~/IdeaProjects/javaAssign3] - .../src/javaDay3/Question4.java
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
javaAssign3 〉 src 〉 javaDay3 〉 Question4.java
Question3
Question4
6: TODO
Terminal
Event Log
All files are up-to-date (a minute ago)
36:14 LF UTF-8 4 spaces

```
        this.number = number;
    }
    @Override
    public void run() {
        for (int i = 1; i < number; i+=2) {
            System.out.println("Odd Thread" + i);
        }
    }
}

public class Question4 {

    public static void main(String[] args) {
        ExecutorService executor  = Executors.newFixedThreadPool( nThreads: 2);
        executor.submit(new Even( number: 20));
        executor.submit(new Odd( number: 20));

        executor.shutdown();
        try{
            executor.awaitTermination( timeout: 1, TimeUnit.DAYS);
        } catch (InterruptedException e){
            e.printStackTrace();
        }

    }

}
```


javaAssign3 [~/IdeaProjects/javaAssign3] - .../src/javaDay3/Question4.java
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
javaAssign3 〉 src 〉 javaDay3 〉 Question4.java
Question4
Run: Question4
4: Run
6: TODO
Terminal
Event Log
All files are up-to-date (moments ago)
36:25 LF UTF-8 4 spaces

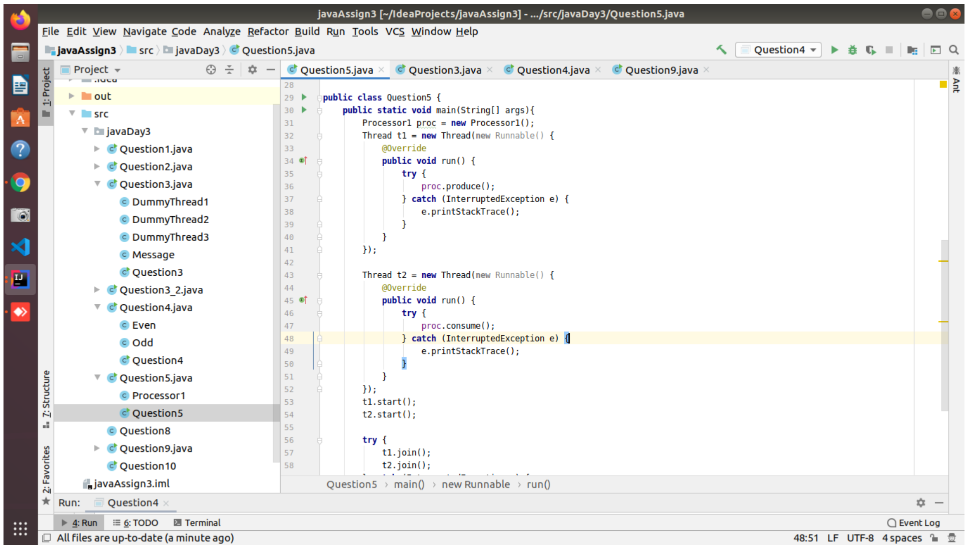
```
/home/parul/.sdkman/candidates/java/8.0.242-zulu/bin/java ...
Even Thread2
Even Thread4
Even Thread6
Even Thread8
Even Thread10
Even Thread12
Even Thread14
Odd Thread1
Odd Thread3
Even Thread16
Even Thread18
Odd Thread5
Odd Thread7
Odd Thread9
Odd Thread11
Odd Thread13
Odd Thread15
Odd Thread17
Odd Thread19

Process finished with exit code 0
```

Question: 5:
Write a program to demonstrate wait and notify methods.



