University of Wolverhampton
Faculty of Science and Engineering
Department of Mathematics and Computer Science

Module Assessment

Module	5CS021 – Numerical Methods and Concurrency
Module Leader	Hiran Patel
Semester	1
Year	2021/2022
Assessment	Portfolio
% of module mark	60%
Due Date	At the end of Week 12
Hand-in – what?	**Portfolio as specified in this document**
Hand-in- where?	Canvas
Pass mark	

Assessment overview

This portfolio is split up into 4 separate tasks which will test your knowledge of mathematics, C programming and concurrency. Each task will be zipped up into a single zip folder containing all C and resource files for the submission on Canvas.

TASK 1: Linear Regression (15% - 100 marks)

This task will test your knowledge of file input and mathematical formulas. Basic linear regression (LR) is used to find a relationship between two types of data. For example, you could use LR to find the relationship between glucose intake and heart rate. These initially would be plotted on a graph where the x axis represents glucose and the y axis represents heart rate. Using the LR formula, you can find y=bx+a (equation of a straight line (more commonly known as y=mx+c) between “n” number of points on a graph. Below are the formulas to find “a” and “b.”

$$A = \frac{(\Sigma y)(\Sigma x^2) - (\Sigma x)(\Sigma xy)}{n(\Sigma x^2) - (\Sigma x)^2}$$

$$B = \frac{n(\Sigma xy) - (\Sigma x)(\Sigma y)}{n(\Sigma x^2) - (\Sigma x)^2}$$

You will be given multiple text files containing “n” number of coordinates. Your program will read in the coordinates (x,y) and use the LR formula to produce the gradient (a) and the constant (b), and being able to print out y=bx+a. For example, if a = 0.5 and b=-2, your program will print out y=-2x+0.5. This equation now represents the trend in the data you will be given. Finally, your program will ask the user to type in a value of x which will then calculate y.

Read data from file appropriately (30 marks)

Use LR formula to process the data and print out the correct equation of a straight line (50 marks)

Find new value of “y” using user input (20 marks)

TASK 2: Calculating Pi using Leibniz formula and multithreading (15% - 100 marks)

The Leibniz formula is an infinite series method of calculating Pi. The formula is a very simple way of calculating Pi, however, it takes a large amount of iterations to produce a low precision value of Pi. This task requires a large amount of computation and therefore it is vital that you use multithreading to speed up the program. Below is the Leibniz formula:

$$\sum_{n=0}^{\infty} \frac{(-1)^n}{2n+1} = \frac{\pi}{4}.$$

As the series can be iterated infinite number of times, your program should allow the user to take in 2 inputs; the first is the number of iterations and the second is the number of threads the user would like to use. This means that the slicing of workload needs to be dynamic.

Calculating Pi using Leibniz formula (20 marks)

Using multithreading with appropriate slicing (60 marks)

Correct value of Pi printed out depending on iteration count (20 marks)

TASK 3: Finding prime numbers from multiple text files using multithreading (15% - 100 marks)

You will be given files containing a list of numbers. The amount and numbers themselves will be random. You will create a C program which counts the number of prime numbers there are within the files and output to a file the amount of prime numbers found, along with the prime numbers themselves. The aim of this task is to use POSIX threads to parallelise the task to take advantage of the multicore processor within your machine to speed up the task. The threads you spawn within the program must compute an equal or close to an equal amount of computations to make the program more efficient in relation to speed. For this section, you will be reading the data from three files and splitting it across many threads, you should load in the data from files and split the data into equal parts, then process each slice within your threads. This task also tests your knowledge of dynamic memory allocation. NOTE – this program should work with any amount of threads.

Creating an algorithm to detect prime numbers (10 marks)

Using dynamic memory – "malloc" (20 marks)

Using multithreading with equal computations (50 marks)

Outputting correct output to a file (20 marks)

TASK 4: Gaussian Blur with multithreading (15% - 100 marks)

Your program will decode a PNG file into an array and apply the gaussian blur filter. Blurring an image reduces noise by taking the average RGB values around a specific pixel and setting it's RGB to the mean values you've just calculated. This smoothens the colour across a matrix of pixels. For this assessment, you will use a 3x3 matrix. For example, if you have a 5x5 image such as the following (be aware that the coordinate values will depend on how you format your 2D array):

0,4	1,4	2,4	3,4	4,4
0,3	1,3	2,3	3,3	4,3
0,2	1,2	2,2	3,2	4,2
0,1	1,1	2,1	3,1	4,1
0,0	1,0	2,0	3,0	4,0

The shaded region above represents the pixel we want to blur, in this case, we are focusing on pixel 1,2 (x,y) (Centre of the matrix). to apply the blur for this pixel, you would sum all the **Red** values from the surrounding coordinates including 1,2 (total of 9 R values) and find the average (divide by 9). This is now the new Red value for coordinate 1,2. You must then repeat this for Green and Blue values. This must be repeated throughout the image. If you are working on a pixel which is not fully surrounded by pixels (8 pixels), you must take the average of however many neighbouring pixels there are.

Reading in an image file into a single or 2D array (10 marks)

Applying Gaussian filter on image (20 marks)

Using multithreading appropriately to apply Gaussian filter (40 marks)

Using dynamic memory – malloc (10 marks)

Outputting the correct image with Gaussian Blur applied (20 marks)

Important Message
You must achieve 40 percent overall to pass this module.

Submission of work

Your completed work for assignments must be handed in on or before the due date. ***You must keep a copy or backup of any assessed work that you submit. Failure to do so may result in your having to repeat that piece of work.***

Penalties for late submission of coursework
ANY late submission (without valid cause) will result in 0 marks being allocated to the coursework.

Cheating
Cheating is any attempt to gain unfair advantage by dishonest means and includes **plagiarism** and **collusion.** Cheating is a serious offence. You are advised to check the nature of each assessment. You must work individually unless it is a group assessment.
Cheating is defined as any attempt by a candidate to gain unfair advantage in an assessment by dishonest means, and includes e.g., all breaches of examination room rules, impersonating another candidate, falsifying data, and obtaining an examination paper in advance of its authorised release.

Plagiarism is defined as incorporating a significant amount of un-attributed direct quotation from, or un-attributed substantial paraphrasing of, the work of another.

Collusion occurs when two or more students collaborate to produce a piece of work to be submitted (in whole or part) for assessment and the work is presented as the work of one student alone.