Games AI

Lecture 10.1

Discussion

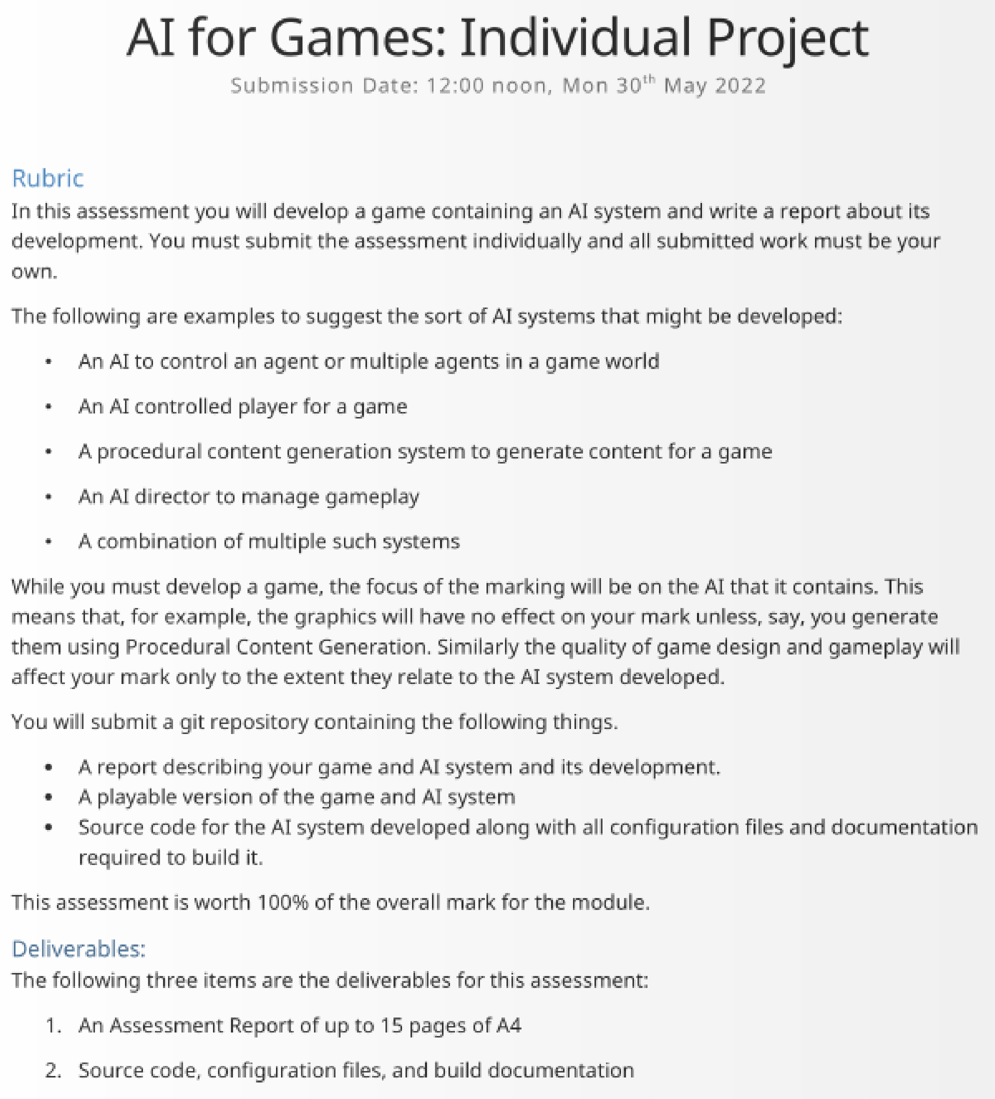
Overview

- Overview
  - Writing the report
  - The Git Repository

Writing the Report

Writing the report

- 15 Pages A4 maximum
  - Do not write 15 pages of just text
  - Space for illustrations / screenshots / graphs / diagrams



AI for Games: Individual Project

Submission Date: 12:00 noon, Mon 30th May 2022

Rubric

In this assessment you will develop a game containing an AI system and write a report about its development. You must submit the assessment individually and all submitted work must be your own.

The following are examples to suggest the sort of AI systems that might be developed:

- An AI to control an agent or multiple agents in a game world
- An AI controlled player for a game
- A procedural content generation system to generate content for a game
- An AI director to manage gameplay
- A combination of multiple such systems

While you must develop a game, the focus of the marking will be on the AI that it contains. This means that, for example, the graphics will have no effect on your mark unless, say, you generate them using Procedural Content Generation. Similarly the quality of game design and gameplay will affect your mark only to the extent they relate to the AI system developed.