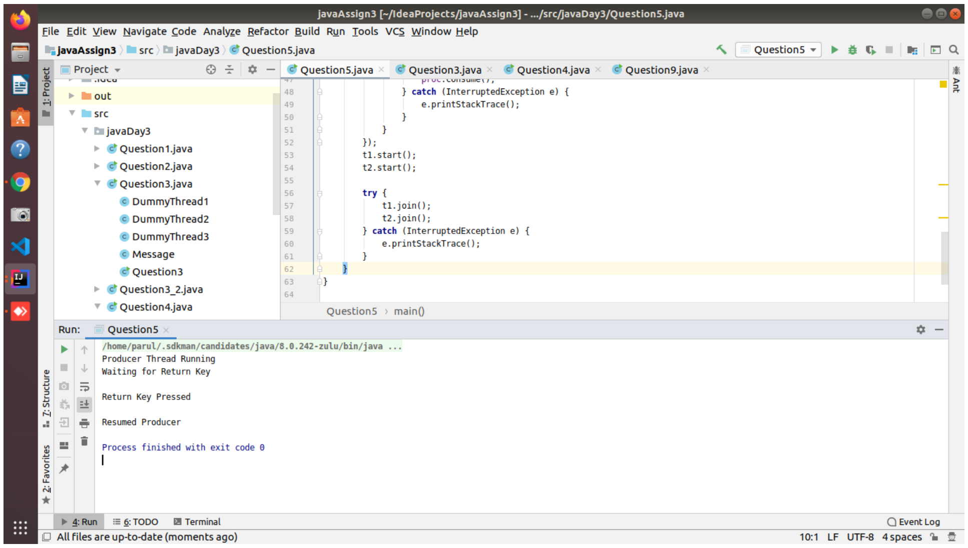
javaAssign3 [~/IdeaProjects/javaAssign3] - .../src/javaDay3/Question5.java
Question5.java
Question3.java
Question4.java
Question9.java
Question5 › main() › new Runnable › run()
All files are up-to-date (moments ago)

```java
package javaDay3;


import java.util.Scanner;

class Processor1 {
    public void produce() throws InterruptedException {
        synchronized (this) {
            System.out.println("Producer Thread Running");
            wait();
            System.out.println("Resumed Producer");
        }
    }

    public void consume() throws InterruptedException {
        Scanner in = new Scanner(System.in);
        Thread.sleep(2000);

        synchronized (this){
            System.out.println("Waiting for Return Key");
            in.nextLine();
            System.out.println("Return Key Pressed");
            notify();
            Thread.sleep(5000);
        }
    }
}

public class Question5 {
    public static void main(String[] args){
        Processor1 proc = new Processor1();
```



javaAssign3 [~/IdeaProjects/javaAssign3] - .../src/javaDay3/Question5.java
Question5.java
Question3.java
Question4.java
Question9.java
Question5 › main() › new Runnable › run()
All files are up-to-date (a minute ago)

```java

public class Question5 {
    public static void main(String[] args){
        Processor1 proc = new Processor1();
        Thread t1 = new Thread(new Runnable() {
            @Override
            public void run() {
                try {
                    proc.produce();
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
            }
        });

        Thread t2 = new Thread(new Runnable() {
            @Override
            public void run() {
                try {
                    proc.consume();
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
            }
        });
        t1.start();
        t2.start();

        try {
            t1.join();
            t2.join();
```

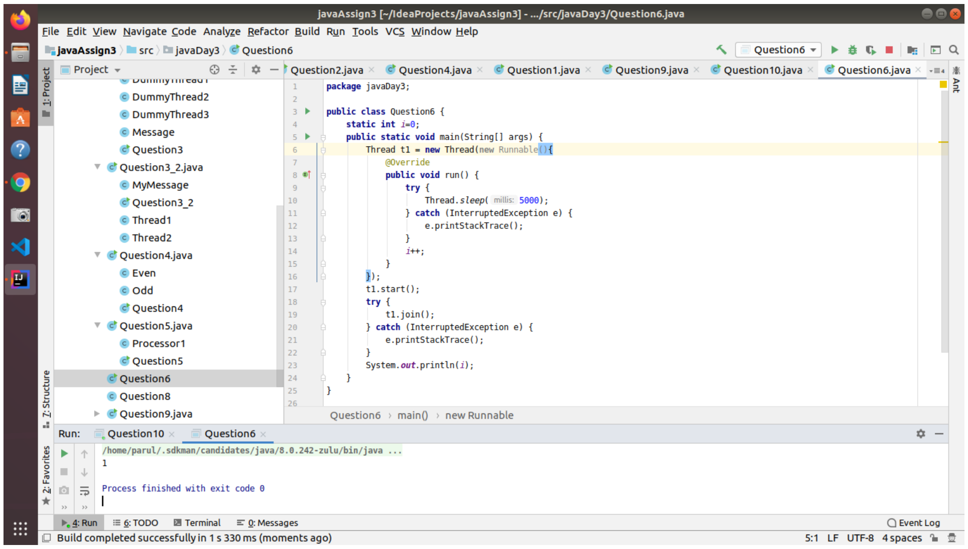


javaAssign3 [~/IdeaProjects/javaAssign3] - .../src/javaDay3/Question5.java

```java
                    proc.consume();
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
            }
        });
        t1.start();
        t2.start();

        try {
            t1.join();
            t2.join();
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
}
```

```
/home/parul/.sdkman/candidates/java/8.0.242-zulu/bin/java ...
Producer Thread Running
Waiting for Return Key

Return Key Pressed

Resumed Producer

Process finished with exit code 0
```

Question: 6:
Write a program to demonstrate sleep and join methods.



