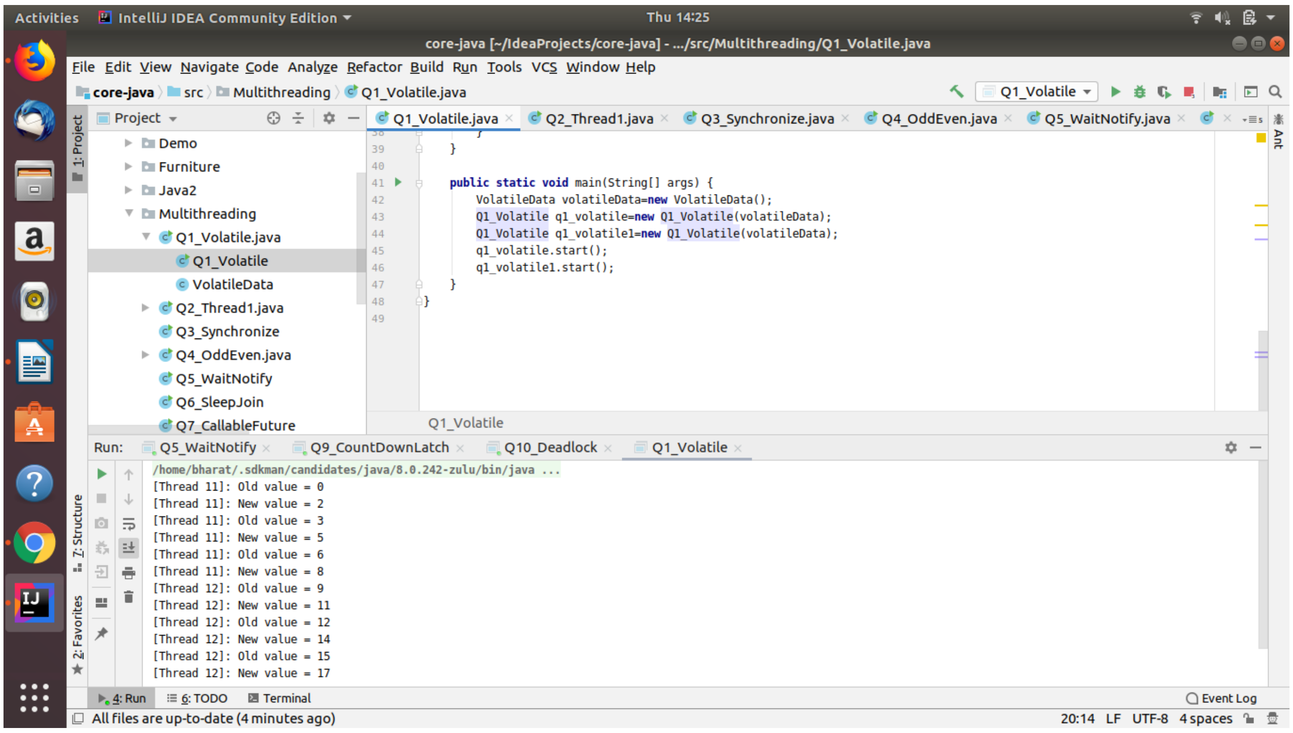
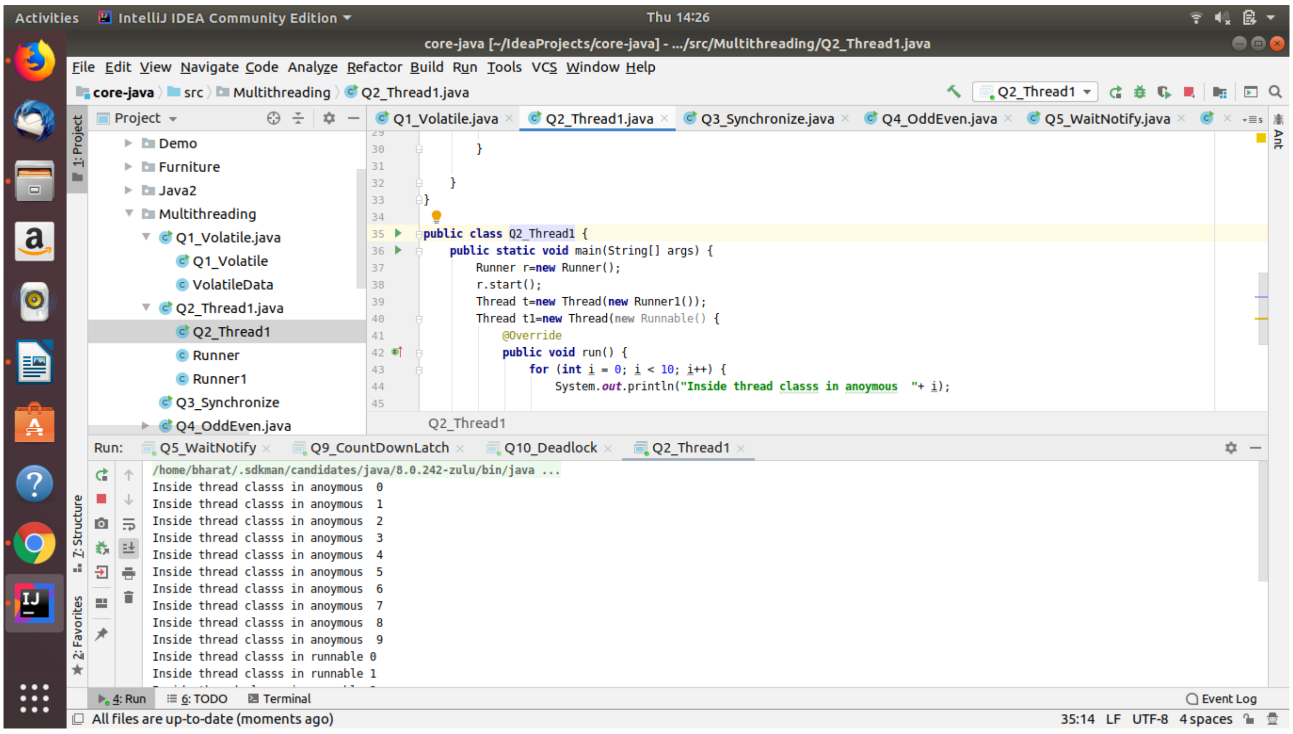
Q.1 Write a programe do to demonstrate the use of volatile keyword.



```java
    }

    public static void main(String[] args) {
        VolatileData volatileData=new VolatileData();
        Q1_Volatile q1_volatile=new Q1_Volatile(volatileData);
        Q1_Volatile q1_volatile1=new Q1_Volatile(volatileData);
        q1_volatile.start();
        q1_volatile1.start();
    }
}
```

```
/home/bharat/.sdkman/candidates/java/8.0.242-zulu/bin/java ...
[Thread 11]: Old value = 0
[Thread 11]: New value = 2
[Thread 11]: Old value = 3
[Thread 11]: New value = 5
[Thread 11]: Old value = 6
[Thread 11]: New value = 8
[Thread 12]: Old value = 9
[Thread 12]: New value = 11
[Thread 12]: Old value = 12
[Thread 12]: New value = 14
[Thread 12]: Old value = 15
[Thread 12]: New value = 17
```

