**Ticket Algorithm Implementation Rubric (due Monday 10.00am 3th February 2020)**

Implement a C++11 or C++14 efficient Ticketing Algorithm to schedule multiple concurrent processes that have a shared critical section. The Ticketing algorithm should support multiple threaded processes (at least 4) which execute concurrently. Your implementation utilise ***mutex*** to achieve mutual exclusion.

The critical section should be Player Vector2 Position and this can be updated or read by multiple systems such as Physics, Input Management and Render System.

**Marking Scheme**

|  |  |  |
| --- | --- | --- |
| **0 -35**  **(0.00 - 1.75)** | **35-75**  **(1.75 - 3.75)** | **75-100**  **(3.75 - 5.00)** |
| * Implementation will achieve minimum functionality * Implementation may contain some syntax and/or run-time errors * Implementation code will be poorly commented and/or formatted * Implementation will contain basic features; application will not be tested properly * Implementation code will not follow applicable coding conventions | * Implementation will achieve expected functionality * Implementation will not contain syntax and/or run-time errors * Implementation code will be reasonably commented and/or formatted * Implementation will contain assignment features * Implementation will be tested to a reasonable degree * Implementation code will follow appropriate coding conventions | * Implementation will achieve advanced functionality * Implementation will not contain syntax and/or run-time errors * Implementation code will be well commented and/or formatted * Implementation will contain assignment features * Application will be expertly tested * Implementation code will follow coding conventions * Implementation will identify and address any Thread Safety issues. |