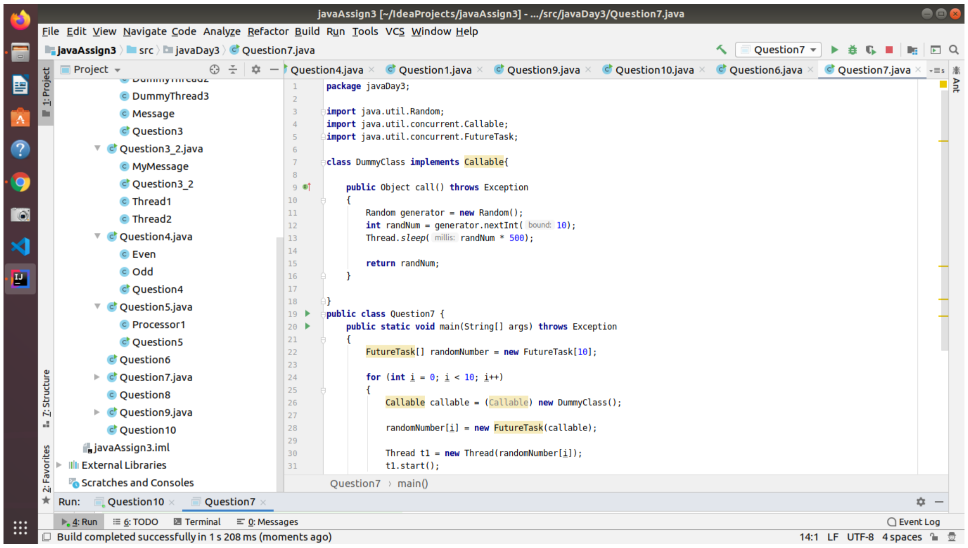
javaAssign3 [~/IdeaProjects/javaAssign3] - .../src/javaDay3/Question6.java

```java
package javaDay3;

public class Question6 {
    static int i=0;
    public static void main(String[] args) {
        Thread t1 = new Thread(new Runnable(){
            @Override
            public void run() {
                try {
                    Thread.sleep( millis: 5000);
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
                i++;
            }
        });
        t1.start();
        try {
            t1.join();
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
        System.out.println(i);
    }
}
```

```
/home/parul/.sdkman/candidates/java/8.0.242-zulu/bin/java ...
1

Process finished with exit code 0
```

Build completed successfully in 1 s 330 ms (moments ago)

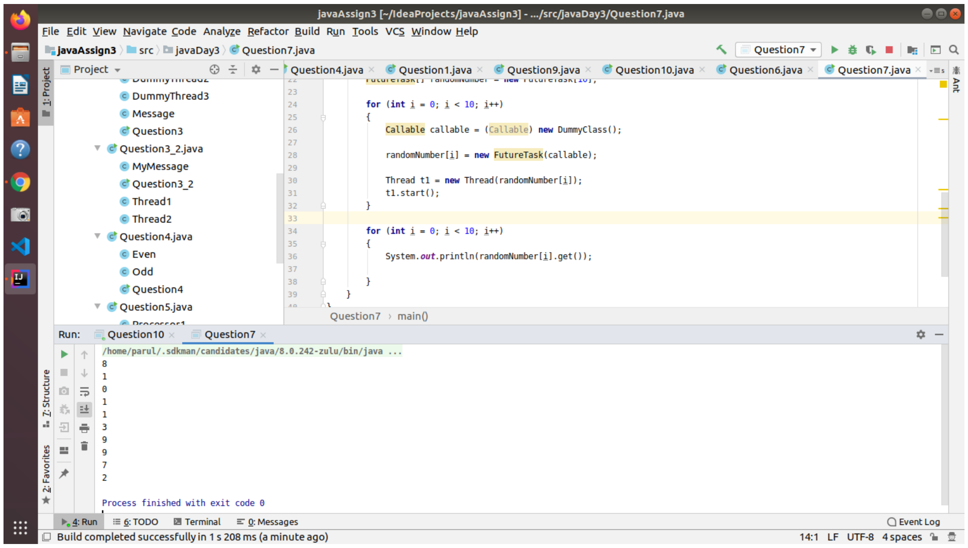
Question: 7:
Run a task with the help of callable and store it's result in the Future.



