Java Multithread Assignment

Part 1

- I will initialize it as 2 rabbits and 10 boxes, I will not take the values from the user.
- I will not add the carrots.
- I will use myThread1 for rabbit1 and myThread2 for rabbit2.
- I will specify an array of 10 in the box.
- Each rabbit will wait 3 seconds between jumps with the sleep() method.
- I will show the “gameover” output and “who won” with the join thread method.
- I will create the threads for the rabbits as “public class **myRunnable** implements **Runnable**”.
- In the **public void run ()** method I will make a print to follow the rabbits and CPU behavior.

1st Version of My Code

Main Class

```
public class Main {

    public static void main(String[] args) {
        int numBoxes = 10; // Number of boxes

        Box[] boxes = new Box[numBoxes];
        for (int i = 0; i < numBoxes; i++) {
            boxes[i] = new Box(i);
        }

        // rabbit yarattım
        myRunnable rabbit1 = new myRunnable("duru", boxes,0);
        myRunnable rabbit2 = new myRunnable("mesude", boxes,0);

        Thread myThread1 = new Thread(rabbit1);
        Thread myThread2 = new Thread(rabbit2);

        // rabit threadlerini başlattım
        myThread1.start();
        myThread2.start();

                                    //join methodu
        //catch e düşünce output yazdırsın diye
        try {
            myThread1.join();
            myThread2.join();
        } catch (InterruptedException e) {
            e.printStackTrace();
        }

        // output
        System.out.println("Game over!");
    }
}
```

Box Class

```
public class Box {
    int boxNumber;

    public Box(int number) {
        this.boxNumber = number;
    }
}
```

myRunnable Class

```
public class myRunnable implements Runnable{
    private String rabbitName;
    private Box[] boxes;
    private int score;
    private int currentBoxIndex;
    private static volatile String winner = null;

    //constructor
    public myRunnable(String name, Box[] boxes, int currentBoxIndex) {
        this.rabbitName = name;
        this.boxes = boxes;
        this.score = 0;
        this.currentBoxIndex = currentBoxIndex;
    }

    //Runnable dan implement ettiğim için gelen method
    public void run() {
        while (currentBoxIndex < boxes.length - 1) {

                                                    long threadId =
Thread.currentThread().getId();
            System.out.println(rabbitName + " jumps to box "

                                               + (currentBoxIndex + 1)

                                               + ".................(Thread ID: " +
threadId + ")");
            try {
                Thread.sleep(3000);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
            currentBoxIndex++;

            if (currentBoxIndex >= 9 && winner == null) {
                synchronized (myRunnable.class) {
                    if (winner == null) {
                        // The first rabbit to reach box 9 wins
                        winner = rabbitName;
                        System.out.println(rabbitName + " is the
winner!!.................(Thread ID: " + threadId +")");
                        break;
                    }
                }

            }
        }
    }
```

synchronized (myRunnable.class) {

If I don't use **synchronized** key word , The output from the program would be like this:

```
…
mesude is the winner!!

duru is the winner!!
```

and we don't want this because I aim to **print the first one to arrive**.

OUTPUT:

```
duru jumps to box 1.............(Thread ID: 11)
mesude jumps to box 1.............(Thread ID: 12)
duru jumps to box 2.............(Thread ID: 11)
mesude jumps to box 2.............(Thread ID: 12)
duru jumps to box 3.............(Thread ID: 11)
mesude jumps to box 3.............(Thread ID: 12)
duru jumps to box 4.............(Thread ID: 11)
mesude jumps to box 4.............(Thread ID: 12)
duru jumps to box 5.............(Thread ID: 11)
mesude jumps to box 5.............(Thread ID: 12)
duru jumps to box 6.............(Thread ID: 11)
mesude jumps to box 6.............(Thread ID: 12)
duru jumps to box 7.............(Thread ID: 11)
mesude jumps to box 7.............(Thread ID: 12)
duru jumps to box 8.............(Thread ID: 11)
mesude jumps to box 8.............(Thread ID: 12)
duru jumps to box 9.............(Thread ID: 11)
mesude jumps to box 9.............(Thread ID: 12)
duru is the winner!!.............(Thread ID: 11)
Game over!

Process finished with exit code 0
```