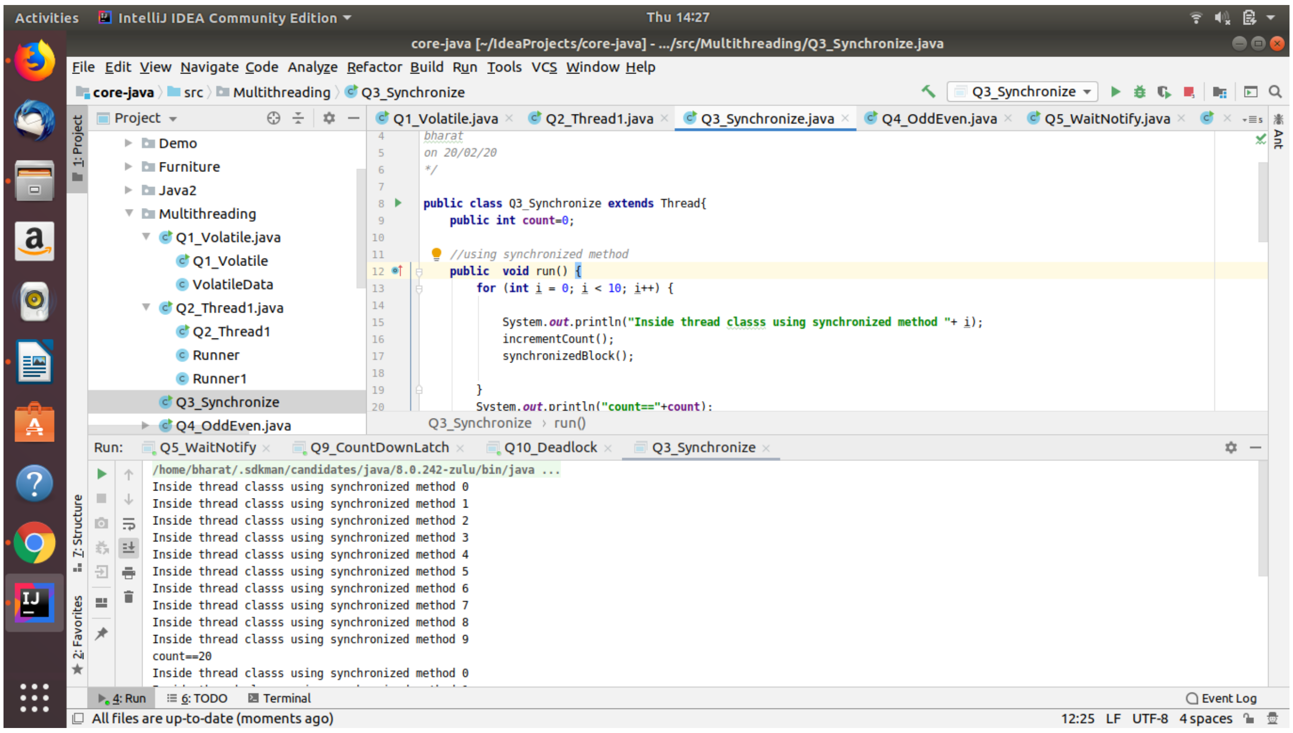
Q.2 Write a program to create a thread using Thread class and Runnable interface each



```java
            }

    }
}

public class Q2_Thread1 {
    public static void main(String[] args) {
        Runner r=new Runner();
        r.start();
        Thread t=new Thread(new Runner1());
        Thread t1=new Thread(new Runnable() {
            @Override
            public void run() {
                for (int i = 0; i < 10; i++) {
                    System.out.println("Inside thread classs in anoymous  "+ i);