javaAssign3 [~/IdeaProjects/javaAssign3] - .../src/javaDay3/Question7.java

```java
package javaDay3;

import java.util.Random;
import java.util.concurrent.Callable;
import java.util.concurrent.FutureTask;

class DummyClass implements Callable{

    public Object call() throws Exception
    {
        Random generator = new Random();
        int randNum = generator.nextInt( 10);
        Thread.sleep( randNum * 500);

        return randNum;
    }

}
public class Question7 {
    public static void main(String[] args) throws Exception
    {
        FutureTask[] randomNumber = new FutureTask[10];

        for (int i = 0; i < 10; i++)
        {
            Callable callable = (Callable) new DummyClass();

            randomNumber[i] = new FutureTask(callable);

            Thread t1 = new Thread(randomNumber[i]);
            t1.start();
```

Build completed successfully in 1 s 208 ms (moments ago)



javaAssign3 [~/IdeaProjects/javaAssign3] - .../src/javaDay3/Question7.java

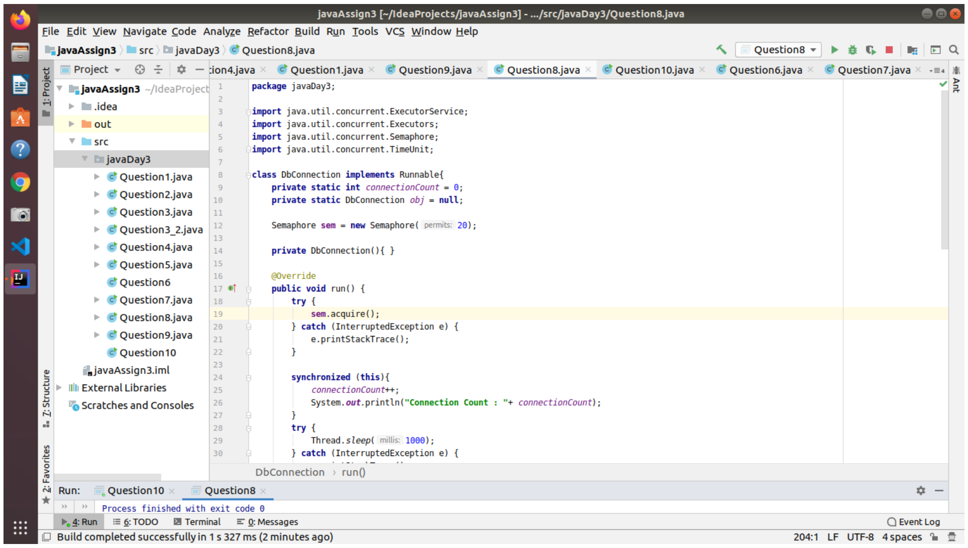
```java
        for (int i = 0; i < 10; i++)
        {
            Callable callable = (Callable) new DummyClass();

            randomNumber[i] = new FutureTask(callable);

            Thread t1 = new Thread(randomNumber[i]);
            t1.start();
        }

        for (int i = 0; i < 10; i++)
        {
            System.out.println(randomNumber[i].get());

        }
    }
```

```
/home/parul/.sdkman/candidates/java/8.0.242-zulu/bin/java ...
8
1
0
1
1
3
9
9
7
2

Process finished with exit code 0
```

Build completed successfully in 1 s 208 ms (a minute ago)

Question: 8:

Write a program to demonstrate the use of semaphore.