You will submit a git repository containing the following things.

- A report describing your game and AI system and its development.
- A playable version of the game and AI system
- Source code for the AI system developed along with all configuration files and documentation required to build it.

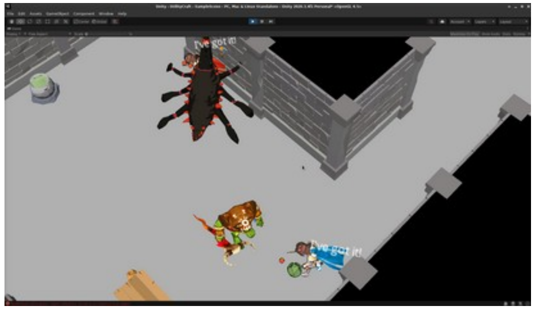
This assessment is worth 100% of the overall mark for the module.

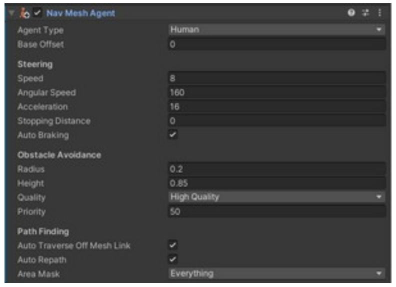
Deliverables:

The following three items are the deliverables for this assessment:

1. An Assessment Report of up to 15 pages of A4
2. Source code, configuration files, and build documentation

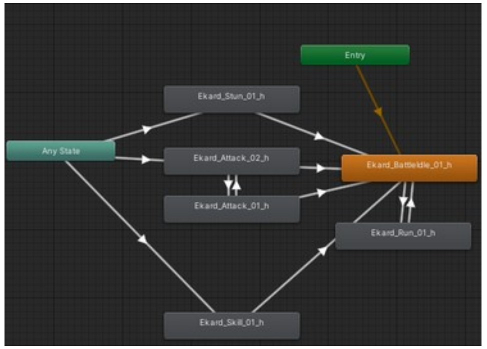
- Screenshots to illustrate, for example:
  - The game
  - The AI in the game
  - Specific behaviours (series of screenshots/overlaid diagram)
  - Specific contexts/levels/scenarios (e.g. if it glitches on certain kinds of level geometry)
  - Test scenarios
  - Examples of content generated
  - Editing/content creation tools (if any). How it looks in the Unity inspector.





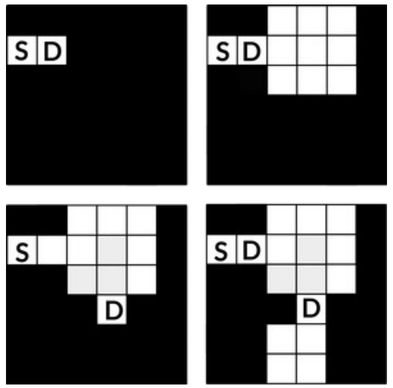
Nav Mesh Agent

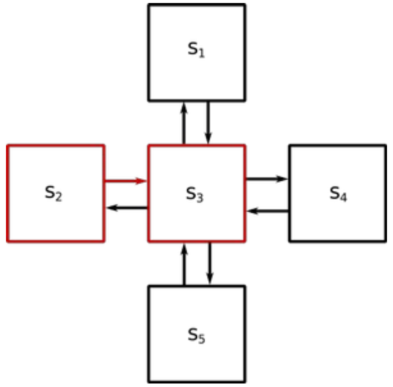
Agent Type	Human
Base Offset	0
Steering	
Speed	8
Angular Speed	160
Acceleration	16
Stopping Distance	0
Auto Braking	✓
Obstacle Avoidance	
Radius	0.2
Height	0.85
Quality	High Quality
Priority	50
Path Finding	
Auto Traverse Off Mesh Link	✓
Auto Repath	✓
Area Mask	Everything



