COMP2212 Programming Language Concepts
Coursework III

Module Code:	COMP2212		
Module Title:	PROGRAMMING LANGUAGE CONCEPTS		
Module Leader:	Muhammad Imran Babar		
Assessment Type:	Individual Coursework	**Weighting:**	15%
Submission Due Date:	03/05/2024 5.00 PM (Malaysia time)		
Method of Submission:	Blackboard		

This assessment relates to the following Module Learning Outcomes:

A. Knowledge and Understanding	A1. A2. A3.
B. Subject Specific Intellectual and Research Skills	B1. B2. Understand the concept of functional programming and be able to write programs in this style. B3.
C. Subject Specific Practical Skills	C1. C2. C3.

Coursework Brief:

<u>INSTRUCTIONS</u>:

Here is the list of the assessments that you will solve in coursework III. Your solution for the given assessment is due by the given hand in date but may be submitted at any point before that.
Repeated multiple submissions are allowed before the deadline. You will be given feedback on how well your solutions perform against our test suit. Marks will only be given to your final submission after the deadline. The coursework III comprises 15% of the overall assessment for the module.

Marks: 100
Weightage: 15%

ASSESSMENT A1

a) What is Bisimulation, and how can it be applied to verify the equivalence of two processes? Write a C++ program demonstrating Bisimulation between two processes. **[Marks: 10]**

b) What is simulation in discrete event systems, and how can you represent it in C++? Write a basic simulation of a queue system where entities enter and exit over time. **[Marks: 10]**

c) How can Bisimulation be applied to verify the equivalence of two concurrent processes? Implement a C++ program with two concurrent processes and demonstrate their bisimilarity. **[Marks: 10]**

ASSESSMENT A2

a) What is the concept of concurrent processes, and how does it differ from sequential processes? Write a C++ program with two concurrent processes that demonstrate parallel execution. **[Marks: 10]**

b) Illustrate a common concurrency issue in C++ and propose a solution. **[Marks: 10]**

ASSESSMENT A3 MINI PROJECT ON SIMULATION

The objective of this simulation project is to design and implement a Traffic Management System using C++. This project aims to provide computer science students with hands-on experience in simulation programming, reinforcing their understanding of concepts such as functional design.

Project Description:

The Traffic Management System simulation will model the behaviour of vehicles on a road network, simulating traffic flow, traffic lights, and intersections. The primary goal is to create a realistic simulation environment where vehicles move through a city, obey traffic rules, and interact with traffic lights at intersections.

Key Features:

Vehicle Simulation: Implement a class for vehicles with attributes such as speed, position, and behaviour. Vehicles should follow traffic rules, including stopping at red lights and yielding to pedestrians. **[Marks: 3]**

Road Network: Design a road network with intersections and roads connecting them. Define the layout of the city, including one-way and two-way streets. **[Marks: 3]**

Traffic Lights: Simulate traffic lights at intersections. The traffic lights should cycle through red, yellow, and green phases, controlling the flow of traffic. **[Marks: 3]**

User Interface: Create a simple user interface to visualize the simulation. Display the road network, vehicles, and traffic lights in real-time. Allow users to interact with the simulation, such as adding or removing vehicles, changing traffic light timings, and pausing/resuming the simulation. **[Marks: 5]**

Simulation Time: Implement a time simulation mechanism to advance the simulation in discrete time steps. Each time step represents a unit of time in the simulation. **[Marks: 3]**

Statistics and Analysis: Collect and display statistics such as average vehicle speed, traffic density, and waiting times at intersections. Provide options for users to analyze and visualize these statistics. **[Marks: 3]**

Implementation Guidelines:

Functional Design: Use functional programming principles to model vehicle behaviour, road network, intersections, and traffic lights using pure functions. Emphasize immutability and avoid mutable state. **[Marks: 6]**

Data Structures: Choose appropriate immutable data structures to represent the road network, vehicle positions, and the state of traffic lights. Consider using persistent data structures for efficient updates. **[Marks: 6]**

Function Composition: Design functions that can be composed to achieve complex behaviours. Utilize higher-order functions to encapsulate common patterns in the simulation. **[Marks: 6]**

User Interface Design: Utilize functional programming principles to create a declarative and modular user interface. Separate UI components into pure functions. **[Marks: 6]**

Documentation: Provide comprehensive documentation for the functional codebase, including function signatures, descriptions, and examples of usage. **[Marks: 6]**

Deliverables:

Source Code: Submit the complete C++ source code for the Traffic Management System simulation project.

Documentation: Include a detailed document explaining the design, implementation, and usage of the simulation. Include any external libraries or dependencies used.

Demo Video: Create a demonstration video showcasing the functionality of the simulation. Explain how to run the simulation and interact with the user interface.

Any work submitted after the deadline's time will be subject to the standard University late penalties unless an extension has been granted, in writing by the Senior Tutor, in advance of the deadline. Details on the University's late penalties can be found here:

- https://www.southampton.ac.uk/~assets/doc/quality-handbook/Late%20Submission.pdf

COMP2212 PROGRAMMING LANGUAGE CONCEPTS
COURSEWORK III

	Unacceptable (25%)	Poor (50%)	Good (75%)	Excellent (100%)
Solution	An incomplete solution is implemented on the required platform. It does not compile and/or run.	A completed solution is implemented on the required platform, and uses the compiler specified. It runs but has logical errors.	A completed solution is tested and runs but does not meet all the specifications and/or work for all test data.	A completed solution runs without errors. It meets all the specifications and works for all test data.
Program Design	Few of the selected structures are appropriate. Program elements are not well designed.	Not all of the selected structures are appropriate. Some of the program elements are appropriately designed.	The program design generally uses appropriate structures. Program elements exhibit good design.	The program design uses appropriate structures. The overall program design is appropriate.
Clarity	Program contains no documentation, or grossly misleading indentation.	Program contains some documentation but has occasionally misleading indentation.	Program contains some documentation on major functions, variables, or non- trivial algorithms. Indentation and other formatting is appropriate.	Program contains appropriate documentation for all major functions, variables, or non-trivial algorithms. Formatting, indentation, and other white spaces aid readability.
Modularity	Program is one big function or is decompose in ways that make little sense.	Program is decomposed into units of appropriate size, but they lack coherence or reusability. Program contains unnecessary repetition.	Program is decomposed into coherent units but may still contain some unnecessary repetition.	Program is decomposed into coherent and reusable units, and there is no unnecessary repetition.
Completeness	Program shows little recognition of how different cases must be handled differently.	Program shows some evidence of case analysis, but may be missing significant cases or mistaken in how to handle some cases	Program shows evidence of case analysis that is mostly complete but may have missed minor or unusual cases.	Program shows evidence of excellent case analysis, and all possible cases are handled appropriately.
Project Design	Did not try to make own artwork. No clear purpose of the project or organization. Does not provide a way for other people to interact with program.	Project uses artwork of others with some effort to change. Has some sense of purpose and structure. Includes way for user to interact with program, may need to be clearer or fit program's purpose better.	Project uses original artwork. Has clear purpose, makes sense, has structure. Includes way for users to interact with program and clear instructions.	Project artwork and creativity significantly support the content. Has multiple layers or complex design. User interface fits content well, is complex; instructions are well written and integrated into design.