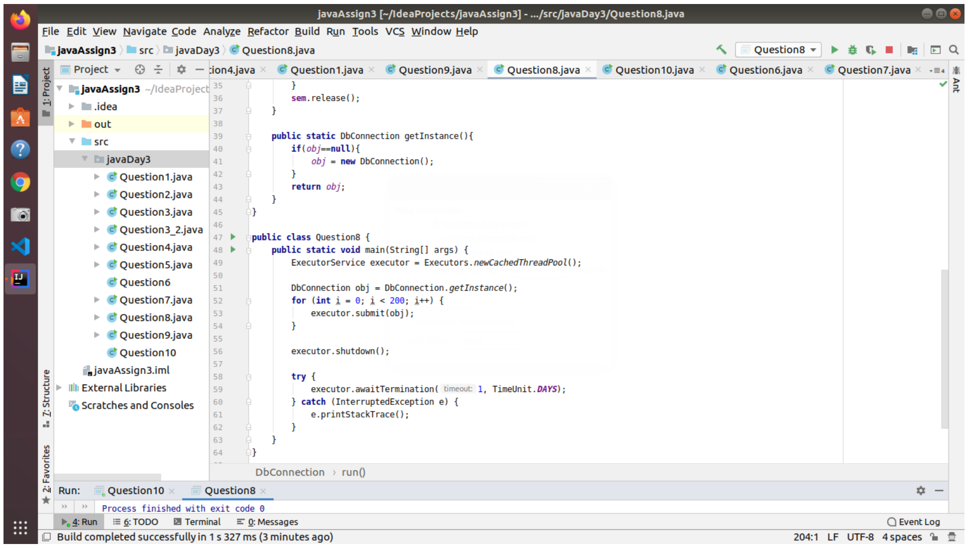


javaAssign3 [~/IdeaProjects/javaAssign3] - .../src/javaDay3/Question8.java

```java
package javaDay3;

import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;
import java.util.concurrent.Semaphore;
import java.util.concurrent.TimeUnit;

class DbConnection implements Runnable{
    private static int connectionCount = 0;
    private static DbConnection obj = null;

    Semaphore sem = new Semaphore( 20);

    private DbConnection(){ }

    @Override
    public void run() {
        try {
            sem.acquire();
        } catch (InterruptedException e) {
            e.printStackTrace();
        }

        synchronized (this){
            connectionCount++;
            System.out.println("Connection Count : "+ connectionCount);
        }
        try {
            Thread.sleep( 1000);
        } catch (InterruptedException e) {
```

```
Process finished with exit code 0
```

Build completed successfully in 1 s 327 ms (2 minutes ago)

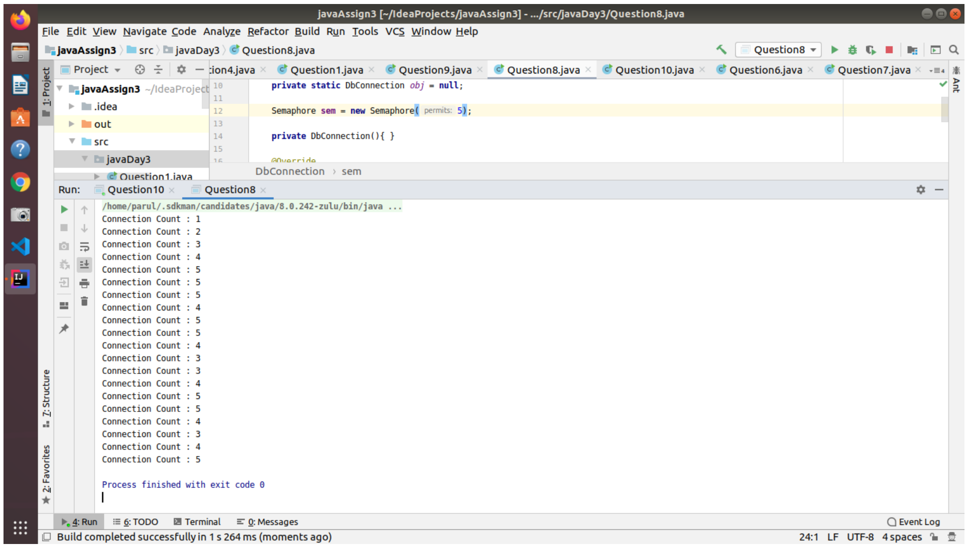


javaAssign3 [~/IdeaProjects/javaAssign3] - .../src/javaDay3/Question8.java

```java
        }
        sem.release();
    }

    public static DbConnection getInstance(){
        if(obj==null){
            obj = new DbConnection();
        }
        return obj;
    }
}

public class Question8 {
    public static void main(String[] args) {
        ExecutorService executor = Executors.newCachedThreadPool();

        DbConnection obj = DbConnection.getInstance();
        for (int i = 0; i < 200; i++) {
            executor.submit(obj);
        }

        executor.shutdown();

        try {
            executor.awaitTermination( 1, TimeUnit.DAYS);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
}
```

```
Process finished with exit code 0
```

Build completed successfully in 1 s 327 ms (3 minutes ago)