```

```
/home/bharat/.sdkman/candidates/java/8.0.242-zulu/bin/java ...
Inside thread classs in anoymous  0
Inside thread classs in anoymous  1
Inside thread classs in anoymous  2
Inside thread classs in anoymous  3
Inside thread classs in anoymous  4
Inside thread classs in anoymous  5
Inside thread classs in anoymous  6
Inside thread classs in anoymous  7
Inside thread classs in anoymous  8
Inside thread classs in anoymous  9
Inside thread classs in runnable 0
Inside thread classs in runnable 1
```

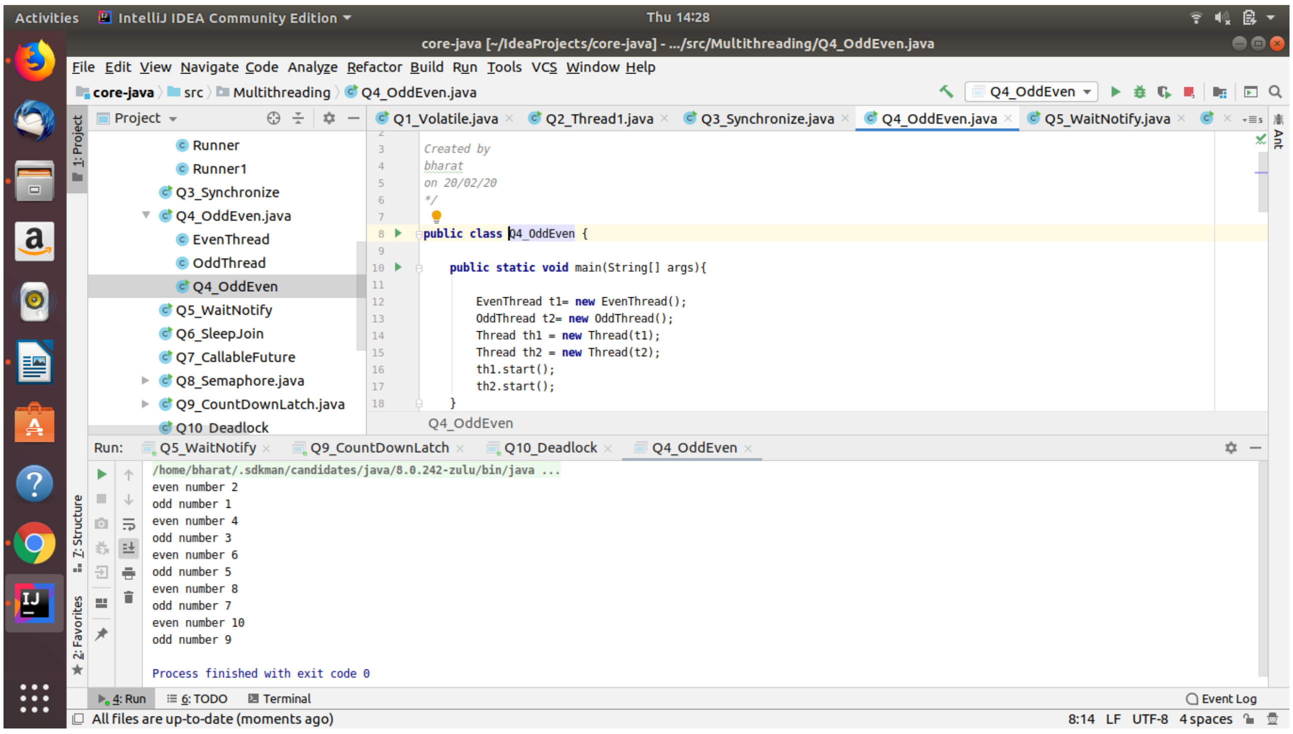
Q.3 Write a program using synchronization block and synchronization method



```
bharat
on 20/02/20
*/

public class Q3_Synchronize extends Thread{
    public int count=0;

    //using synchronized method
    public  void run() {
        for (int i = 0; i < 10; i++) {

            System.out.println("Inside thread classs using synchronized method "+ i);
            incrementCount();
            synchronizedBlock();

        }
        System.out.println("count=="+count);
```

```
/home/bharat/.sdkman/candidates/java/8.0.242-zulu/bin/java ...
Inside thread classs using synchronized method 0
Inside thread classs using synchronized method 1
Inside thread classs using synchronized method 2
Inside thread classs using synchronized method 3
Inside thread classs using synchronized method 4
Inside thread classs using synchronized method 5
Inside thread classs using synchronized method 6
Inside thread classs using synchronized method 7
Inside thread classs using synchronized method 8
Inside thread classs using synchronized method 9
count==20
Inside thread classs using synchronized method 0
```

