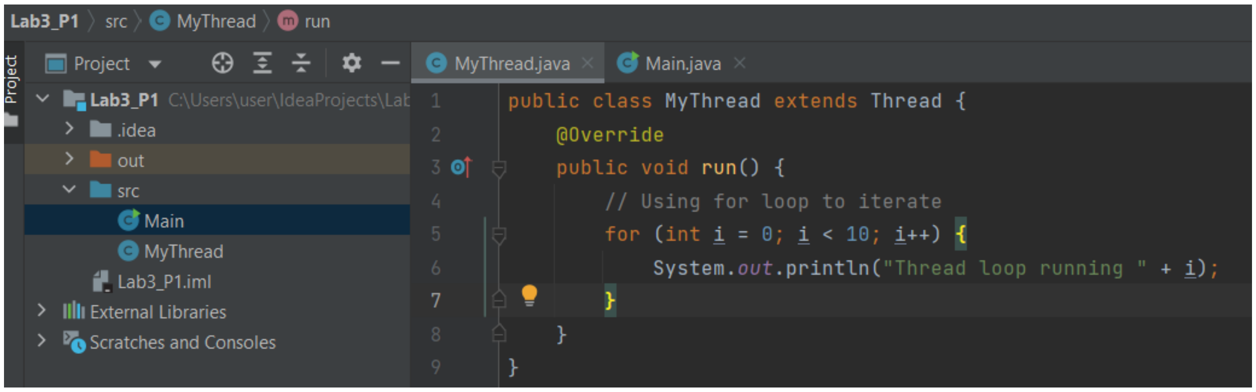
Draghici Andrei-Maria
CR 3.1B

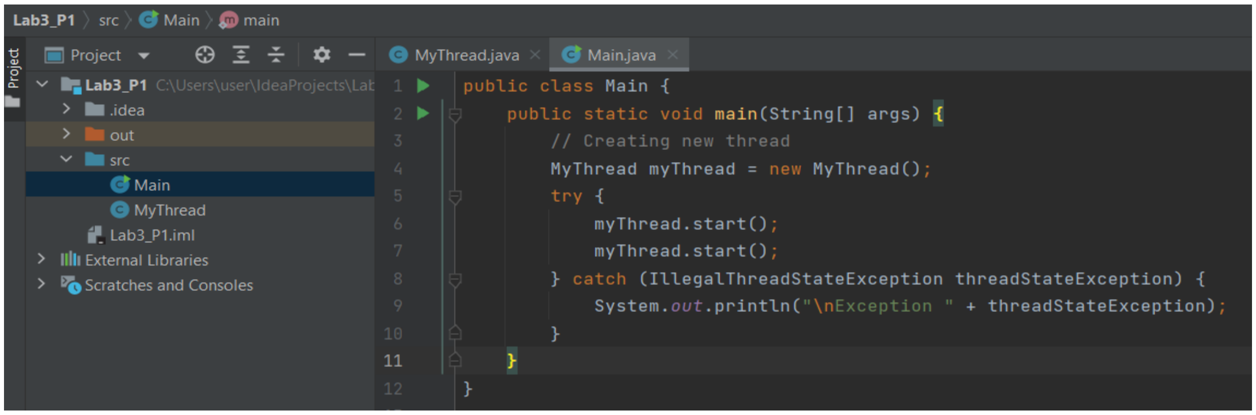
Raport Laborator 3

Problema 1

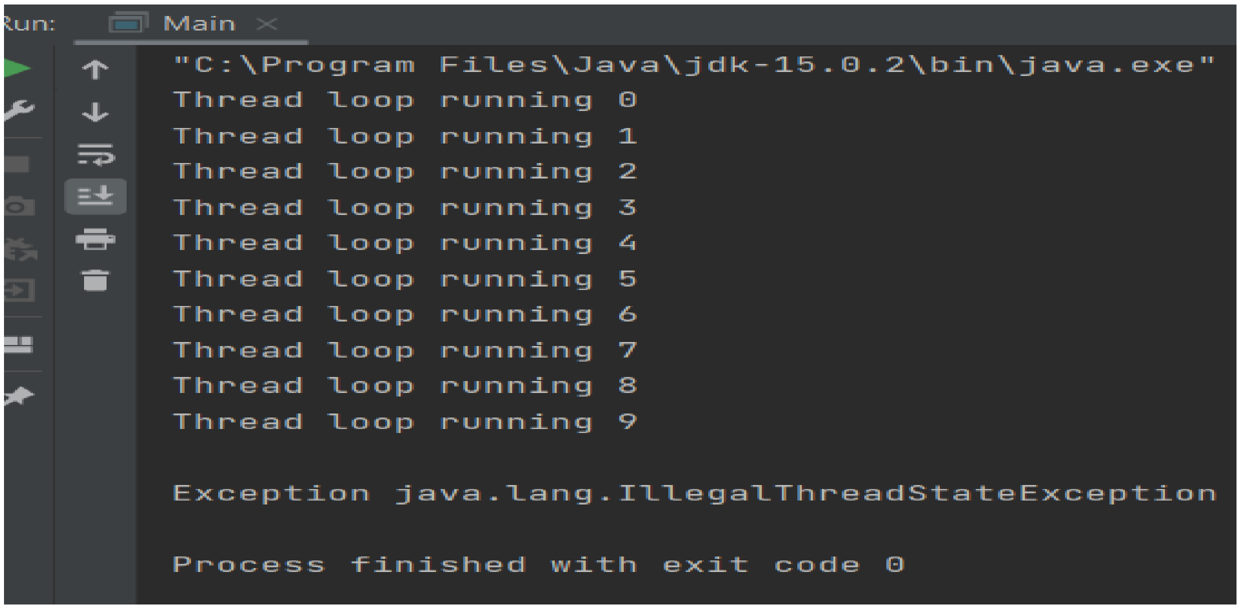
Programul afiseaza valorile de la 0, la 9 utiziand bucla for. Daca instantiem acelasi fir de doua ori, obtinem exceptia java.lang.IllegalThreadStateException dupa calculul primei instructiuni a primei instante.



```java
public class MyThread extends Thread {
    @Override
    public void run() {
        // Using for loop to iterate
        for (int i = 0; i < 10; i++) {
            System.out.println("Thread loop running " + i);
        }
    }
}
```



```java
public class Main {
    public static void main(String[] args) {
        // Creating new thread
        MyThread myThread = new MyThread();
        try {
            myThread.start();
            myThread.start();
        } catch (IllegalThreadStateException threadStateException) {
            System.out.println("\nException " + threadStateException);
        }
    }
}
```

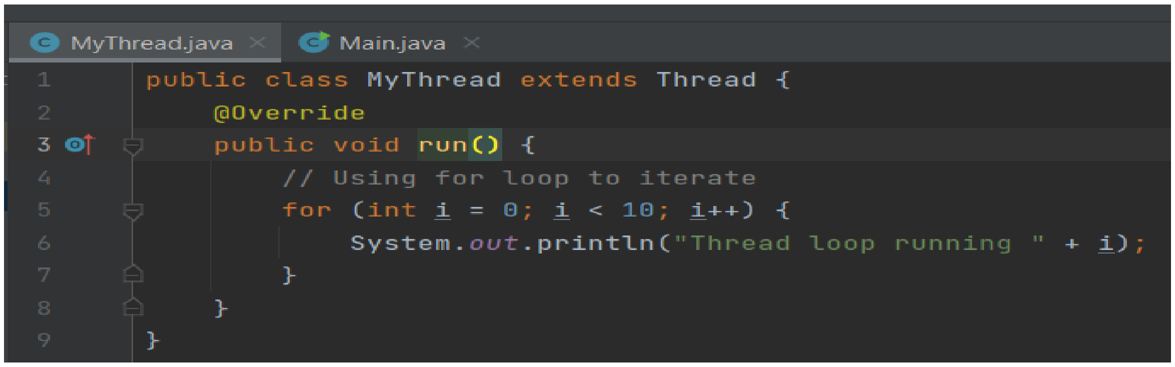


```
"C:\Program Files\Java\jdk-15.0.2\bin\java.exe"
Thread loop running 0
Thread loop running 1
Thread loop running 2
Thread loop running 3
Thread loop running 4
Thread loop running 5
Thread loop running 6
Thread loop running 7
Thread loop running 8
Thread loop running 9

Exception java.lang.IllegalThreadStateException

Process finished with exit code 0
```