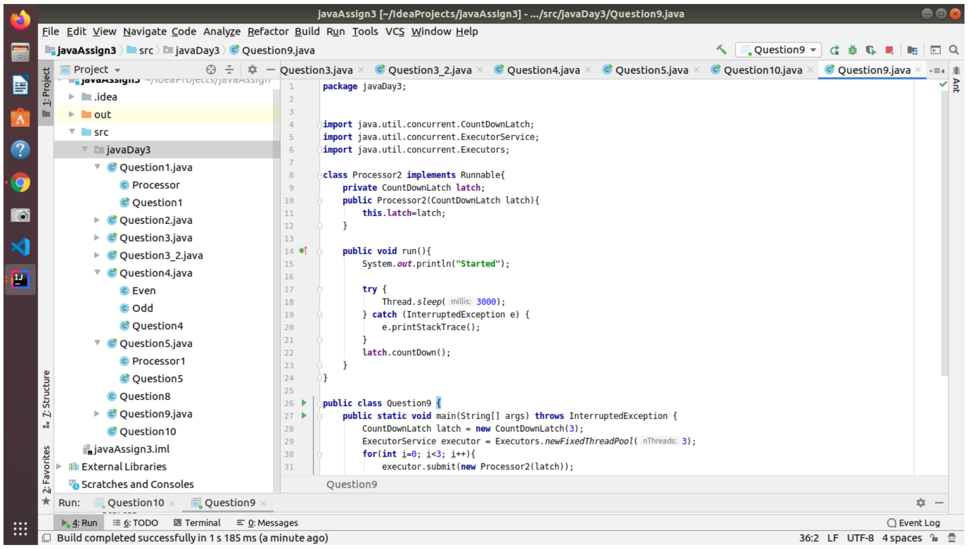
javaAssign3 [~/IdeaProjects/javaAssign3] - .../src/javaDay3/Question8.java

```java
    private static DbConnection obj = null;

    Semaphore sem = new Semaphore( 5);

    private DbConnection(){ }
```

```
/home/parul/.sdkman/candidates/java/8.0.242-zulu/bin/java ...
Connection Count : 1
Connection Count : 2
Connection Count : 3
Connection Count : 4
Connection Count : 5
Connection Count : 5
Connection Count : 5
Connection Count : 4
Connection Count : 5
Connection Count : 5
Connection Count : 4
Connection Count : 3
Connection Count : 3
Connection Count : 4
Connection Count : 5
Connection Count : 5
Connection Count : 4
Connection Count : 3
Connection Count : 4
Connection Count : 5

Process finished with exit code 0
```

Build completed successfully in 1 s 264 ms (moments ago)

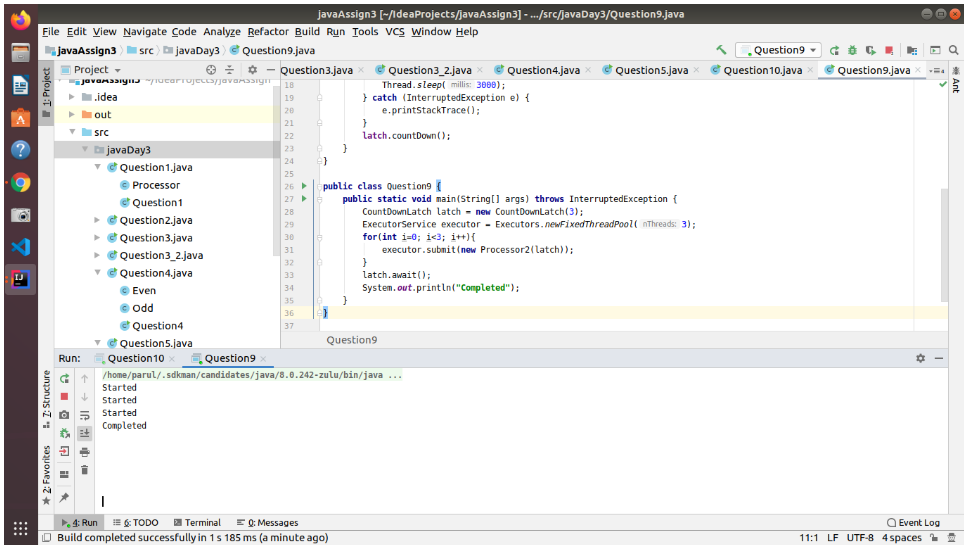
Question: 9:
Write a program to demonstrate the use of CountDownLatch.