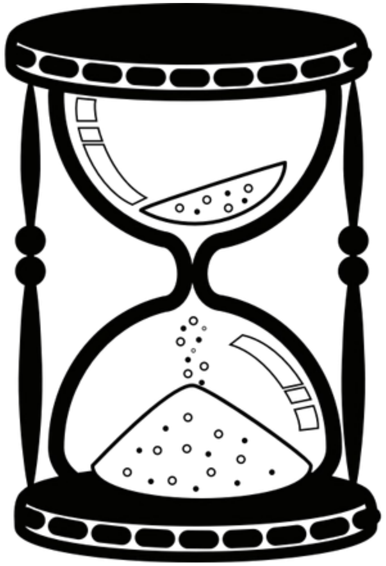
Writing the report

- Diagrams to illustrate, for example:
  - Finite State Machines
  - Behaviour Trees
  - Utility AI considerations
  - Level generation algorithm



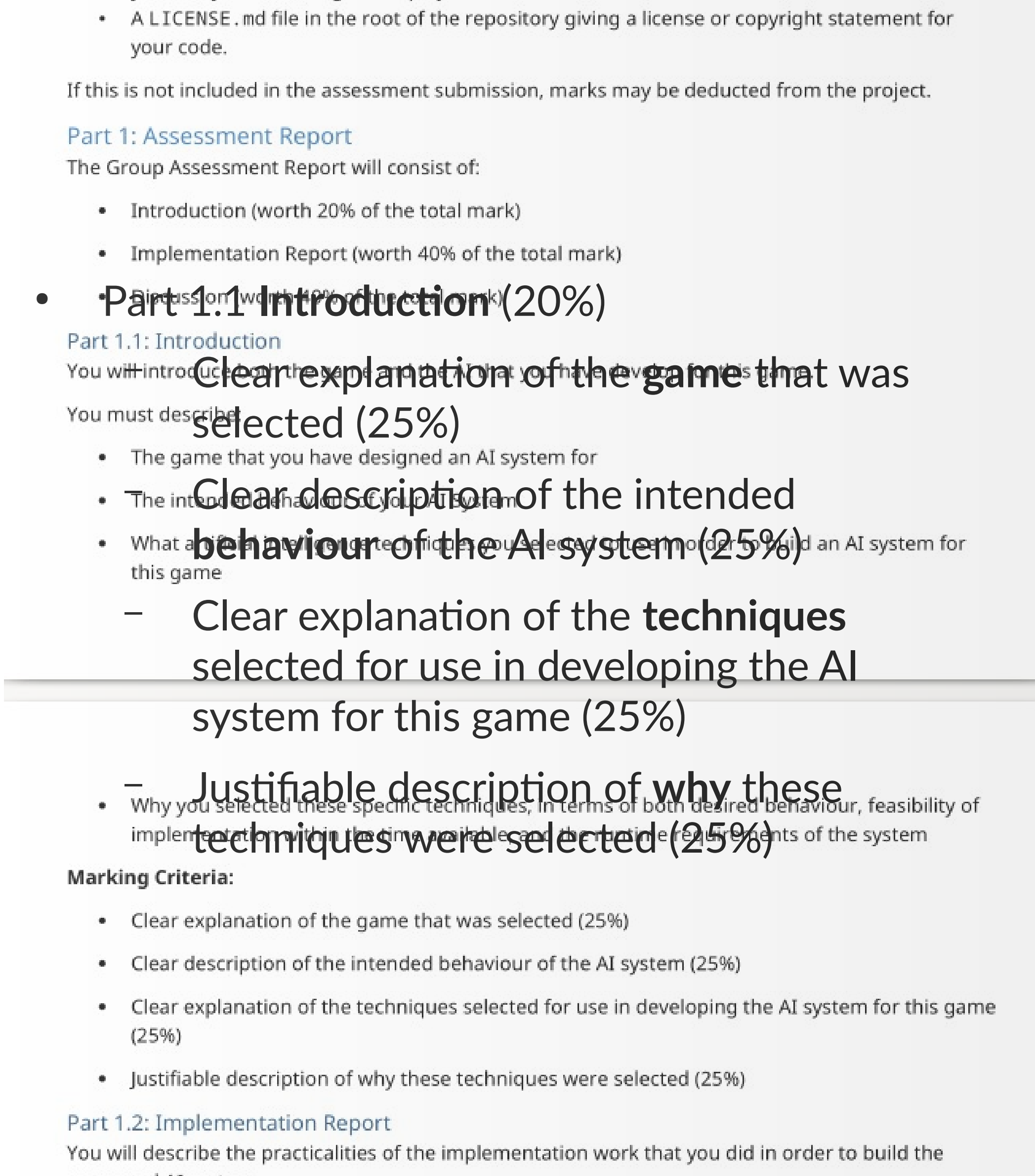


- Structure
  - Part 1.1 Introduction (20%)
  - Part 1.2 Implementation Report (40%)
  - Part 1.3 Discussion (40%)



