**CCT College Dublin**

**Assessment Cover Page**

|  |  |
| --- | --- |
| **Module Title:** | Concurrent Systems |
| **Assessment Title:** | CA1: Application of Concurrency to Common Tasks |
| **Lecturer Name:** | Sam Weiss |
| **Student Full Name:** | Leonardo Oliveira |
| **Student Number:** | 2021361 |
| **GitHub Rep Link:** | https://github.com/CCT-Dublin/ca1-LeoStudentCCT.git  https://github.com/LeoStudentCCT/ca1-personal-LeoStudentCCT.git |
| **Assessment Due Date:** | 24/03/2024 |
| **Date of Submission:** | 24/03/2024 |

**Declaration**

|  |
| --- |
| By submitting this assessment, I confirm that I have read the CCT policy on Academic Misconduct and understand the implications of submitting work that is not my own or does not appropriately reference material taken from a third party or other source. I declare it to be my own work and that all material from third parties has been appropriately referenced. I further confirm that this work has not previously been submitted for assessment by myself or someone else in CCT College Dublin or any other higher education institution. |

Contents

[Task 1: Calculating Standard Deviation 3](#_Toc162198150)

[Advantages of Concurrency 3](#_Toc162198151)

[Disadvantages of Concurrency 3](#_Toc162198152)

[Task 2: Multiplying Matrices 3](#_Toc162198153)

[Advantages of Concurrency 3](#_Toc162198154)

[Disadvantages of Concurrency 3](#_Toc162198155)

[Task 3: Sorting Numbers with MergeSort 4](#_Toc162198156)

[Advantages of Concurrency 4](#_Toc162198157)

[Disadvantages of Concurrency 4](#_Toc162198158)

[Reading Data from CSV File 4](#_Toc162198159)

[Advantages of Concurrency 4](#_Toc162198160)

[Disadvantages of Concurrency 4](#_Toc162198161)

[Bibliography 6](#_Toc162198162)

# Task 1: Calculating Standard Deviation

## Advantages of Concurrency

* + Efficiency: If we’re dealing with a large dataset, calculating the standard deviation can be a hard task. By splitting the task into different threads, we can cut down some of the overall computation time.
  + Scalability: If the dataset becomes very large, a concurrent solution can take advantage of additional processing power more effectively than a single-threaded solution.

## Disadvantages of Concurrency

* + Overhead: Creating and managing multiple threads can introduce overhead, especially if the dataset is small. Overhead can be explained, basically, as computational resources and time to run the program. When a program runs concurrently for example, it needs to create, coordinate, and manage multiple threads or processes. Including tasks like scheduling threads, synchronizing shared data, and communicating between threads. All of these tasks take time and computational resources, which can add to the total running time of the program. This is what we refer to as “overhead”. The overhead, in this case, might outweigh the benefits of concurrency.
  + Debugging: Debugging concurrent programs can be more difficult than debugging single-threaded programs due to the potential for race conditions and other timing-related bugs.

# Task 2: Multiplying Matrices

## Advantages of Concurrency

* + Speed: Matrix multiplication is a task that can be easily divided into subtasks, each of which can be performed independently. So, using concurrency to solve it makes a lot more sense as with can significantly speed up the computation of the problem.
  + Responsiveness: If the matrix multiplication is part of a larger program, using a concurrent approach can keep the program responsive while the calculations are being performed.

## Disadvantages of Concurrency

* + Complexity: Implementing concurrent matrix multiplication can be more complex than a single-threaded one. We need to make sure that the multiple threads do not interfere with each other, and that the results are put back together correctly.
  + Resource contention: If the matrices are large, multiple threads working on it will start competing for memory resources, which eventually will lead to slower performance.

# Task 3: Sorting Numbers with MergeSort

## Advantages of Concurrency

* + Performance: Sorting algorithms like merge sort can benefit from concurrency, as different parts of the list can be sorted independently.
  + Parallelism: Merge sort is in its nature parallelizable, which is basically when something can be divided into smaller independent subtasks that can be processed simultaneously. This makes it an ideal candidate for a concurrent approach.

## Disadvantages of Concurrency

* + Synchronization: We need to be sure that the sub-lists are merged correctly, but that can be hard. If done incorrectly, we are left with errors.
  + Overhead: The overhead of managing multiple threads and combining their results can outweigh the benefits of concurrency, especially if we are dealing with small lists, which is the case with this assignment.

# Reading Data from CSV File

## Advantages of Concurrency

* + I/O Efficiency: If the CSV file is large, reading the file can take a bit of time. So, if I were to use multiple threads to read different parts of the file concurrently, I could potentially reduce the I/O time.
  + Throughput: By reading multiple parts of the file concurrently, there can potentially be an increase in the throughput, reducing the total time to read the file.

## Disadvantages of Concurrency

* + Ordering: If the order of the data in the file is important, concurrent reading can make everything harder because I would need to make sure that the data is put back together in the correct order.
  + Complexity: Reading a file concurrently can introduce additional complexity, as we need to manage multiple threads and ensure that the data is read in the correct order.

My program, for example, does not use concurrency to read the CVS file. This is because since it’s a small file, the time and effort to create something like that would easily outweigh the benefits and add complexity to my program. I could eventually run into problems if I made a mistake on the code, and the results of the calculation could be well off if the numbers were not put back together in the correct way. Sure, if it was a bigger file, with way more numbers, maybe that could’ve been better, but that’s not the case here.

In conclusion, concurrency can provide significant benefits in terms of performance, it also introduces additional complexity and potential mistakes. So, the decision to use concurrency is really down to the requirements of the tasks the program needs to perform. It’s also important to test concurrent programs to make sure they work correctly and avoid running into errors and unexpected results.

# Bibliography

(No date) *Concurrency pros and cons*. Available at: https://www.cs.huji.ac.il/course/2010/os/ex-slides/ex5.pdf (Accessed: 22 March 2024).

Bhatia, J. (2023) *Understanding the pros and cons of concurrency*, *Learn Coding Anywhere Anytime -PW Skills Blog*. Available at: https://pwskills.com/blog/pros-and-cons-of-concurrency/ (Accessed: 21 March 2024).

Deep, S. (no date) *Understanding the Pros and Cons of Concurrency*, *Coding ninjas studio*. Available at: https://www.codingninjas.com/studio/library/understanding-the-pros-and-cons-of-concurrency (Accessed: 21 March 2024).

GfG (2023) *What is concurrency testing in software testing?*, *GeeksforGeeks*. Available at: https://www.geeksforgeeks.org/concurrency-testing-software-testing/ (Accessed: 22 March 2024).