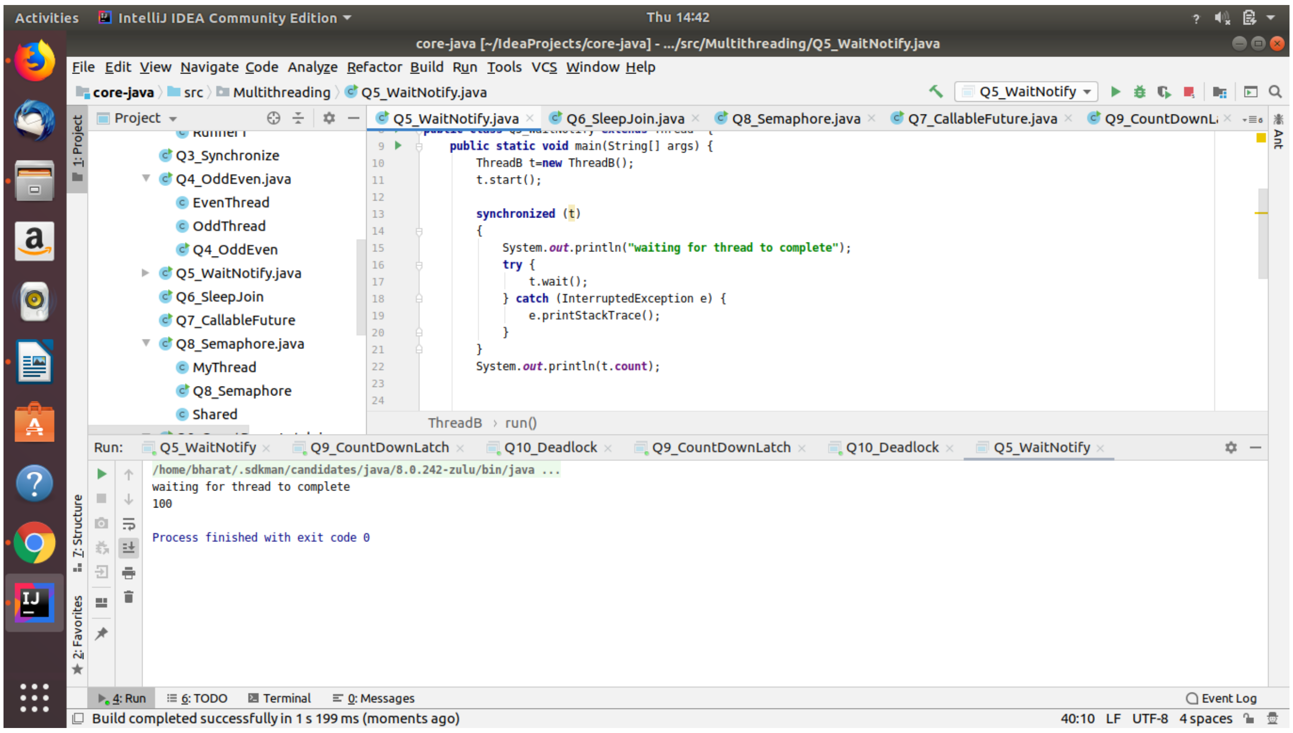
Q.4 Write a program to create a Thread pool of 2 threads where one Thread will print even numbers and other will print odd numbers.



```
Created by
bharat
on 20/02/20
*/

public class Q4_OddEven {

    public static void main(String[] args){

        EvenThread t1= new EvenThread();
        OddThread t2= new OddThread();
        Thread th1 = new Thread(t1);
        Thread th2 = new Thread(t2);
        th1.start();
        th2.start();
    }
```

```
/home/bharat/.sdkman/candidates/java/8.0.242-zulu/bin/java ...
even number 2
odd number 1
even number 4
odd number 3
even number 6
odd number 5
even number 8
odd number 7
even number 10
odd number 9

Process finished with exit code 0
```

Q.5 Write a program to demonstrate wait and notify methods.