Writing the report

- Part 1.1 **Introduction** (20%)
  - Clear explanation of the **game** that was selected (25%)
  - Clear description of the intended **behaviour** of the AI system (25%)
  - Clear explanation of the **techniques** selected for use in developing the AI system for this game (25%)
  - Justifiable description of **why** these techniques were selected (25%)



- A `LICENSE.md` file in the root of the repository giving a license or copyright statement for your code.

If this is not included in the assessment submission, marks may be deducted from the project.

Part 1: Assessment Report

The Group Assessment Report will consist of:

- Introduction (worth 20% of the total mark)
- Implementation Report (worth 40% of the total mark)
- Discussion (worth 40% of the total mark)

Part 1.1: Introduction

You will introduce both the game and the AI that you have develop for this game.

You must describe:

- The game that you have designed an AI system for
- The intended behaviour of your AI system
- What artificial intelligence techniques you selected to use in order to build an AI system for this game
- Why you selected these specific techniques, in terms of both desired behaviour, feasibility of implementation within the time available and the runtime requirements of the system

Marking Criteria:

- Clear explanation of the game that was selected (25%)
- Clear description of the intended behaviour of the AI system (25%)
- Clear explanation of the techniques selected for use in developing the AI system for this game (25%)
- Justifiable description of why these techniques were selected (25%)

Part 1.2: Implementation Report

You will describe the practicalities of the implementation work that you did in order to build the

- Clear explanation of the game that was selected
  - You’re designing an AI system **for a game**
    - What is the game (intended to be) like?
    - It’s fine if you haven’t implemented everything!
  - This will let you talk about your AI **in context**
    - Lets you justify your decisions for your game
    - What AI is appropriate depends on the game

Writing the report

- Clear description of the intended behaviour of the AI system
  - The **goals** you have for your AI System
    - **These should relate to the game**
    - It's fine not to achieve them all!
  - Think about
    - What would the e.g. agents do? (Anything it should avoid doing?)
    - How would this impact the game and the player?
    - Does your game put any special requirements on the AI?
      - performs well (mobile?)
      - is customisable/reusable (lots of content creation?)
      - is controllable (authorial intent?)
      - is unpredictable (replay value?)

