Process Introduction: Exercise II - C# Thread Creation

Create Threads with C#

Exercise a)

Write a C# program which starts 20 threads in a loop. The method executed by each thread shall print out the current thread ID. The thread ID can be determined by the following statment:

```
Thread.CurrentThread.ManagedThreadId
```

The property `ManagedThreadId` returns the ID of the thread as `integer`. The output should be similar as the follwoing:

```
Thread id: 5
Thread id: 6
Thread id: 4
Thread id: 7
Thread id: 8
Thread id: 9
Thread id: 10
Thread id: 11
Thread id: 12
Thread id: 13
Thread id: 14
Thread id: 15
Thread id: 16
Thread id: 17
Thread id: 18
Thread id: 19
Thread id: 20
Thread id: 21
Thread id: 22
Main: Thread creation finished.
Thread id: 23
```

The code for the output`"Main: Creating Threads ..."` is located straight before the loop, the output `"Main: Thread creation finished."` is located straight after the loop.

Exerrcise b)

Extend your program by a static integer variable `number` which shall be initialzed by `0`. Each time a new thread has been created, the variable will be incremented by 1. Additionally to the thread ID, the static variable `number` shall be printed out by the method executed by the thread. Since `number` is a static variable, access from this method is possible. Ideally the output looks as the following:

```
Main: Creating Threads ...
Thread id: 1 Number 0
Thread id: 2 Number 1
Thread id: 3 Number 2
```

```
...

Main: Thread creation finished.
```

In real life, the output may look like this:

```
Main: Creating Threads ...
Thread id: 4Thread id: 5 Number: 5
Thread id: 9 Number: 6
Thread id: 8 Number: 6
Thread id: 10Thread id: 7 Number: 7
Thread id: 6 Number: 8
 Number: 5
 Number: 7
Thread id: 11Thread id: 12 Number: 9
 Number: 9
Thread id: 13 Number: 10
Thread id: 14 Number: 11
Thread id: 15 Number: 12
Thread id: 16 Number: 13
Thread id: 17 Number: 14
Thread id: 18 Number: 15
Thread id: 19 Number: 16
Thread id: 20 Number: 17
Thread id: 21 Number: 18
Thread id: 22 Number: 19
Main: Thread creation finished.
Thread id: 23 Number: 20
```

Put the whole C# solution in a `*.zip` archive and submit this to Moodle.

Task c)

Please answer to the following questions:

- Task a) How can it happen, that the **output is not in sequence**?
- Task b) Why are some output lines longer than others and why isn't every line printed exactely like `Thread id: <id> Number <number>`?
- Do you have any other ideas (except of a static variable) how to hand over data to the method executed by the thread?
- Do you have any ideas how to hand over return values from the thread ot the parent proces?

Pls. write the answers to a separate MarkDown file and submit this file. **Do not submit the *.pdf file!!**

Threads erzeugen mit C#

Aufgabe a)

Erstellen Sie ein C# Programm in welchem 20 Threads in einer Schleife erzeugt werden. Die Methode, die jeder dieser Threads ausführt, soll seine aktuelle Thread ID ausgeben. Die Thread ID erhalten Sie durch folgendes Statement:

```
Thread.CurrentThread.ManagedThreadId
```

Das Property `ManagedThreadId` gibt die ID des Threads als `Integer` zurück. Die Ausgabe könnte in etwa folgender- maßen aussehen:

```
Thread id: 5
Thread id: 6
Thread id: 4
Thread id: 7
Thread id: 8
Thread id: 9
Thread id: 10
Thread id: 11
Thread id: 12
Thread id: 13
Thread id: 14
Thread id: 15
Thread id: 16
Thread id: 17
Thread id: 18
Thread id: 19
Thread id: 20
Thread id: 21
Thread id: 22
Main: Thread creation finished.
Thread id: 23
```

Der Code für die Ausgabe `"Main: Creating Threads ..."` steht unmittelbar vor der Schleife, die Ausgabe `"Main: Thread creation finished."` unmittelbar nach der Schleife.

Aufgabe b)

Erweitern Sie Ihr Programm um eine statische Integer Variable `number` welche Sie mit `0` initialisieren. Jedes mal, nachdem Sie einen neuen Thread erzeugt haben, wird diese Variable um 1 erhöht. In der Methode, die von den Threads ausgeführt wird, geben Sie nun nach der Thread Id auch die statische Variable `number` aus. Da `number` statisch ist, können Sie ja in dieser Methode darauf zugreifen. Im Idealfall sieht die Ausgabe so aus:

```
Main: Creating Threads ...
Thread id: 1 Number 0
Thread id: 2 Number 1
Thread id: 3 Number 2

...

Main: Thread creation finished.
```

In der Realität wird die Ausgabe jedoch vielleicht so aussehen:

```
Main: Creating Threads ...
Thread id: 4Thread id: 5 Number: 5
Thread id: 9 Number: 6
Thread id: 8 Number: 6
Thread id: 10Thread id: 7 Number: 7
Thread id: 6 Number: 8
 Number: 5
 Number: 7
Thread id: 11Thread id: 12 Number: 9
 Number: 9
Thread id: 13 Number: 10
Thread id: 14 Number: 11
Thread id: 15 Number: 12
Thread id: 16 Number: 13
Thread id: 17 Number: 14
Thread id: 18 Number: 15
Thread id: 19 Number: 16
Thread id: 20 Number: 17
Thread id: 21 Number: 18
Thread id: 22 Number: 19
Main: Thread creation finished.
Thread id: 23 Number: 20
```

Geben Sie die gesamte C# Solution in ein *.zip Archiv und geben Sie diese ab.

Aufgabe c)

Beantworten Sie folgende Fragen:

- Wie kann es sein, dass bei Aufgabe a) die Ausgaben nicht der Reihe nach erfolgen?
- Warum sind bei Aufgabe b) manche Zeilen länger als andere, bzw. warum steht nicht in jeder Zeile genau `Thread id: <id> Number <number>`?
- Abgesehen von einer statischen Variable, haben Sie noch andere Ideen, wie Sie Daten an die vom Thread ausgeführte Methode weiterreichen können?
- Haben Sie Ideen wie Sie Rückgabewerte vom Thread auf den Elternprozess übergeben können?

Beantworten Sie die Fragen in einem seperaten Markdown File und geben sie dieses ab (NICHT die *.pdf Datei!)