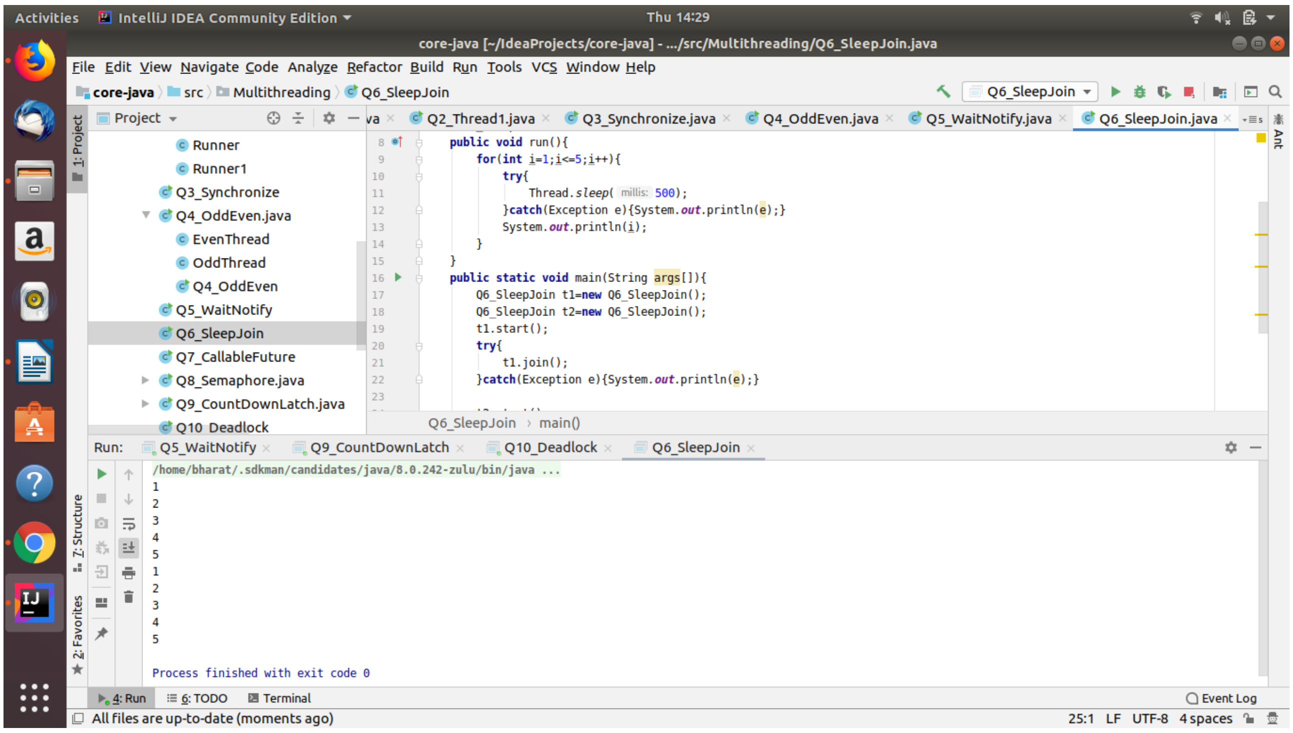
core-java [~/IdeaProjects/core-java] - .../src/Multithreading/Q5_WaitNotify.java

```java
    public static void main(String[] args) {
        ThreadB t=new ThreadB();
        t.start();

        synchronized (t)
        {
            System.out.println("waiting for thread to complete");
            try {
                t.wait();
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
        System.out.println(t.count);
```

```
/home/bharat/.sdkman/candidates/java/8.0.242-zulu/bin/java ...
waiting for thread to complete
100

Process finished with exit code 0
```

Build completed successfully in 1 s 199 ms (moments ago)

Q.6Write a program to demonstrate sleep and join methods.