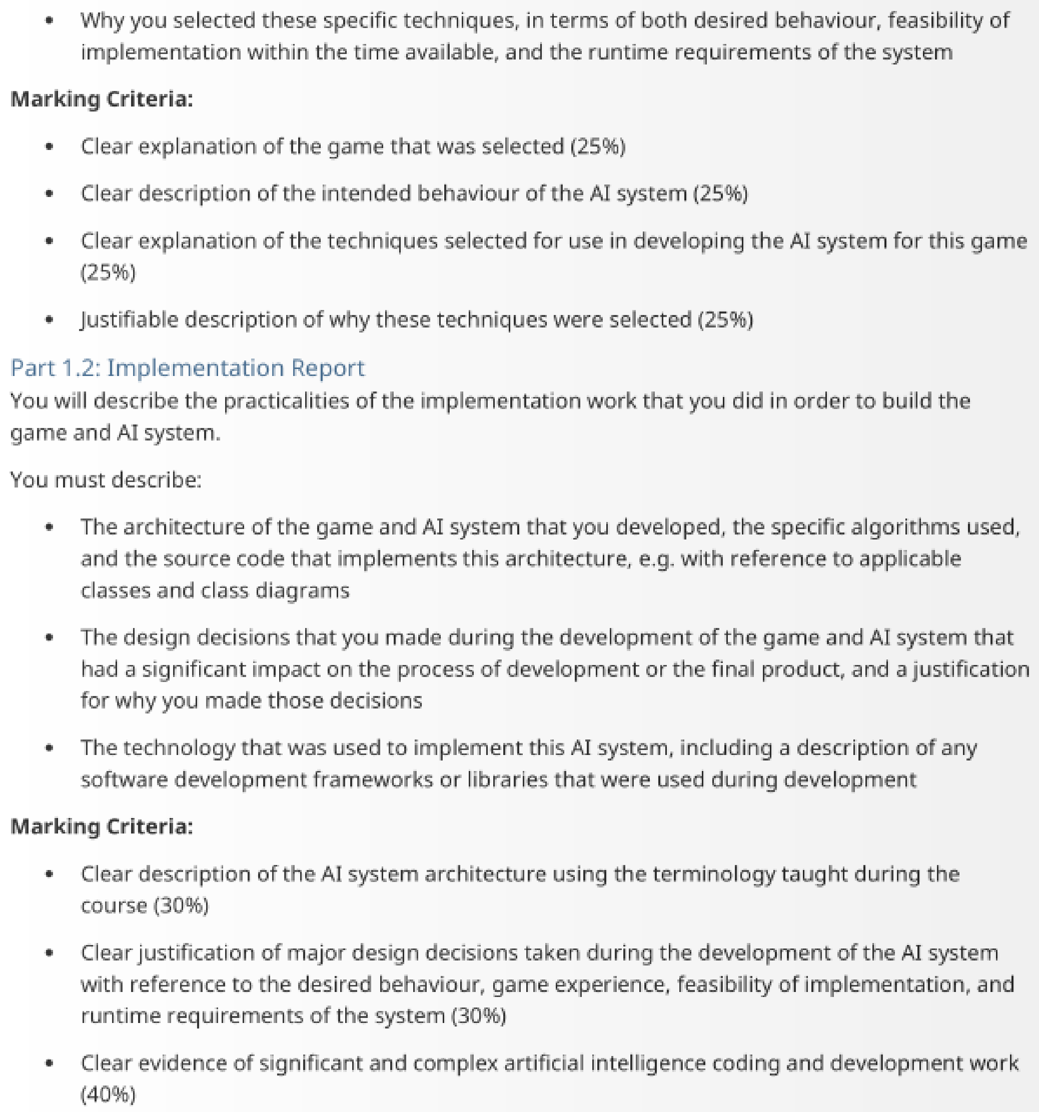
Writing the report

- Clear explanation of the techniques selected for use in developing the AI system for this game
  - What **techniques** did you use?
    - Introduce them in general terms (save implementation details for later)
    - Show that you understand the principles/theory involved
  - Think of this a bit like a Literature Review - introduce us to the concepts we will need **to understand what you have done or the decisions you've made**
    - Refer back to this when talking about implementation and design decisions

- Justifiable description of why these techniques were selected
  - Why did you use these techniques?
    - Why were these suitable for achiving your goals?
    - Make sure to contextualise it in your (intended) game
  - Were there any other factors influencing their selection?
    - Implementation time/effort? (What is it about these techniques that makes them easier to implement?)
    - Technical/library/game engine limitations?
    - Build requirements? (available compute to train)
    - Runtime requirements? (memory/CPU footprint)

Writing the report

- Part 1.2 **Implementation Report** (40%)
  - Clear description of the **AI system architecture** using the terminology taught during the course (30%)
  - Clear justification of **major design decisions** taken during the development of the AI system with reference to the desired behaviour, game experience feasibility of implementation, and runtime requirements of the system (30%)
  - Clear evidence of **significant and complex artificial intelligence** coding an development work (40%)



- Why you selected these specific techniques, in terms of both desired behaviour, feasibility of implementation within the time available, and the runtime requirements of the system

Marking Criteria:

- Clear explanation of the game that was selected (25%)
- Clear description of the intended behaviour of the AI system (25%)
- Clear explanation of the techniques selected for use in developing the AI system for this game (25%)
- Justifiable description of why these techniques were selected (25%)

Part 1.2: Implementation Report

You will describe the practicalities of the implementation work that you did in order to build the game and AI system.

You must describe:

- The architecture of the game and AI system that you developed, the specific algorithms used, and the source code that implements this architecture, e.g. with reference to applicable classes and class diagrams
- The design decisions that you made during the development of the game and AI system that had a significant impact on the process of development or the final product, and a justification for why you made those decisions
- The technology that was used to implement this AI system, including a description of any software development frameworks or libraries that were used during development

Marking Criteria:

- Clear description of the AI system architecture using the terminology taught during the course (30%)
- Clear justification of major design decisions taken during the development of the AI system with reference to the desired behaviour, game experience, feasibility of implementation, and runtime requirements of the system (30%)
- Clear evidence of significant and complex artificial intelligence coding and development work (40%)

Writing the report

- Clear description of the AI system architecture using the terminology taught during the course
  - Imagine you’ve hired a programmer. Explain everything they need to know to keep developing your AI
    - Describe the implementation of techniques introduced earlier
    - This will be technical
    - Use technical terminology (states, response curves, fitness landscapes, classes, interfaces, delegates, …)
    - Use diagrams/flowcharts/tables - a picture is worth 1000 words
  - Show you understand any third-party AI (e.g. Unity NavMeshAgent), but don’t claim credit
  - Describe the game code to the extent that it’s relevant to the AI
    - How does the AI get it’s data?
    - What behaviours are there? How do they work?

