Assignments for week 11

Anders Degn Lapiki, Jacob Kjærulff Furberg, Jonas Ishøj Nielsen

12/11/2020

Contents

1 11.1 - green

See `ReadTweets.kt`

1.1 11.1.12 - green

See `ReadTweets.kt`

1.2 11.1.3 - green

Can't get it to work.

1.3 11.1.4 - Yellow

Can't get it to work.

1.4 11.1.5 - red

Can't get it to work.

1.5 11.1.6 - red

Can't get it to work.

2 11.2 - Coroutine creation time

2.1 11.2.1 - green

See `StartThreadTiming.java`.

```
Average time to start 1.124.000 with 10 repeats
Average time to start 630.305   with 20 repeats
Average time to start 573.466   with 50 repeats
Average time to start 399.877   with 100 repeats
Average time to start 447.670   with 500 repeats
Average time to start 334.127   with 1000 repeats
```

Increasing cores decreases time to start. Determining the start time and creating the thread should be made atomic to avoid this. Printing in thread would increase time returned, as a core would be occupied longer. However our repeated solution only prints when all threads are done.

2.2 11.2.2 - green

See `kotlin11_2_2.kt`.

```
    Average time to start: 1        14,623,137
    Average time to start: 10       832,246
    Average time to start: 20       670,942
    Average time to start: 50       1,556,275
    Average time to start: 100      4,775,352
    Average time to start: 500      16,234,158
    Average time to start: 1000     22,059,475
    Average time to start: 10000    51,799,061
    Average time to start: 100000   1,296,711,347
```

Coroutines doesn't appear to start faster, but it was also started in a browser which may affect result.

2.3 11.2.3 - yellow

Removing the print statements and only taking the average time gives the following results:
Taking the average after 5 using kotlin11$_{23}runsgives$:

```
    Average time to start: 100 4,801,751
    Average time to start: 100 5,236,937
    Average time to start: 100 4,616,845
    Average time to start: 100 4,563,662
    Average time to start: 100 7,873,509
    Average time to start: 100 6,446,489
\end{verbatim}
While the other

However after after 5 using \texttt{kotlin11_2_3_2} runs gives:
\begin{verbatim}
    Average time to start: 100 42,187,381
    Average time to start: 100 11,386,917
    Average time to start: 100 19,369,379
    Average time to start: 100 15,691,117
    Average time to start: 100 8,998,332
    Average time to start: 100 9,544,058
\end{verbatim}
\newline

Without removing the print statements the result is that:\\
After running the 2 methods several times, the following values were generated:
\begin{verbatim}
...
Measurement1 Starttime: 24,011,110
Measurement1 Starttime: 24,060,610
Measurement1 Starttime: 24,124,812
Measurement1 Starttime: 24,177,560
```

```
Measurement1 Starttime: 24,229,900
Measurement1 Starttime: 24,296,380
Now measure 2
...
Measurement2 Starttime: 7,327,567
Measurement2 Starttime: 7,326,615
Measurement2 Starttime: 7,325,820
Measurement2 Starttime: 7,324,634
Measurement2 Starttime: 7,323,233
Measurement2 Starttime: 7,321,982
\end{verbatim}
The difference in the reported started up times also appear to be a factor of 4.\\
No we didn't trust it, therefore we changed and improved the code when we made the first exercise. The

\subsection{11.2.4 - Yellow}
Using default gives:\\
\begin{verbatim}
    Average time to start: 100 336,261
    Average time to start: 100 3,109,020
    Average time to start: 100 2,377,758
    Average time to start: 100 11,353,999
    Average time to start: 100 9,586,355
\end{verbatim}

Using Unconfined gives:\\
\begin{verbatim}
    Average time to start: 100 71,812
    Average time to start: 100 108,075
    Average time to start: 100 113,731
    Average time to start: 100 70,898
    Average time to start: 100 104,886
\end{verbatim}
Conclusion: Unconfined is many times faster.


\section{11.3 - Handling timing info}
\subsection{11.3.1 - Yellow}
Done. See \texttt{kotlin11_3_1.kt}

\subsection{11.3.2 - Yellow}
1000000: out of memory\\
10000: overflow for standard deviation\\
1000: stable results + doable\\

\subsection{11.3.3 - Yellow}
Done. See \texttt{kotlin11_3_1.kt}.\\
```

```
Doesn't give any noticeable differences.


\subsection{11.3.4 - red}



\begin{comment}
   %https://tex.stackexchange.com/questions/277560/counting-characters-in-a-section-of-a-documen
    \quickwordcount{main}
    \quickcharcount{main}
   \todo{There are \thechar characters and approximately \theword spaces.
   That makes approximately \the\numexpr\theword+\thechar\relax\ characters total.}
\end{comment}
```