javaAssign3 [~/IdeaProjects/javaAssign3] - .../src/javaDay3/Question9.java

```java
package javaDay3;


import java.util.concurrent.CountDownLatch;
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;

class Processor2 implements Runnable{
    private CountDownLatch latch;
    public Processor2(CountDownLatch latch){
        this.latch=latch;
    }

    public void run(){
        System.out.println("Started");

        try {
            Thread.sleep( millis: 3000);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
        latch.countDown();
    }
}

public class Question9 {
    public static void main(String[] args) throws InterruptedException {
        CountDownLatch latch = new CountDownLatch(3);
        ExecutorService executor = Executors.newFixedThreadPool( nThreads: 3);
        for(int i=0; i<3; i++){
            executor.submit(new Processor2(latch));
```

Build completed successfully in 1 s 185 ms (a minute ago)



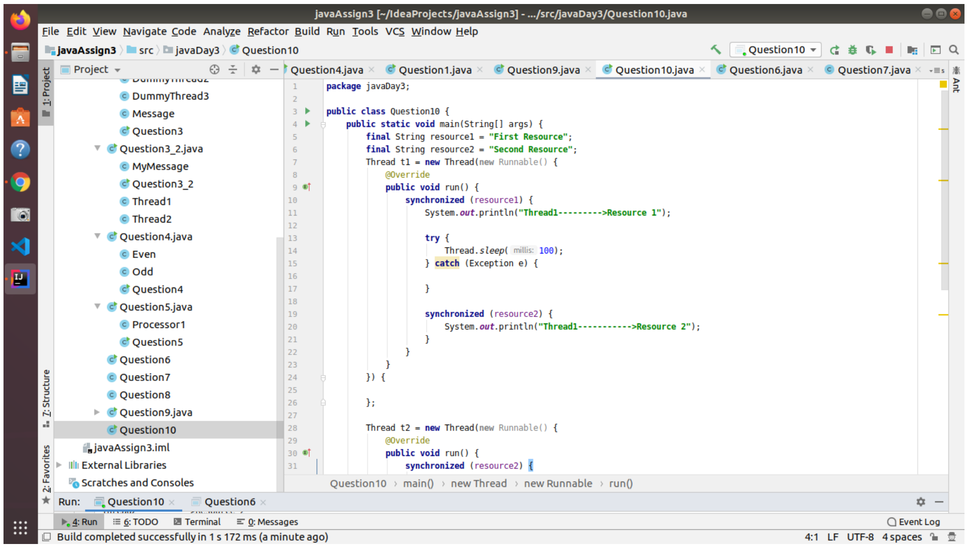
javaAssign3 [~/IdeaProjects/javaAssign3] - .../src/javaDay3/Question9.java

```java
            Thread.sleep( millis: 3000);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
        latch.countDown();
    }
}

public class Question9 {
    public static void main(String[] args) throws InterruptedException {
        CountDownLatch latch = new CountDownLatch(3);
        ExecutorService executor = Executors.newFixedThreadPool( nThreads: 3);
        for(int i=0; i<3; i++){
            executor.submit(new Processor2(latch));
        }
        latch.await();
        System.out.println("Completed");
    }
}
```

```
/home/parul/.sdkman/candidates/java/8.0.242-zulu/bin/java ...
Started
Started
Started
Completed
```

Build completed successfully in 1 s 185 ms (a minute ago)

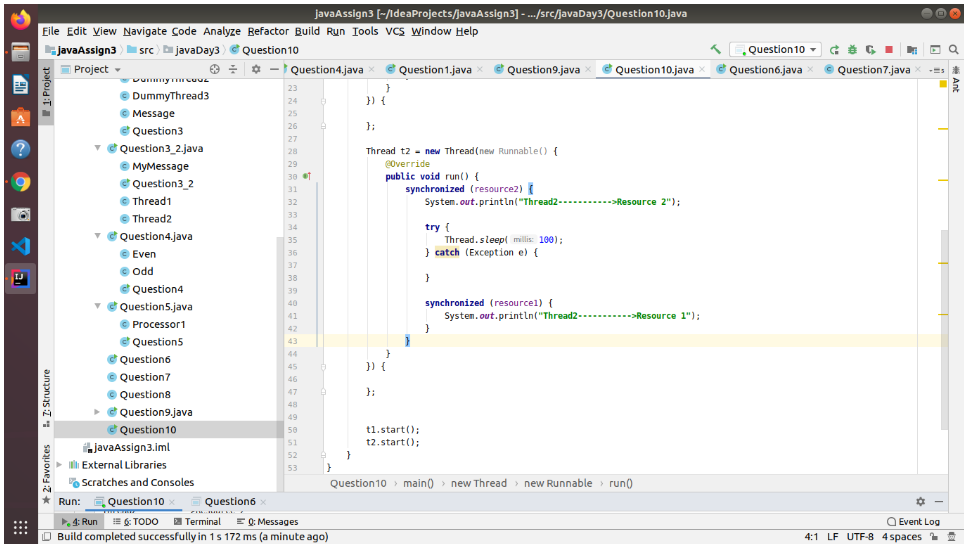
Question: 10:
Write a program which creates deadlock between 2 threads.