Writing the report

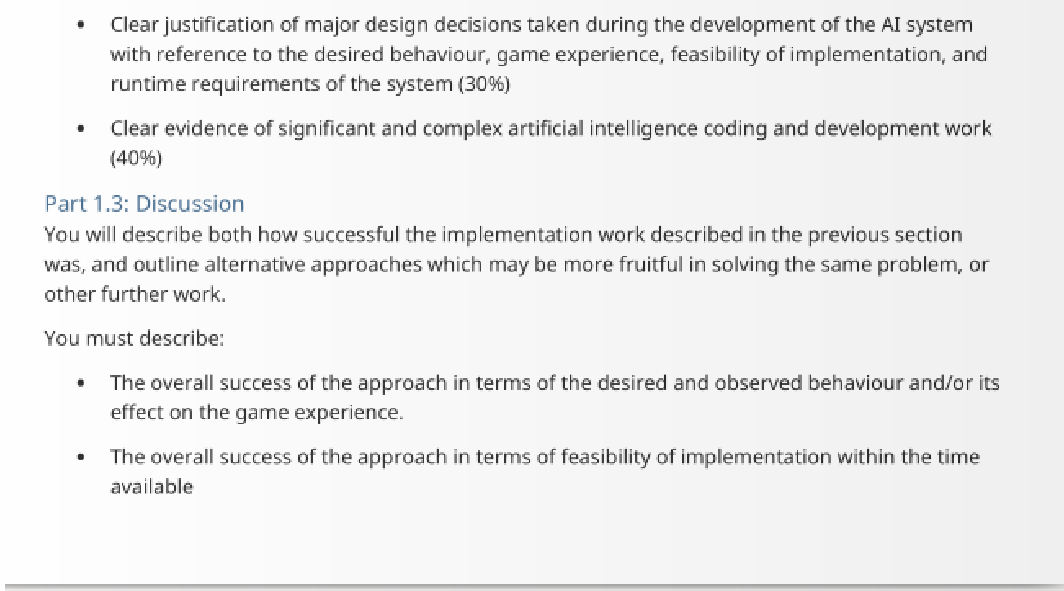
- Clear justification of major design decisions taken during the development of the AI system with reference to the desired behaviour, game experience feasibility of implementation, and runtime requirements of the system
  - This implementation was the result of your decisions - what were they and why did you make them?
    - Was technique X not working? (Why?)
    - Did changing to Y improve something (What?)
    - Did your timeline change? (How? How did your implementation plan change?)
    - Did you compare A and B in some metric? (What? Why? Which was better?)
  - Here you can technically justify your implementation
    - Was X the most performant implementation?
    - Did Y minimise garbage collection / draw calls / cache misses?
    - Was Z done to follow best coding practices / achieve modular AI?

Writing the report

- Clear evidence of significant and complex artificial intelligence coding and development work
  - Do something technically sophisticated
  - Describe it in a way that shows off its technical sophistication
    - I’m going to assume you’ve told me everything worth marks in the report
      - Make sure I don’t overlook something
      - I won’t go digging through your code to find it - it must be evidenced!
  - Use UML diagrams/code snippets/psuedocode to describe your code

Writing the report

- Part 1.3 **Discussion** (40%)
  - Clear understanding how the selected techniques and the implementation effected the behaviour of the system and a judgment of how successful this is at achieving the desired behaviour and/or desired game experience (50%)
  - Clear understanding of how the approach taken influenced the feasibility of implementation and the runtime requirements of the system (25%)
  - Clear understanding of alternative approaches that could be taken to solving the same problem and discussion of the pros and cons of these techniques (25%)