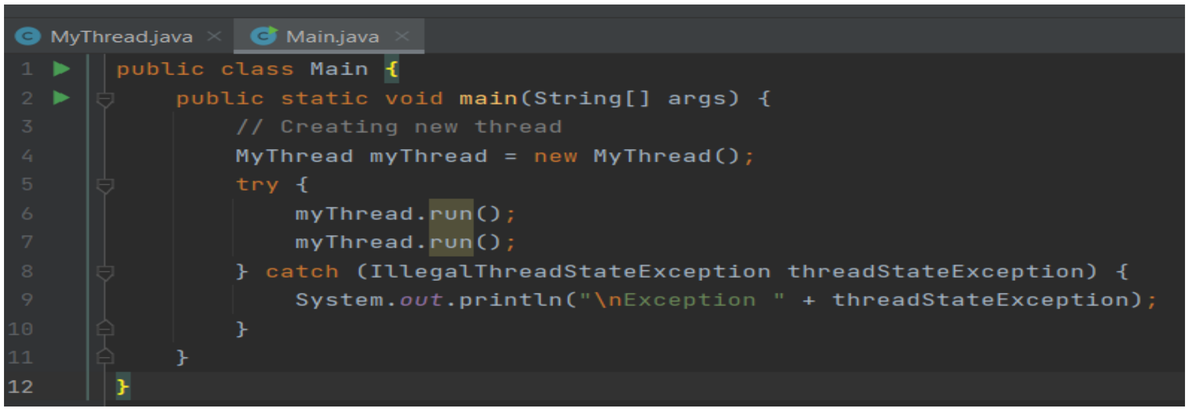
Draghici Andrei-Maria
CR 3.1B

Problema 2

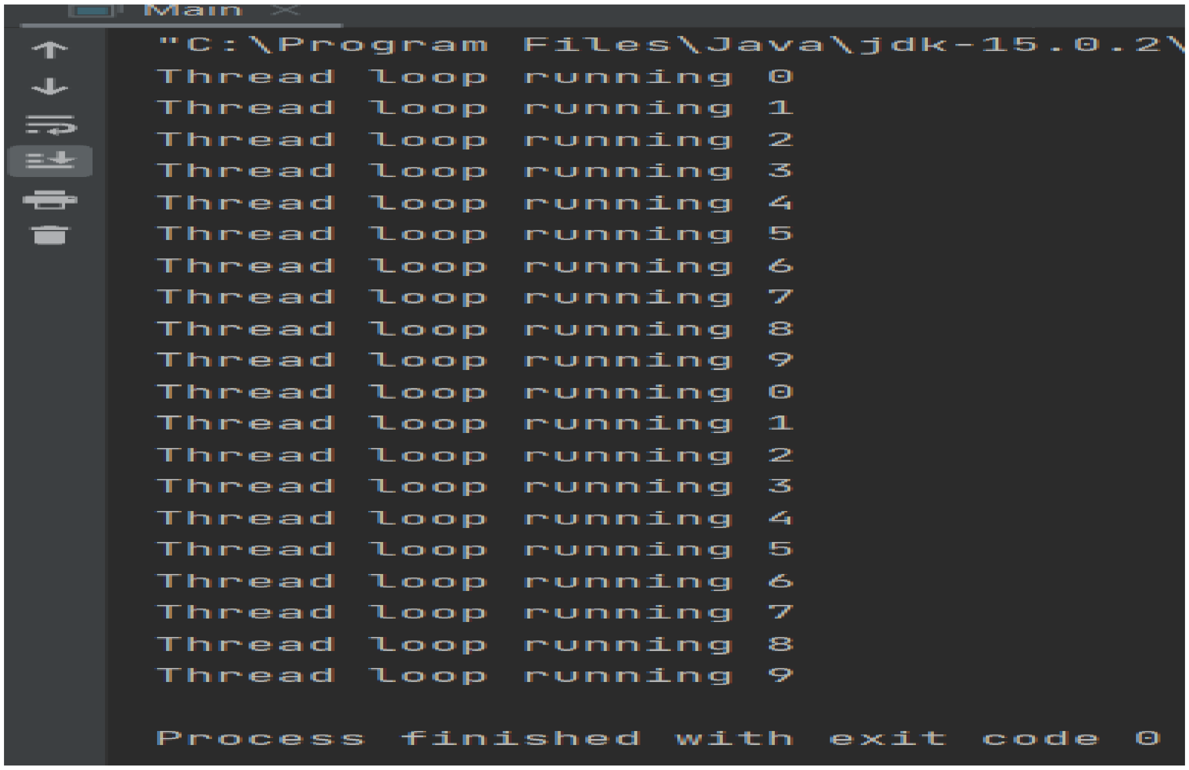
Apelul metodei run() ne returenaza pentru fiecare instanta valorile de la 0 pana la 10.



```java
public class MyThread extends Thread {
    @Override
    public void run() {
        // Using for loop to iterate
        for (int i = 0; i < 10; i++) {
            System.out.println("Thread loop running " + i);
        }
    }
}
```



```java
public class Main {
    public static void main(String[] args) {
        // Creating new thread
        MyThread myThread = new MyThread();
        try {
            myThread.run();
            myThread.run();
        } catch (IllegalThreadStateException threadStateException) {
            System.out.println("\nException " + threadStateException);
        }
    }
}
```

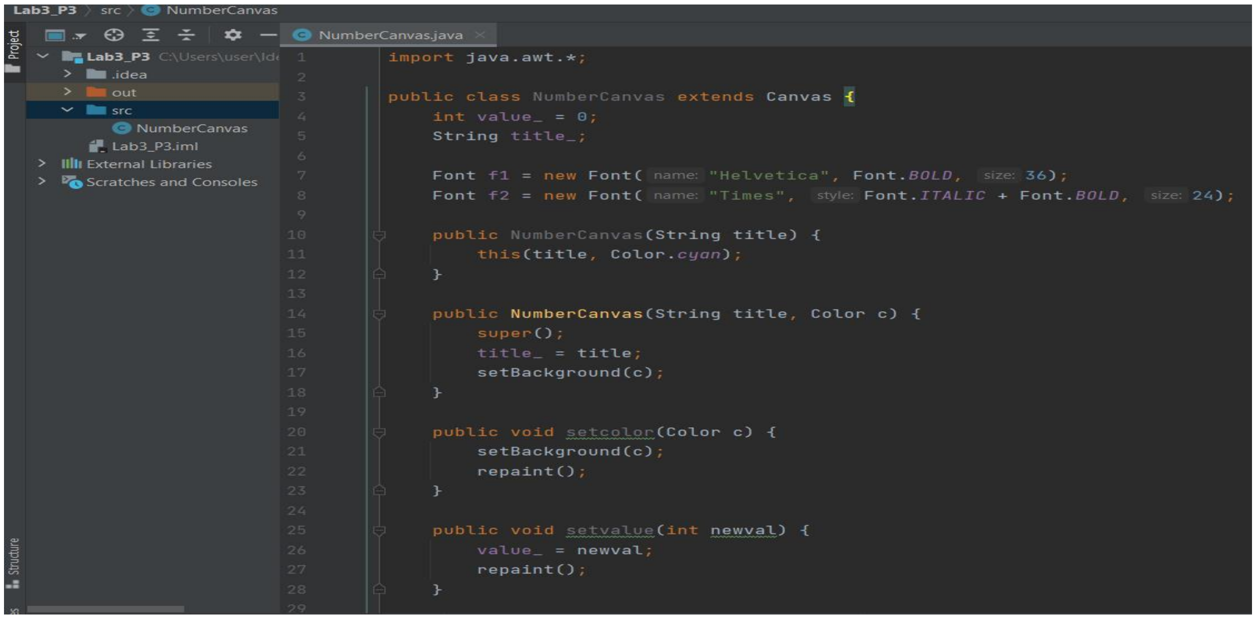


```
"C:\Program Files\Java\jdk-15.0.2\
Thread loop running 0
Thread loop running 1
Thread loop running 2
Thread loop running 3
Thread loop running 4
Thread loop running 5
Thread loop running 6
Thread loop running 7
Thread loop running 8
Thread loop running 9
Thread loop running 0
Thread loop running 1
Thread loop running 2
Thread loop running 3
Thread loop running 4
Thread loop running 5
Thread loop running 6
Thread loop running 7
Thread loop running 8
Thread loop running 9

Process finished with exit code 0
```

