New University logo


|  |  |  |  |
| --- | --- | --- | --- |
| Academic Year | 2024 | | |
| Semester | Fall | Winter | Summer |
| Course Code - Name | CSCI 3310 – System Programming | | |
| Instructor | Dr. Razi Iqbal | | |
| Assessment | Assignment 1 |  | |

**Instructions:**

In order to obtain maximum marks in this assignment, please ensure the followings:

* Submit this assignment by writing your solution in this document under the Solution heading below. Do not use a separate document.
* Copy and paste the code from your editor in this document for both the parts.
* Make sure to take a screenshot of the output of both the parts.
* **Make sure you take the screenshot of the Activity Monitor for Requirement 1 and 2.**
* **For Requirement 3, make sure to provide a concrete reason in plain English.**
* This assignment has a weightage of **10%** marks of the course.
* This is **NOT** a group assignment so **students having similar assignments will get a 0.**
* The assignment deadline is **midnight Feb. 11, 2024**. Submissions after the deadline will not be accepted.

**Question 1**

Threads in modern operating systems are used as a mean of improving the efficiency of the programs and avoiding lags in performance. Most of the modern operating systems provide threading using Thread Libraries. Some programming languages as Java also provide libraries for managing threads that can be incorporated in to programs to enhance the performance of the programs.

In this assessment, you are required to demonstrate the use of threading and provide an evidence of performance boost of the program using the built-in threading libraries. The assessment is two-folds. Below are the requirements of the assessment:

**Part 1:**

You are required to write a program in C that takes (1,000,000,000) as an input (you are free to take this input from command line arguments or from the user). Once the input is taken, there should be 3 threads in your program performing the following operations:

1. Child Thread 1:
   1. Displays its ID
   2. Find incremental sum of all the numbers until and including 1,000,000,000
   3. Exit itself once completed the operation
2. Child Thread 2:
   1. Displays its ID
   2. Find incremental sum of all the even numbers until and including 1,000,000,000
   3. Exit itself once completed the operation
3. Parent / Main Thread:
   1. Displays its ID
   2. Creates Child Threads 1 and 2
   3. Merges Child Thread 1 and 2
   4. Displays the sum of all the numbers returned by Child Thread 1
   5. Displays the sum of all the numbers returned by Child Thread 2
   6. Calculates the following:
      1. sum of all the numbers returned by Child Thread 1 / sum of all the numbers returned by Child Thread 2
   7. Displays the result of above calculation

**Requirement 1:**

Analyze the time taken by the program to complete, e.g., review the usage of cores using Activity Monitor program in Ubuntu and find a way to monitor the time taken by the program to complete its execution. Furthermore, make sure, the calculations made by the program are correct and accurate. Do NOT forget to take a screenshot of the Activity Monitor.

**Part 2:**

You are required to write a program in C that takes (1,000,000,000) as an input (you are free to take this input from command line arguments or from the user). Once the input is taken, the program should perform the following operations:

1. Find incremental sum of all the numbers until and including 1,000,000,000
2. Find incremental sum of all the even numbers until and including 1,000,000,000
3. Display the sum of all the numbers until and including 1,000,000,000
4. Displays the sum of all the even numbers until and including 1,000,000,000
5. Calculates the following:
   * sum of all the numbers in Step 1 / sum of all the numbers in Step 2
6. Displays the result of above calculation

**Requirement 2:**

Analyze the time taken by the program to complete, e.g., review the usage of cores using Activity Monitor program in Ubuntu and find a way to monitor the time taken by the program to complete its execution. Do NOT forget to take a screenshot of the Activity Monitor.

**Requirement 3:**

Compare the results obtained by the analysis of Requirement 1 and Requirement 2. Make sure to write down the reasons of your analysis in plain English.

**Solution**