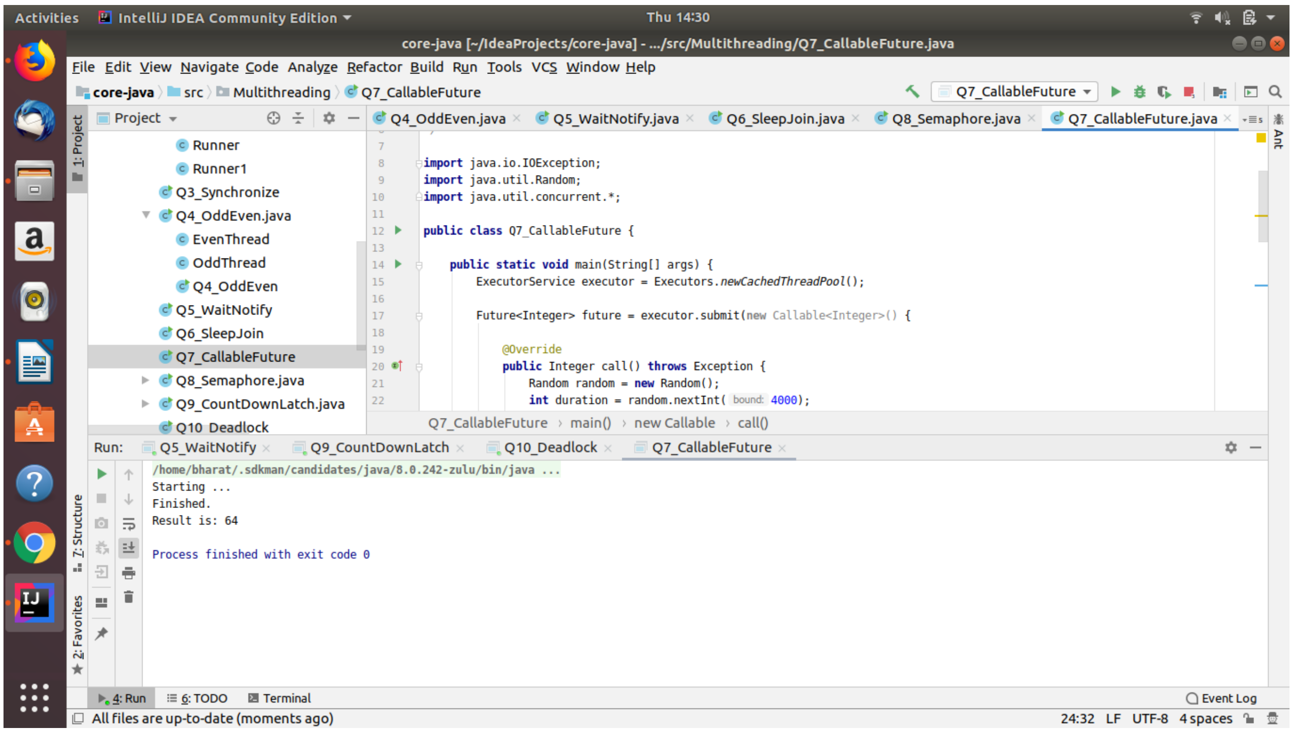
core-java [~/IdeaProjects/core-java] - .../src/Multithreading/Q6_SleepJoin.java

```java
    public void run(){
        for(int i=1;i<=5;i++){
            try{
                Thread.sleep( millis: 500);
            }catch(Exception e){System.out.println(e);}
            System.out.println(i);
        }
    }
    public static void main(String args[]){
        Q6_SleepJoin t1=new Q6_SleepJoin();
        Q6_SleepJoin t2=new Q6_SleepJoin();
        t1.start();
        try{
            t1.join();
        }catch(Exception e){System.out.println(e);}
```

```
/home/bharat/.sdkman/candidates/java/8.0.242-zulu/bin/java ...
1
2
3
4
5
1
2
3
4
5

Process finished with exit code 0
```

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

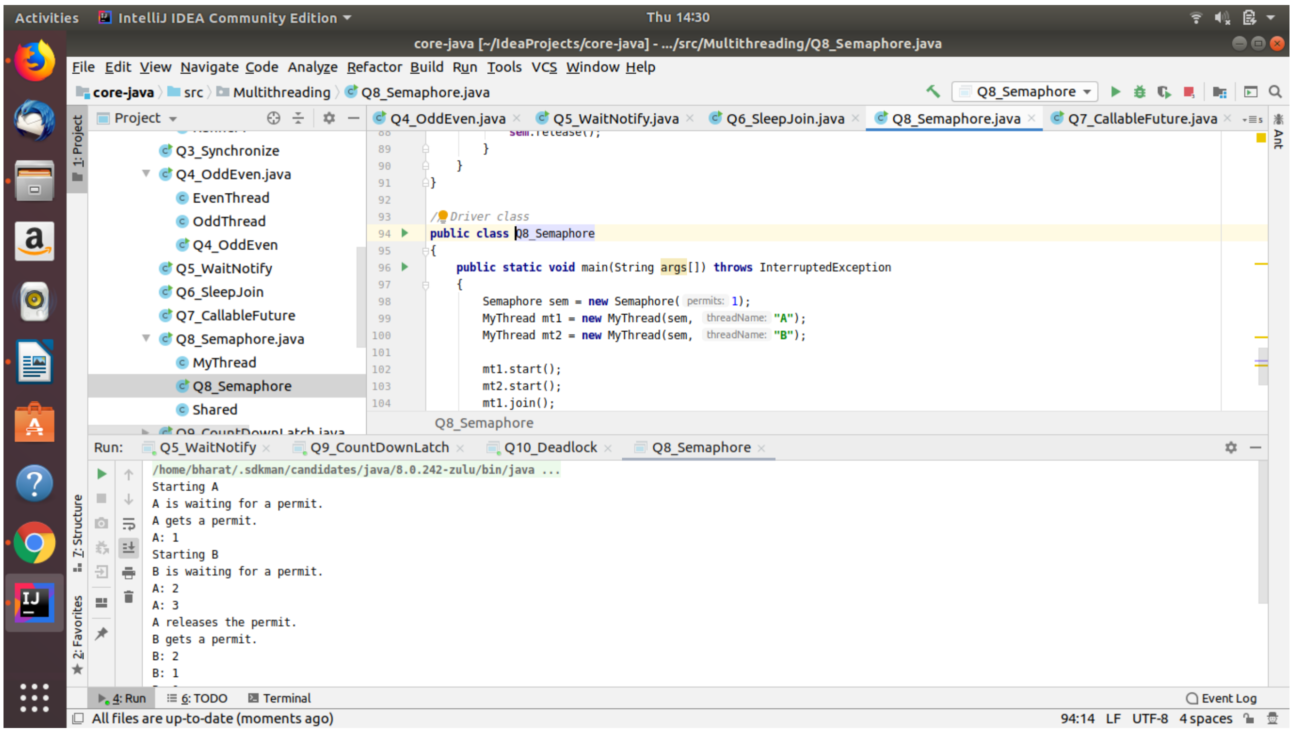
Q.7 Run a task with the help of callable and store it's result in the Future.



```java
import java.io.IOException;
import java.util.Random;
import java.util.concurrent.*;

public class Q7_CallableFuture {

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

        Future<Integer> future = executor.submit(new Callable<Integer>() {

            @Override
            public Integer call() throws Exception {
                Random random = new Random();
                int duration = random.nextInt( bound: 4000);
```

```
/home/bharat/.sdkman/candidates/java/8.0.242-zulu/bin/java ...
Starting ...
Finished.
Result is: 64

Process finished with exit code 0
```