Draghici Andrei-Maria
CR 3.1B

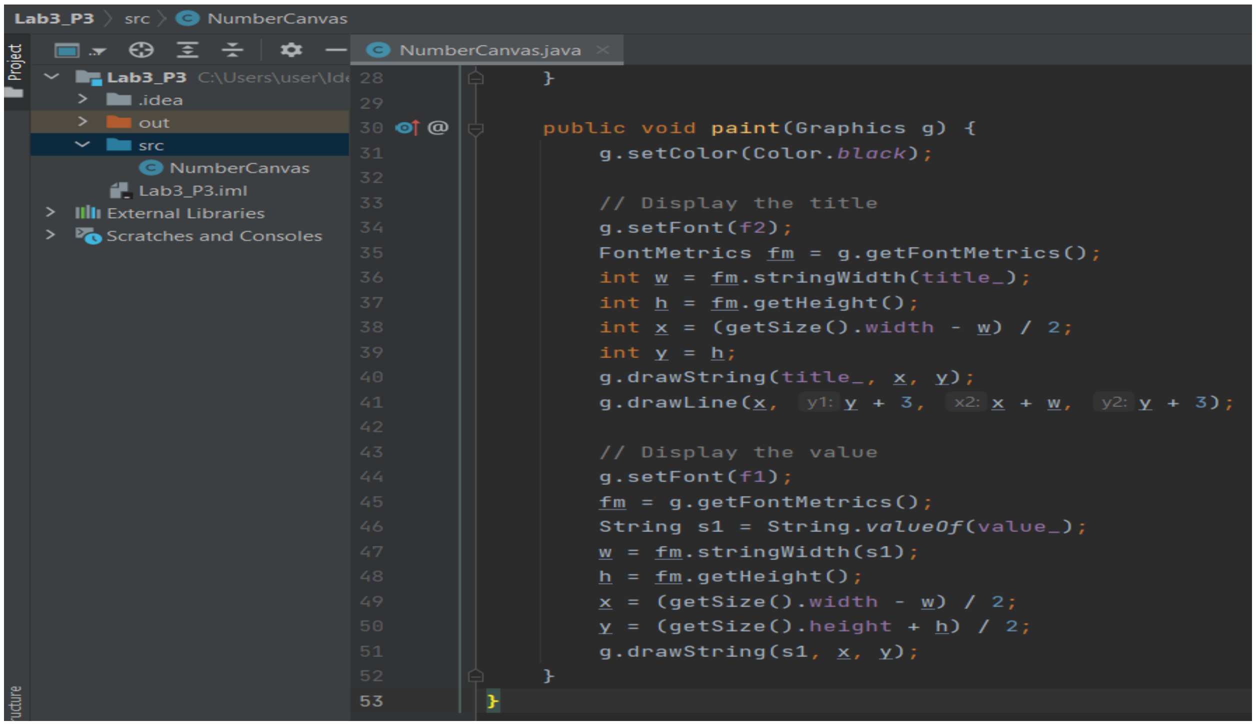
Problema 3

Am implementat exemplul NumberCanvas din cursul 1, aplicatia deseneaza o fereastra in Java Swing care contorizeaza un contor cu limitele intre 0 si 4, iar cand contorul ajunge la valoarea 0 se reseteaza la 4.



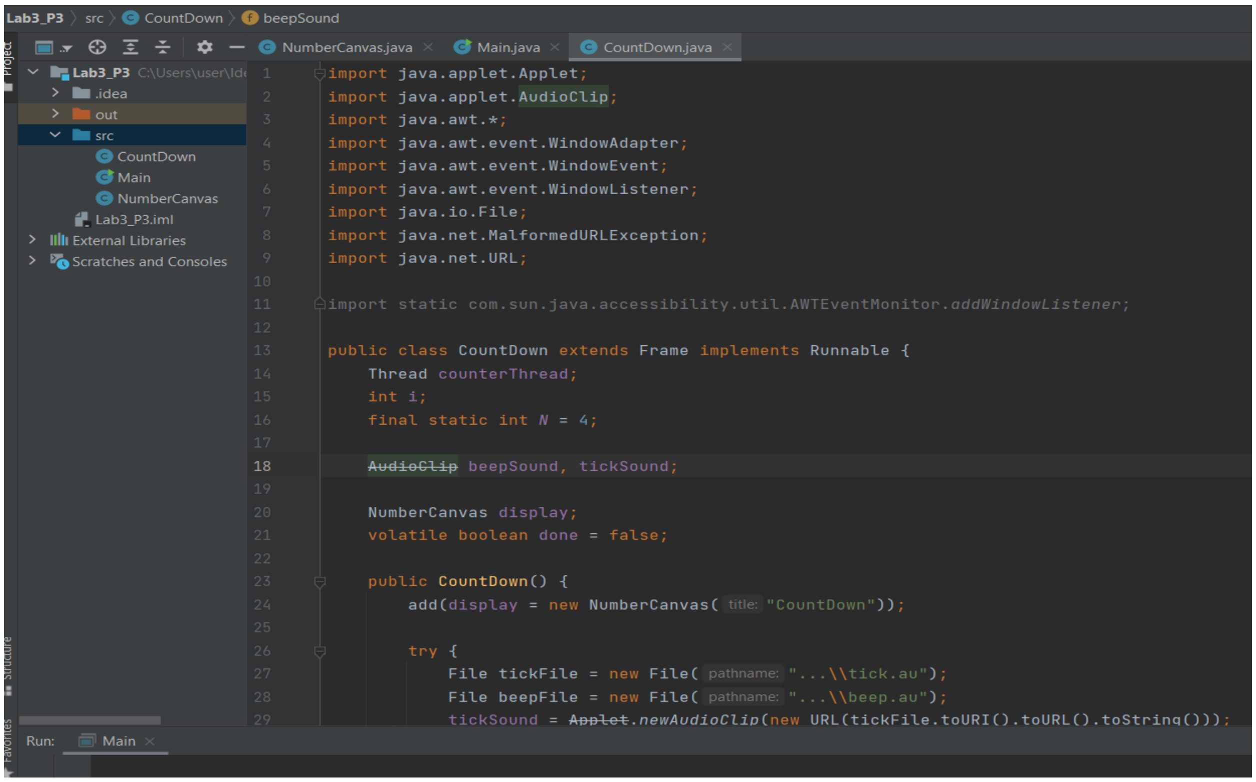
```java
import java.awt.*;

public class NumberCanvas extends Canvas {
    int value_ = 0;
    String title_;

    Font f1 = new Font( name: "Helvetica", Font.BOLD,  size: 36);
    Font f2 = new Font( name: "Times",  style: Font.ITALIC + Font.BOLD,  size: 24);

    public NumberCanvas(String title) {
        this(title, Color.cyan);
    }

    public NumberCanvas(String title, Color c) {
        super();
        title_ = title;
        setBackground(c);
    }

    public void setcolor(Color c) {
        setBackground(c);
        repaint();
    }

    public void setvalue(int newval) {
        value_ = newval;
        repaint();
    }
```

Clasa NumberCanvas implementeaza o plansa de desen Canvas unde se afiseaza valoarea numaratorului., valoare setata folosind metoda setvalue().