javaAssign3 [~/IdeaProjects/javaAssign3] - .../src/javaDay3/Question10.java

```java
package javaDay3;

public class Question10 {
    public static void main(String[] args) {
        final String resource1 = "First Resource";
        final String resource2 = "Second Resource";
        Thread t1 = new Thread(new Runnable() {
            @Override
            public void run() {
                synchronized (resource1) {
                    System.out.println("Thread1--------->Resource 1");

                    try {
                        Thread.sleep( millis: 100);
                    } catch (Exception e) {

                    }

                    synchronized (resource2) {
                        System.out.println("Thread1----------->Resource 2");
                    }
                }
            }
        }) {

        };

        Thread t2 = new Thread(new Runnable() {
            @Override
            public void run() {
                synchronized (resource2) {
```

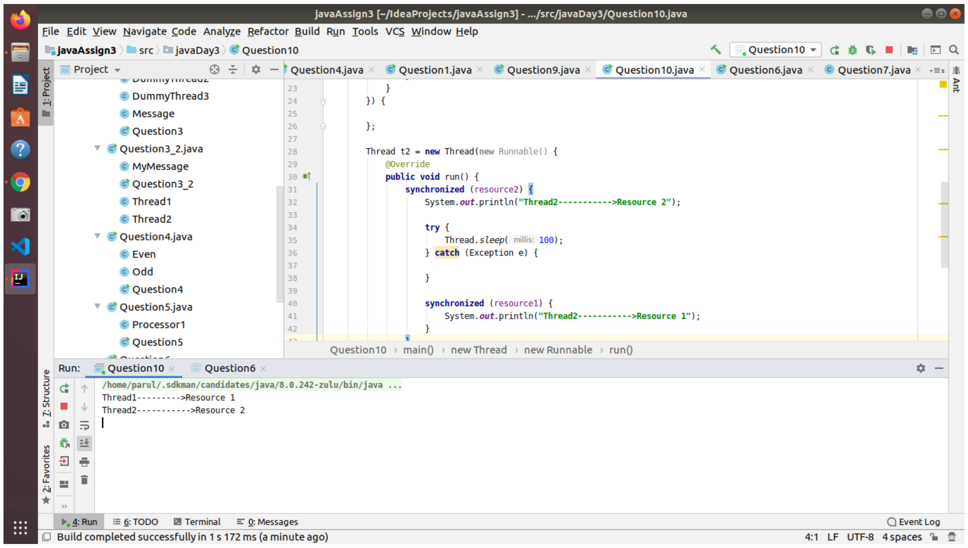
Build completed successfully in 1 s 172 ms (a minute ago)



javaAssign3 [~/IdeaProjects/javaAssign3] - .../src/javaDay3/Question10.java

```java
            }
        }) {

        };

        Thread t2 = new Thread(new Runnable() {
            @Override
            public void run() {
                synchronized (resource2) {
                    System.out.println("Thread2----------->Resource 2");

                    try {
                        Thread.sleep( millis: 100);
                    } catch (Exception e) {

                    }

                    synchronized (resource1) {
                        System.out.println("Thread2----------->Resource 1");
                    }
                }
            }
        }) {

        };


        t1.start();
        t2.start();
    }
}
```

Build completed successfully in 1 s 172 ms (a minute ago)



javaAssign3 [~/IdeaProjects/javaAssign3] - .../src/javaDay3/Question10.java

```java
            }
        }) {

        };

        Thread t2 = new Thread(new Runnable() {
            @Override
            public void run() {
                synchronized (resource2) {
                    System.out.println("Thread2----------->Resource 2");

                    try {
                        Thread.sleep( millis: 100);
                    } catch (Exception e) {

                    }

                    synchronized (resource1) {
                        System.out.println("Thread2----------->Resource 1");
                    }
```

```
/home/parul/.sdkman/candidates/java/8.0.242-zulu/bin/java ...
Thread1--------->Resource 1
Thread2----------->Resource 2
```

Build completed successfully in 1 s 172 ms (a minute ago)