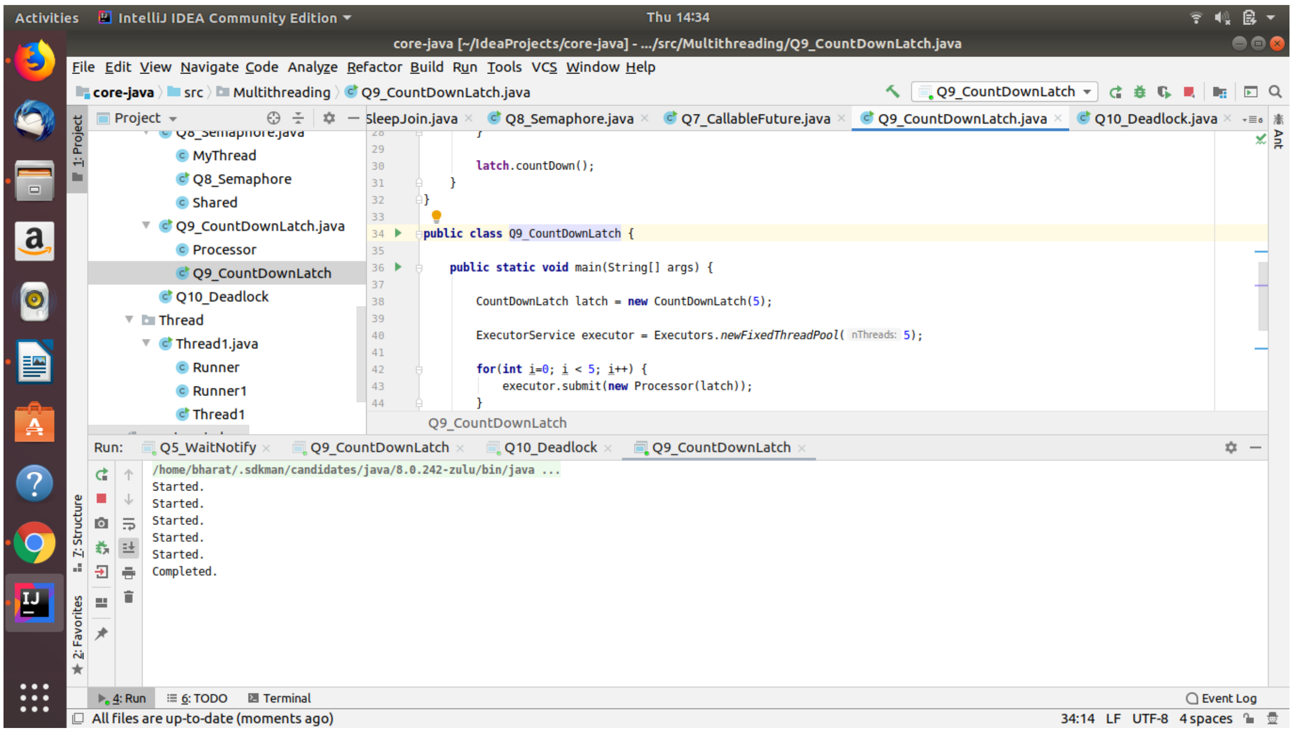
Q.8 Write a program to demonstrate the use of semaphore