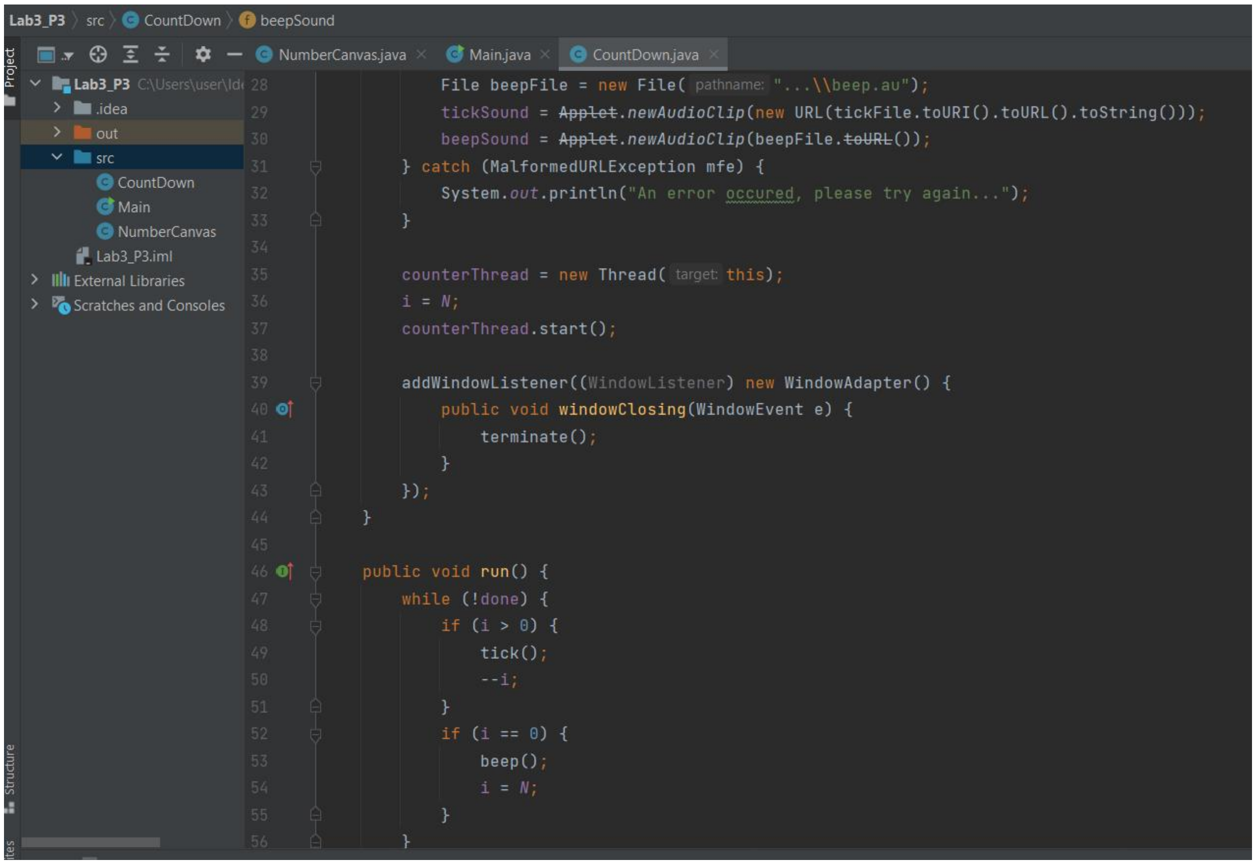


```java
    }

    public void paint(Graphics g) {
        g.setColor(Color.black);

        // Display the title
        g.setFont(f2);
        FontMetrics fm = g.getFontMetrics();
        int w = fm.stringWidth(title_);
        int h = fm.getHeight();
        int x = (getSize().width - w) / 2;
        int y = h;
        g.drawString(title_, x, y);
        g.drawLine(x,  y1: y + 3,  x2: x + w,  y2: y + 3);

        // Display the value
        g.setFont(f1);
        fm = g.getFontMetrics();
        String s1 = String.valueOf(value_);
        w = fm.stringWidth(s1);
        h = fm.getHeight();
        x = (getSize().width - w) / 2;
        y = (getSize().height + h) / 2;
        g.drawString(s1, x, y);
    }
}
```

Draghici Andrei-Maria
CR 3.1B



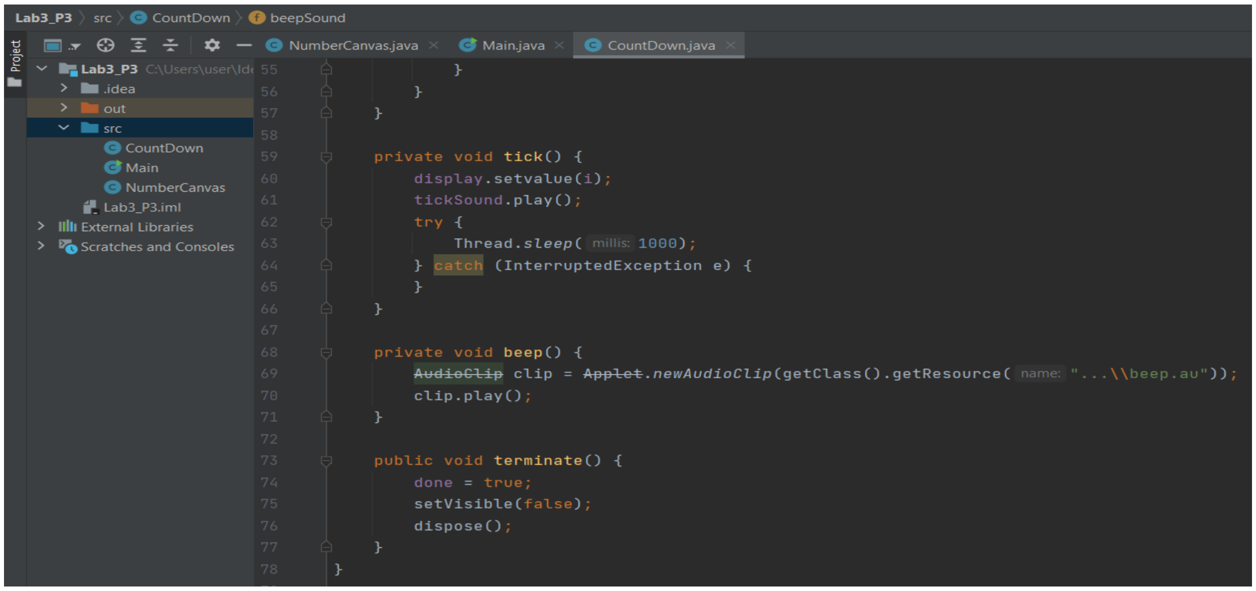
```java
import java.applet.Applet;
import java.applet.AudioClip;
import java.awt.*;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.io.File;
import java.net.MalformedURLException;
import java.net.URL;

import static com.sun.java.accessibility.util.AWTEventMonitor.addWindowListener;

public class CountDown extends Frame implements Runnable {
    Thread counterThread;
    int i;
    final static int N = 4;

    AudioClip beepSound, tickSound;

    NumberCanvas display;
    volatile boolean done = false;

    public CountDown() {
        add(display = new NumberCanvas( title: "CountDown"));

        try {
            File tickFile = new File( pathname: "...\\tick.au");
            File beepFile = new File( pathname: "...\\beep.au");
            tickSound = Applet.newAudioClip(new URL(tickFile.toURI().toURL().toString()));
```



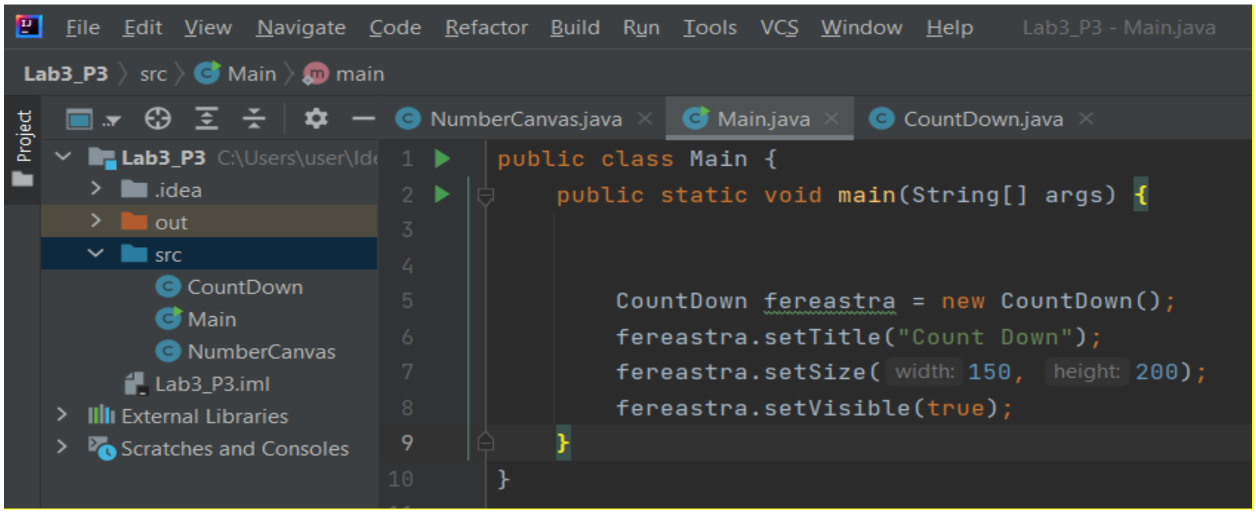
```java
            File beepFile = new File( pathname: "...\\beep.au");
            tickSound = Applet.newAudioClip(new URL(tickFile.toURI().toURL().toString()));
            beepSound = Applet.newAudioClip(beepFile.toURL());
        } catch (MalformedURLException mfe) {
            System.out.println("An error occured, please try again...");
        }

        counterThread = new Thread( target: this);
        i = N;
        counterThread.start();

        addWindowListener((WindowListener) new WindowAdapter() {
            public void windowClosing(WindowEvent e) {
                terminate();
            }
        });
    }

    public void run() {
        while (!done) {
            if (i > 0) {
                tick();
                --i;
            }
            if (i == 0) {
                beep();
                i = N;
            }
        }
```

Draghici Andrei-Maria
CR 3.1B

Numarul este intarziat cu o secunda utilizand Thread.sleep(1000). Astfel putem pastra o secunda intre fiecare contor. Contorizarea se face in metoda run().