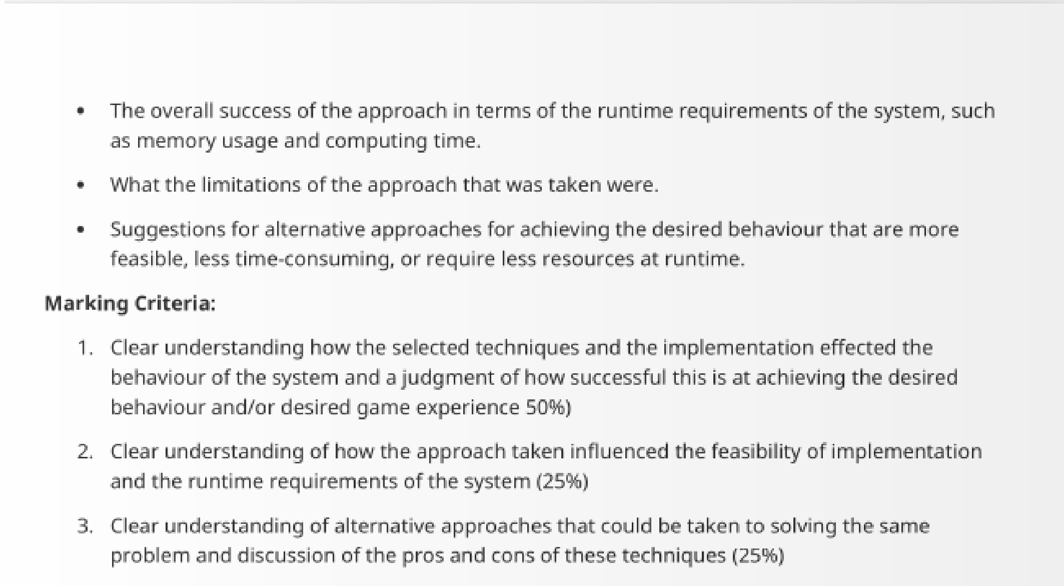
- Clear justification of major design decisions taken during the development of the AI system with reference to the desired behaviour, game experience, feasibility of implementation, and runtime requirements of the system (30%)
- Clear evidence of significant and complex artificial intelligence coding and development work (40%)

Part 1.3: Discussion

You will describe both how successful the implementation work described in the previous section was, and outline alternative approaches which may be more fruitful in solving the same problem, or other further work.

You must describe:

- The overall success of the approach in terms of the desired and observed behaviour and/or its effect on the game experience.
- The overall success of the approach in terms of feasibility of implementation within the time available



- The overall success of the approach in terms of the runtime requirements of the system, such as memory usage and computing time.
- What the limitations of the approach that was taken were.
- Suggestions for alternative approaches for achieving the desired behaviour that are more feasible, less time-consuming, or require less resources at runtime.

Marking Criteria:

1. Clear understanding how the selected techniques and the implementation effected the behaviour of the system and a judgment of how successful this is at achieving the desired behaviour and/or desired game experience 50%)
2. Clear understanding of how the approach taken influenced the feasibility of implementation and the runtime requirements of the system (25%)
3. Clear understanding of alternative approaches that could be taken to solving the same problem and discussion of the pros and cons of these techniques (25%)

Writing the report

- Clear understanding how the selected techniques and the implemetation effected the behaviour of the system and a judgment of how successful this is at achieving the desired behaviour and/or desired game experience
  - Describe how what you've implemented actually works
    - Clearly and objectively describe behaviour
    - Consider different contexts (e.g many/few opponents; levels size; level geometry) particularly those relevant to your game (intention)
    - Give examples/screenshots/diagrams
  - Evaluate it
    - Call back to your goals
    - Be honest - your might have tried something that failed, that's okay.
  - Quantify, if possible, in ways meaningful to your goals
    - Win rate? FPS? Average play time? Performance on mobile? User feedback?

Writing the report

- Clear understanding of how the approach taken influenced the feasibility of implementation and the runtime requirements of the system
  - A chance to talk about practicalities
    - Did it take longer to develop than you expected? Why?
    - Was the goal too big/small? Was the approach taken suited to implementing this within the time constraints?
    - Was it more resource-intensive than hoped? Why?
    - Did it take too long to train?
    - Did the game engine/programming language make your approach/technique more difficult? (e.g. was Unity suited to the style of game/AI?)

Writing the report

- Clear understanding of alternative approaches that could be taken to solving the same problem and discussion of the pros and cons of these techniques
  - You should have already discusssed specific technical decisions (e.g. why A* instead of BFS). The section should have a broader perspective.
  - Look further afield, show you understand the wider AI landscape
    - