|  |  |
| --- | --- |
| ****Subject Name and CRN:**** | INFT563 – Advanced Data Structures and Algorithms  CRN - 83111 |
| **Assessment Type:** | Project |
| **Due Date:** | August 13th, 11:59PM AEST |
| **Weight:** | 50% |
| **Marks:** | 100 |
| **Learning Outcomes:** | 1, 2, 3 & 4 |
| **Canvas:** | <https://aie.instructure.com/courses/188/assignments/550> |
| **Topic:** | Artificial Intelligence Project |

# Project Brief

You are to implement multiple artificial intelligence techniques within a graphical real-time application that runs a simulation with multiple A.I. controlled entities, demonstrating decision making, steering behaviours and pathfinding techniques.

# Project Features

Your simulation must include multiple entities that use decision making techniques. The choice of technique is up to you, with marks being awarded based on the technical difficulty of the technique chosen, and on the quantity of techniques. For example, implementing entities that use two different techniques will be marked higher than all entities using the same technique.

The decision making technique must make use of data structures and algorithms created for that specific purpose. For example, implementing a Finite State Machine using a switch statement is not acceptable, but a Finite State Machine implemented using State classes that Transition, is.

Your entities must be visibly drawn where appropriate.

Your entities must also demonstrate the use of Steering Behaviours, in addition to Decision Making. Using multiple steering behaviours working in unison is worth more marks than simply having one behaviour active at a time.

Finally, your simulation must also include entities using Pathfinding techniques to navigate, in addition to using Steering Behaviours and Decision Making techniques.

To summarise, your simulation must:

* Be real-time, with graphically drawn entities
* Entities must:
  + All must use Decision Making techniques, such as Finite State Machines, Decision Trees, Behaviour Trees and Planners
  + Some or all must use multiple Steering Behaviours at the same time
  + Some or all must make use of Pathfinding, which may include Navigational Meshes

# Submission Process

All submissions must be made by the due date and time listed.

Students submitting an assessment after the due date and **without** an approved extension of time will lose 10% of the maximum available mark on the assessment per day it is late. In addition, assessments submitted late will only be awarded a maximum of a **pass** grade. This means that an assessment submitted 2 days late can achieve a mark of 80%, which contributes to the subject’s overall mark, but the assessment item itself will only receive a pass mark.

You are to submit a single zipped file via the Canvas link, or via an alternative approved method, that contains:

* A folder containing an executable run-time of your application that can be run external to any IDE. This folder must not contain more than is required for executing the runtime and must contain all required resources and assets.
* A folder containing the project source, **excluding temporary build files**, which can be successfully compiled by an assessor, without errors.

# Grading Rubric

Your assessment will be graded based on the grading rubric below and feedback will be given to the student as required.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Criteria** | **Mark** | **High Distinction** | **Distinction** | **Credit** | **Pass** | **Fail** |
| **Decision Making Techniques** | 30 | 3 or more well implemented Decision Making techniques, such as a complex implementation of a Finite State Machine, Behaviour Tree or Planner, used on multiple entities. (**30-25.5**) | 2 or more well implemented Decision Making techniques, such as a complex implementation of a Finite State Machine, Behaviour Tree or Planner, used on multiple entities. (**25.5-22.5**) | Well implemented Decision Making technique, such as a complex implementation of a Finite State Machine, Behaviour Tree or Planner, used on multiple entities. (**22.5-19.5**) | Well implemented simple Decision Making technique, such as a basic Finite State Machine, Decision Tree or Behaviour Tree, used on multiple entities. (**19.5-15**) | Decision making techniques do not meet minimum requirements. (**15-0**) |
| **Pathfinding Techniques** | 30 | Navigational Mesh based pathfinding used that uses path smoothing, and allows for custom heuristics, implemented on multiple entities. (**30-25.5**) | A complex and highly optimised pathfinding implementation used on multiple entities using the same graph, that allows for custom heuristics. (**25.5-22.5**) | Simple A\* implementation used for multiple entities using the same graph. (**22.5-19.5**) | Simple A\* implementation used on a single entity. (**19.5-15**) | Pathfinding techniques do not meet minimum requirements. (**15-0**) |
| **Steering Behaviours** | 20 | Multiple steering behaviours implemented using some form of automatic weighted combination, including Collision Avoidance against for more than 2 shapes, used on multiple entities. (**20-17**) | More than 4 Steering Behaviours implemented using some form of automatic weighted combination, demonstrated on multiple entities. (**17-15**) | 2 to 4 Steering Behaviours implemented using some form of automatic weighted combination, demonstrated on multiple entities. (**15-13**) | 2 Steering Behaviours implemented on multiple entities, such as Seek and Wander. (**13-10**) | Steering Behaviour techniques do not meet minimum requirements. (**10-0**) |
| **Completeness, Code Documentation and Coding Standards** | 20 | Code is fully commented when explanation is needed for a section of code. Function declarations include comments that describe parameters and return values. Code adheres to a coherent standard throughout the project. (**20-17**) | Code is commented when explanation is needed for non-obvious sections. Function declarations include comments that describe parameters and return values. Code adheres to a coherent standard through majority of the project. (**17-15**) | Code has few comments for obscure sections of code. Function declarations include comments that explain functionality. Code mostly adheres to a standard. Some external documentation provided. (**15-13**) | Code has few comments for obscure code sections. Function declarations include comments describing functionality. Code is neat and readable. (**13-10**) | Code is poorly written and follows no coding standard. Little to no commenting used. Obscure code has no comments describing functionality. (**10-0**) |