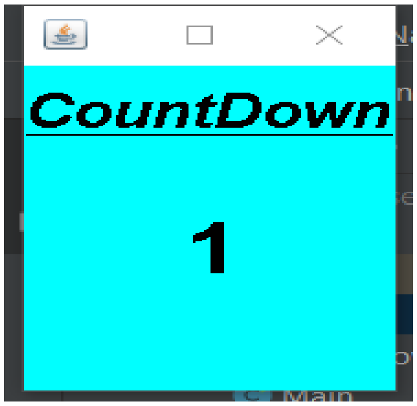


```java
            }
        }
    }

    private void tick() {
        display.setvalue(i);
        tickSound.play();
        try {
            Thread.sleep( millis: 1000);
        } catch (InterruptedException e) {
        }
    }

    private void beep() {
        ~~AudioClip~~ clip = ~~Applet~~.newAudioClip(getClass().getResource( name: "...\\beep.au"));
        clip.play();
    }

    public void terminate() {
        done = true;
        setVisible(false);
        dispose();
    }
}
```



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

        CountDown fereastra = new CountDown();
        fereastra.setTitle("Count Down");
        fereastra.setSize( width: 150, height: 200);
        fereastra.setVisible(true);
    }
}
```

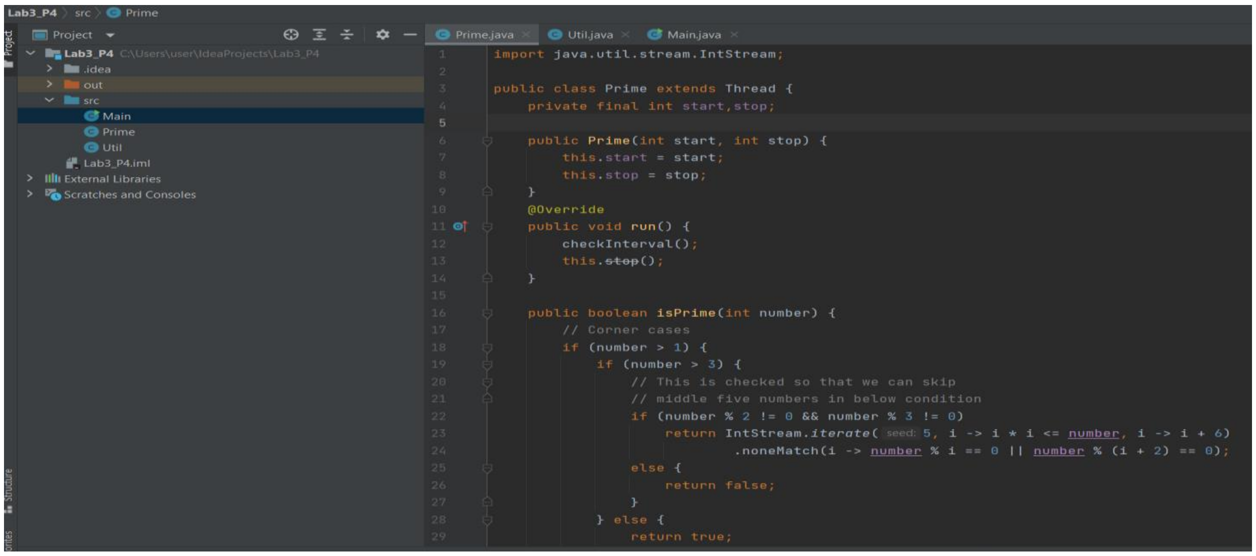
Output:



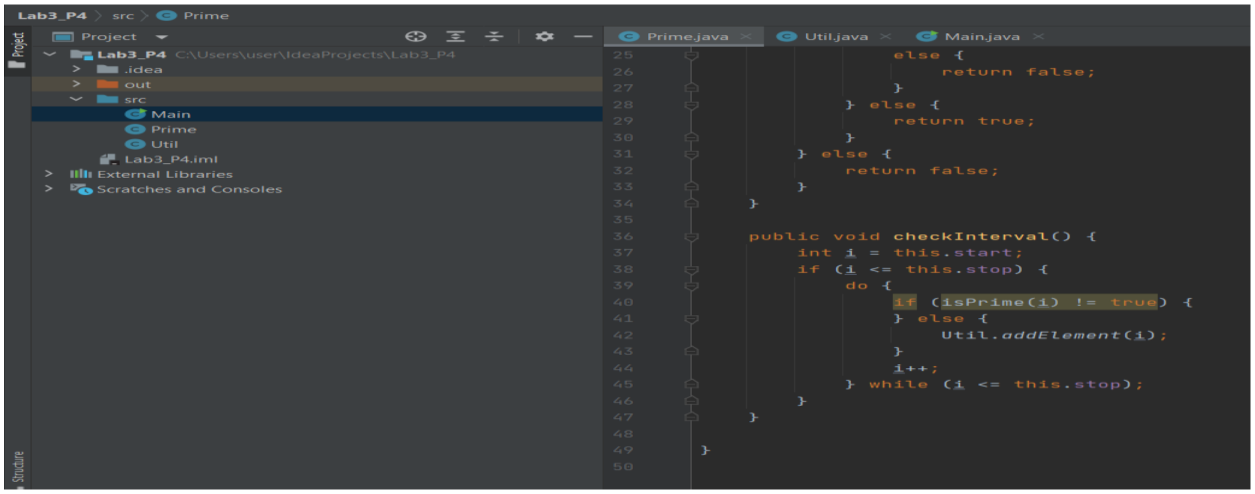
Draghici Andrei-Maria
CR 3.1B

Problema 4

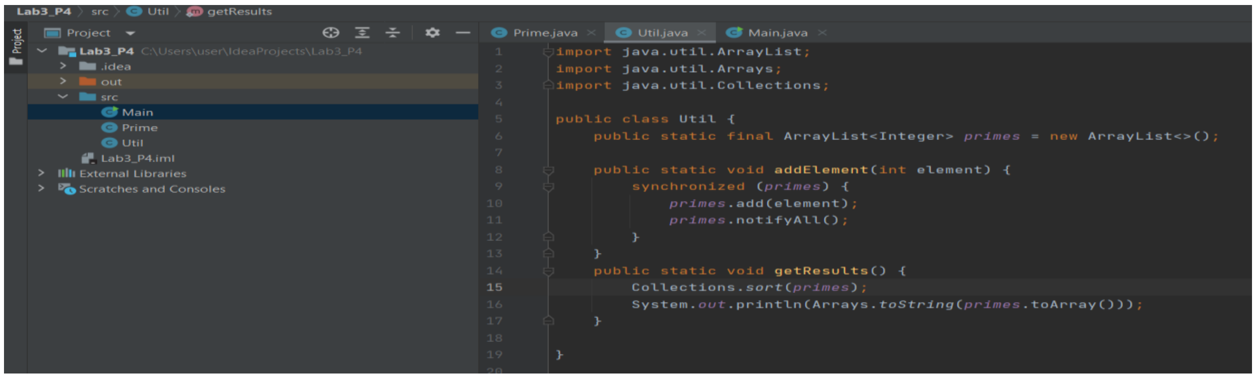
Metoda main() creeaza k fire care vor verifica daca intervalele au numere prime. Apoi, clasa Prime calculeaza verificare, utilizand mai multe conditii. Toate variabilele sunt stocate intr-o lista aflata in clasa Util care va fi afisara dupa terminarea tuturor firelor . Adaugarea de elemente / date noi este sincronizata pentru a putea evita pierderea anumitor date.



