0.0.0.0.0.0.0.0.0.0.1. Homework 10

1. Homework 10

Posted: October/25/2018
Due: November/4/2018 24.00

All homework solutions are due November/4/2018 24.00. I recommend to submit at least one version of all homework solutions long before due date.

1.1. Homework 10.1 (10 Points)

Objective: Using a ThreadFactory and ThreadPoolExecutor

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

You have to implement a simulation of elevator. The elevator has to serve at least two stories. The elevator has to be one thread, and each person who is using the elevator has to be a single thread.

Explanation:
The design of the this simulation is not given to you. The design of the solution is completely up to you. You need to be able to explain your design decisions to your grader.

Your Work:
You have to design the simulation. You solution will use multiple threads. You have to use for your solution and

Submission:

```
% ssh glados.cs.rit.edu # or use queeg.cs.rit.edu if glados is down
# password
# go to the directory where your solution is ...
% try hpb-grd lab10-1 'All files required'
```

1.2. Homework 10.2 (10 Points)

Objective: Modification of code based on a new requirement

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

You have to modify homework 9.3 (Producer Consumer). you have now 3 kind of producers.

Explanation:
The consumer can only take items if they're storage space has enough items. The producer produces *n* items of the kind it produces each time the production starts. The first kind of producers produces *item 1*. The second kind of producers produces *item 2*. The third kind of producers produces *item 3*. Each consumer needs to be able to consume 3 *item 1*, and 5 *item 2*, and 2 *item 3* at the time time of consumption.

Your Work:
n, *k*, how many consumer, how many producer, and the length of the storage is given as command line arguments.

Submission:

```
% ssh glados.cs.rit.edu # or use queeg.cs.rit.edu if glados is down
# password
# go to the directory where your solution is ...
% try hpb-grd lab10-2 'All files required'
```

1.3. Homework 10.3 (10 Points)

Objective: Analyzing multi threaded programs

Grading:

Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

Given are the following programs:

```
import java.util.*;

public class T_5 extends Thread    {
    static Object o = new Object();
    static int    counter = 0;
    int id;

    public T_5(int id)  {
        this.id = id;
    }
    public void run () {
        if ( ++counter == 1 )
                o = new Object();

        synchronized ( o ) {
            System.err.println(id + " --->" );
            System.err.println(id + " <---" );
        }
    }

    public static void main (String args []) {
        new T_5(1).start();
        new T_5(2).start();
        new T_5(3).start();
    }
}
```

Source Code: Src/30/T_5.java

```
import java.util.*;

public class T_6 extends Thread    {
    String o = " ";
    int id;

    public T_6(int id)  {
        this.id = id;
    }
    public void run () {
        synchronized ( o ) {
            System.err.println(id + " --->" );
            System.err.println(id + " <---" );
        }
    }

    public static void main (String args []) {
        new T_6(1).start();
        new T_6(2).start();
        new T_6(3).start();
    }
}
```

Source Code: Src/30/T_6.java

```

import java.util.*;

public class T_7 extends Thread    {
        static Object o = new Object();
        static int    counter = 0;
        int id;

        public T_7(int id)  {
            this.id = id;
        }
        public void run () {
            if ( ++counter == 1 )
                    o = new Object();
            else
                    o = new Object();

            synchronized ( o ) {
                System.err.println(id + " --->" );
                System.err.println(id + " <---" );
            }
        }

        public static void main (String args []) {
            new T_7(1).start();
            new T_7(2).start();
            new T_7(3).start();
        }
}
```

Source Code: Src/30/T_7.java

Your Work:

What are all possible outputs of these programs? You have to add your answers to the programs and intstrument the code so such the code produces the outcome. This is typicaly done with the use of *sleep()*.

Submission:

```
% ssh glados.cs.rit.edu # or use queeg.cs.rit.edu if glados is down
# password
# go to the directory where your solution is ...
% try hpb-grd lab10-3 ’All files required’
```