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

//Driver class
public class Q8_Semaphore
{
    public static void main(String args[]) throws InterruptedException
    {
        Semaphore sem = new Semaphore( permits: 1);
        MyThread mt1 = new MyThread(sem,  threadName: "A");
        MyThread mt2 = new MyThread(sem,  threadName: "B");

        mt1.start();
        mt2.start();
        mt1.join();
```

```
/home/bharat/.sdkman/candidates/java/8.0.242-zulu/bin/java ...
Starting A
A is waiting for a permit.
A gets a permit.
A: 1
Starting B
B is waiting for a permit.
A: 2
A: 3
A releases the permit.
B gets a permit.
B: 2
B: 1
```

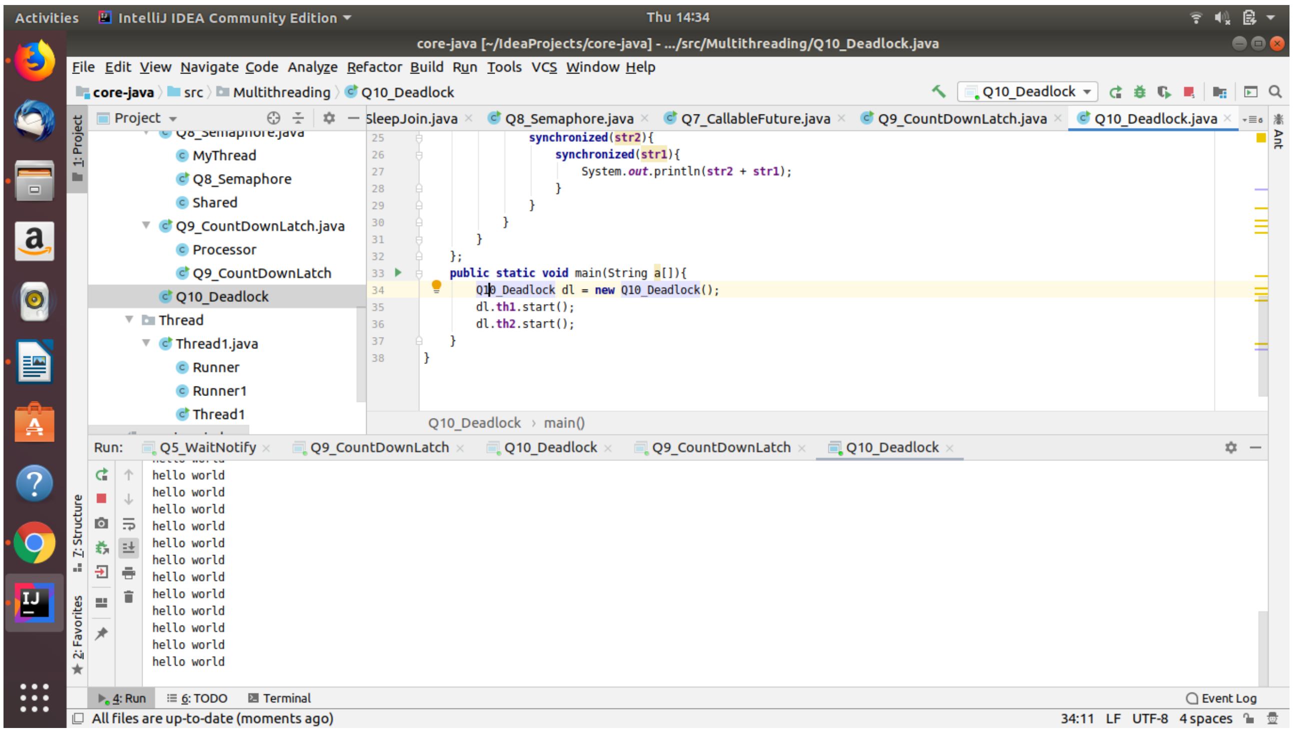
Q.9 Write a program to demonstrate the use of CountDownLatch



```java
            latch.countDown();
        }
}

public class Q9_CountDownLatch {

    public static void main(String[] args) {

        CountDownLatch latch = new CountDownLatch(5);

        ExecutorService executor = Executors.newFixedThreadPool( nThreads: 5);

        for(int i=0; i < 5; i++) {
            executor.submit(new Processor(latch));
        }
```

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

Q.10 Write a program which creates deadlock between 2 threads



```java
                synchronized(str2){
                    synchronized(str1){
                        System.out.println(str2 + str1);
                    }
                }
            }
        }
    };
    public static void main(String a[]){
        Q10_Deadlock dl = new Q10_Deadlock();
        dl.th1.start();
        dl.th2.start();
    }
}
```

```
hello world
hello world
hello world
hello world
hello world
hello world
hello world
hello world
hello world
hello world
hello world
hello world
```