```java
import java.util.stream.IntStream;

public class Prime extends Thread {
    private final int start,stop;

    public Prime(int start, int stop) {
        this.start = start;
        this.stop = stop;
    }
    @Override
    public void run() {
        checkInterval();
        this.~~stop~~();
    }

    public boolean isPrime(int number) {
        // Corner cases
        if (number > 1) {
            if (number > 3) {
                // This is checked so that we can skip
                // middle five numbers in below condition
                if (number % 2 != 0 && number % 3 != 0)
                    return IntStream.iterate( seed: 5, i -> i * i <= number, i -> i + 6)
                            .noneMatch(i -> number % i == 0 || number % (i + 2) == 0);
                else {
                    return false;
                }
            } else {
                return true;
```

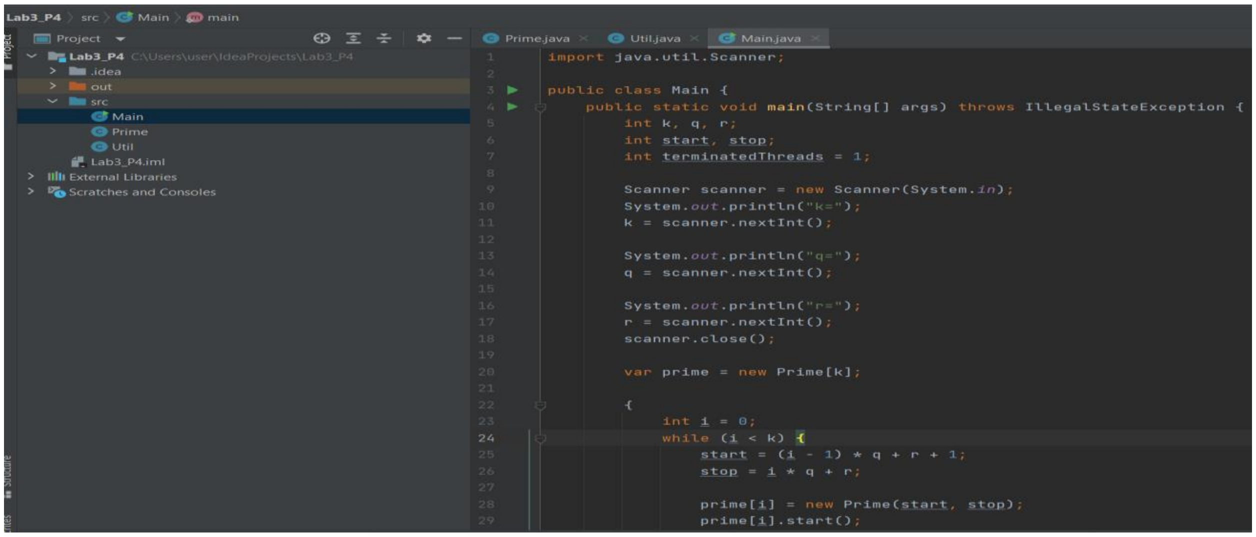


```java
                else {
                    return false;
                }
            } else {
                return true;
            }
        } else {
            return false;
        }
    }

    public void checkInterval() {
        int i = this.start;
        if (i <= this.stop) {
            do {
                if (isPrime(i) != true) {
                } else {
                    Util.addElement(i);
                }
                i++;
            } while (i <= this.stop);
        }
    }

}
```

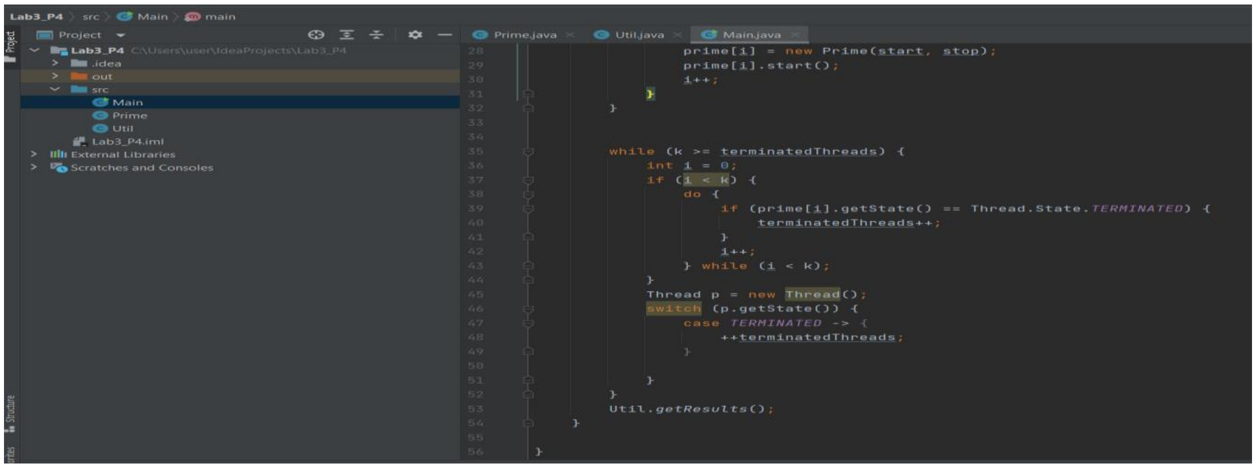


```java
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Collections;

public class Util {
    public static final ArrayList<Integer> primes = new ArrayList<>();

    public static void addElement(int element) {
        synchronized (primes) {
            primes.add(element);
            primes.notifyAll();
        }
    }
    public static void getResults() {
        Collections.sort(primes);
        System.out.println(Arrays.toString(primes.toArray()));
    }

}
```

Draghici Andrei-Maria
CR 3.1B

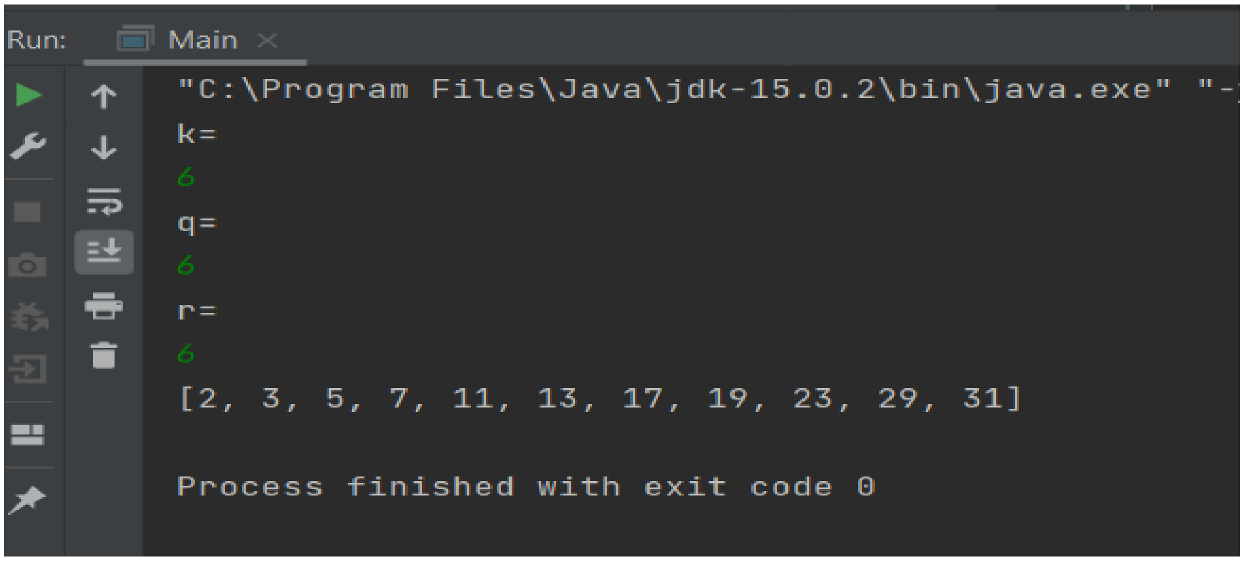


```java
import java.util.Scanner;

public class Main {
    public static void main(String[] args) throws IllegalStateException {
        int k, q, r;
        int start, stop;
        int terminatedThreads = 1;

        Scanner scanner = new Scanner(System.in);
        System.out.println("k=");
        k = scanner.nextInt();

        System.out.println("q=");
        q = scanner.nextInt();

        System.out.println("r=");
        r = scanner.nextInt();
        scanner.close();

        var prime = new Prime[k];

        {
            int i = 0;
            while (i < k) {
                start = (i - 1) * q + r + 1;
                stop = i * q + r;

                prime[i] = new Prime(start, stop);
                prime[i].start();
```



```java
                prime[i] = new Prime(start, stop);
                prime[i].start();
                i++;
            }
        }


        while (k >= terminatedThreads) {
            int i = 0;
            if (i < k) {
                do {
                    if (prime[i].getState() == Thread.State.TERMINATED) {
                        terminatedThreads++;
                    }
                    i++;
                } while (i < k);
            }
            Thread p = new Thread();
            switch (p.getState()) {
                case TERMINATED -> {
                    ++terminatedThreads;
                }

            }
        }
        Util.getResults();
    }

}
```

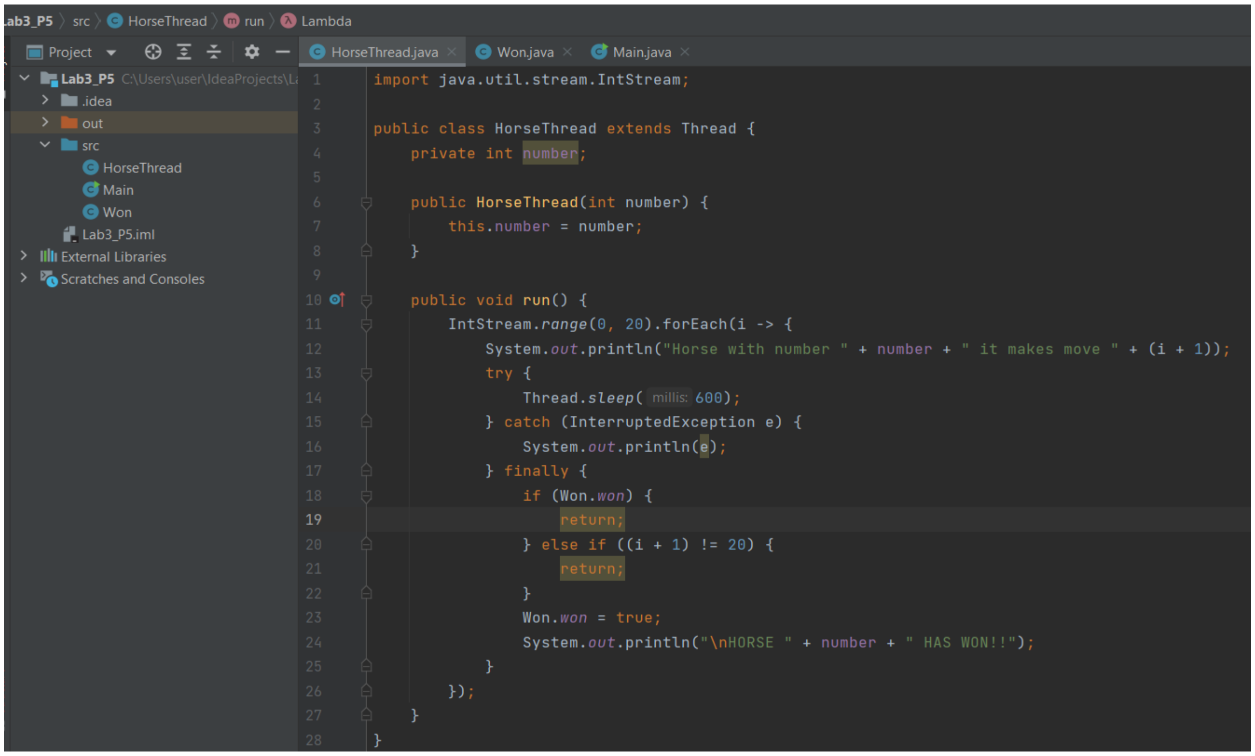
Output:



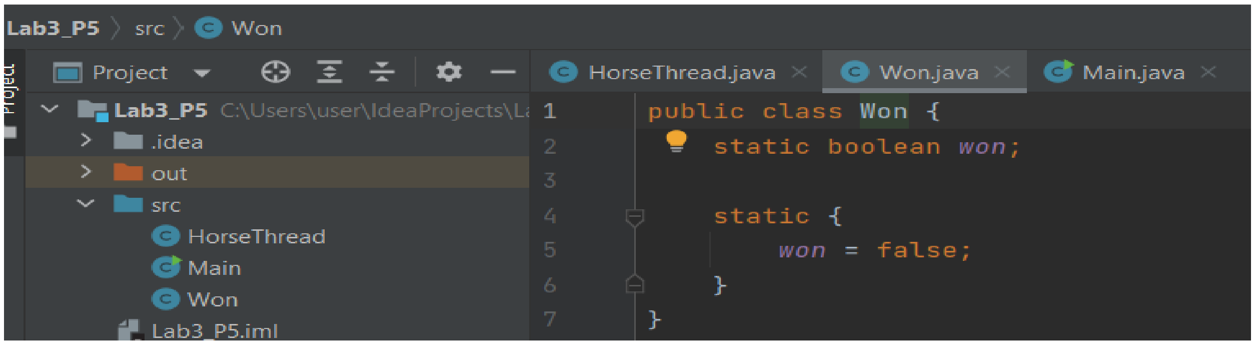
```
"C:\Program Files\Java\jdk-15.0.2\bin\java.exe" "-
k=
6
q=
6
r=
6
[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31]

Process finished with exit code 0
```

Draghici Andrei-Maria
CR 3.1B

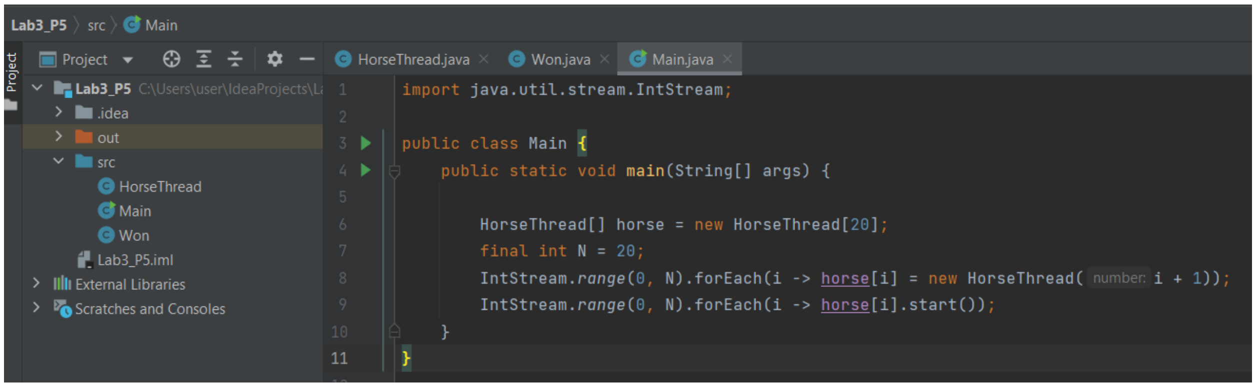
Problema 5



```java
import java.util.stream.IntStream;

public class HorseThread extends Thread {
    private int number;

    public HorseThread(int number) {
        this.number = number;
    }

    public void run() {
        IntStream.range(0, 20).forEach(i -> {
            System.out.println("Horse with number " + number + " it makes move " + (i + 1));
            try {
                Thread.sleep( millis: 600);
            } catch (InterruptedException e) {
                System.out.println(e);
            } finally {
                if (Won.won) {
                    return;
                } else if ((i + 1) != 20) {
                    return;
                }
                Won.won = true;
                System.out.println("\nHORSE " + number + " HAS WON!!");
            }
        });
    }
}
```



```java
public class Won {
    static boolean won;

    static {
        won = false;
    }
}
```



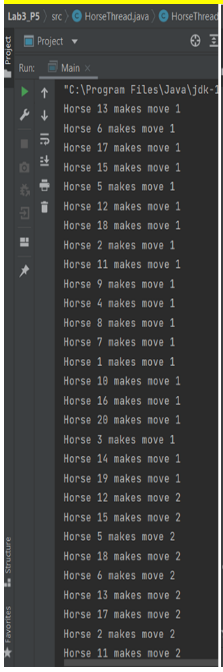
```java
import java.util.stream.IntStream;

public class Main {
    public static void main(String[] args) {

        HorseThread[] horse = new HorseThread[20];
        final int N = 20;
        IntStream.range(0, N).forEach(i -> horse[i] = new HorseThread( number: i + 1));
        IntStream.range(0, N).forEach(i -> horse[i].start());
    }
}
```

Draghici Andrei-Maria
CR 3.1B

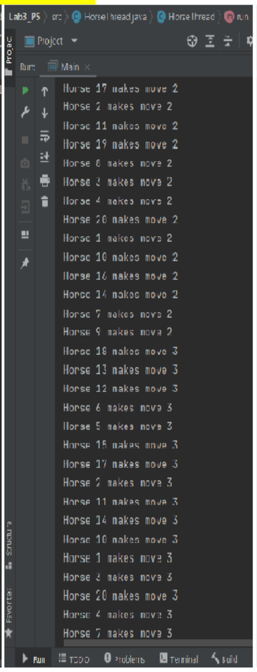
Output:

```
Horse with number 7 it makes move 1
Horse with number 10 it makes move 1
Horse with number 11 it makes move 1
Horse with number 20 it makes move 1
Horse with number 4 it makes move 1
…
Horse with number 6 it makes move 20
Horse with number 3 it makes move 20
Horse with number 11 it makes move 20
Horse with number 1 it makes move 20
Horse with number 12 it makes move 20
```

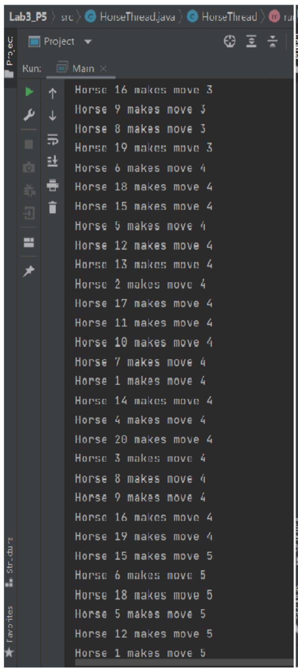
HORSE 5 HAS WON!!



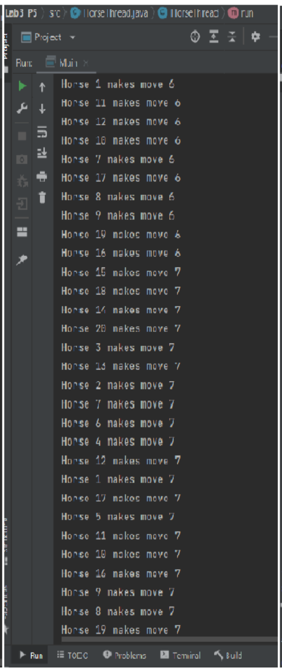
```
"C:\Program Files\Java\jdk-1
Horse 13 makes move 1
Horse 6 makes move 1
Horse 17 makes move 1
Horse 15 makes move 1
Horse 5 makes move 1
Horse 12 makes move 1
Horse 18 makes move 1
Horse 2 makes move 1
Horse 11 makes move 1
Horse 9 makes move 1
Horse 4 makes move 1
Horse 8 makes move 1
Horse 7 makes move 1
Horse 1 makes move 1
Horse 10 makes move 1
Horse 16 makes move 1
Horse 20 makes move 1
Horse 3 makes move 1
Horse 14 makes move 1
Horse 19 makes move 1
Horse 12 makes move 2
Horse 15 makes move 2
Horse 5 makes move 2
Horse 18 makes move 2
Horse 6 makes move 2
Horse 13 makes move 2
Horse 17 makes move 2
Horse 2 makes move 2
Horse 11 makes move 2
```



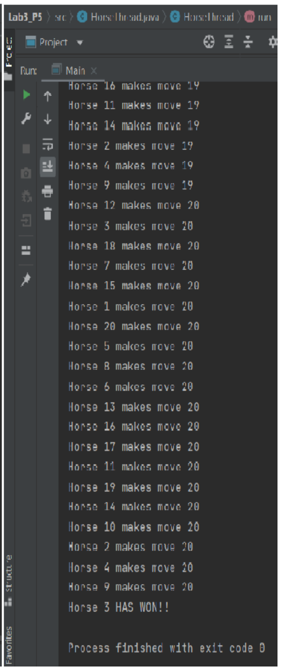
```
Horse 17 makes move 2
Horse 2 makes move 2
Horse 11 makes move 2
Horse 19 makes move 2
Horse 8 makes move 2
Horse 3 makes move 2
Horse 4 makes move 2
Horse 20 makes move 2
Horse 1 makes move 2
Horse 10 makes move 2
Horse 16 makes move 2
Horse 14 makes move 2
Horse 7 makes move 2
Horse 9 makes move 2
Horse 18 makes move 3
Horse 13 makes move 3
Horse 12 makes move 3
Horse 6 makes move 3
Horse 5 makes move 3
Horse 15 makes move 3
Horse 17 makes move 3
Horse 2 makes move 3
Horse 11 makes move 3
Horse 14 makes move 3
Horse 10 makes move 3
Horse 1 makes move 3
Horse 3 makes move 3
Horse 20 makes move 3
Horse 4 makes move 3
Horse 7 makes move 3
```



```
Horse 16 makes move 3
Horse 9 makes move 3
Horse 8 makes move 3
Horse 19 makes move 3
Horse 6 makes move 4
Horse 18 makes move 4
Horse 15 makes move 4
Horse 5 makes move 4
Horse 12 makes move 4
Horse 13 makes move 4
Horse 2 makes move 4
Horse 17 makes move 4
Horse 11 makes move 4
Horse 10 makes move 4
Horse 7 makes move 4
Horse 1 makes move 4
Horse 14 makes move 4
Horse 4 makes move 4
Horse 20 makes move 4
Horse 3 makes move 4
Horse 8 makes move 4
Horse 9 makes move 4
Horse 16 makes move 4
Horse 19 makes move 4
Horse 15 makes move 5
Horse 6 makes move 5
Horse 18 makes move 5
Horse 5 makes move 5
Horse 12 makes move 5
Horse 1 makes move 5
```



```
Horse 1 makes move 6
Horse 11 makes move 6
Horse 12 makes move 6
Horse 10 makes move 6
Horse 7 makes move 6
Horse 17 makes move 6
Horse 8 makes move 6
Horse 9 makes move 6
Horse 19 makes move 6
Horse 16 makes move 6
Horse 15 makes move 7
Horse 18 makes move 7
Horse 14 makes move 7
Horse 20 makes move 7
Horse 3 makes move 7
Horse 13 makes move 7
Horse 2 makes move 7
Horse 7 makes move 7
Horse 6 makes move 7
Horse 4 makes move 7
Horse 12 makes move 7
Horse 1 makes move 7
Horse 17 makes move 7
Horse 5 makes move 7
Horse 11 makes move 7
Horse 10 makes move 7
Horse 16 makes move 7
Horse 9 makes move 7
Horse 8 makes move 7
Horse 19 makes move 7
```



```
Horse 16 makes move 19
Horse 11 makes move 19
Horse 14 makes move 19
Horse 2 makes move 19
Horse 4 makes move 19
Horse 9 makes move 19
Horse 12 makes move 20
Horse 3 makes move 20
Horse 18 makes move 20
Horse 7 makes move 20
Horse 15 makes move 20
Horse 1 makes move 20
Horse 20 makes move 20
Horse 5 makes move 20
Horse 8 makes move 20
Horse 6 makes move 20
Horse 13 makes move 20
Horse 16 makes move 20
Horse 17 makes move 20
Horse 11 makes move 20
Horse 19 makes move 20
Horse 14 makes move 20
Horse 10 makes move 20
Horse 2 makes move 20
Horse 4 makes move 20
Horse 9 makes move 20
Horse 3 HAS WON!!

Process finished with exit code 0
```

Problema 6

Am verficat clasa Thread de pe pagina https://docs.oracle.com/javase/7/docs/api/java/lang/Thread.html si am observat urmatoarele metode, am explicat rolul fiecarei metode mai jos:

isAlive() – checks if the a certain thread is running or not, returns a True or False value;
isDaemon() – checks if this thread is alive;
notify() – used for "waking up" a certain thread;
notifyAll() – "wakes up" all the threads that are waiting;
setPriority() – set the priority of a certain thread, it's used to maximize the scheduler;
yield() – if a thread with a higher priority waits for a thread with a lower priority, the lower
getPriory() – return the priority of a thread, self explanatory;
wait() – puts a thread to "sleep", it "wakes up" when another thread finish its notify;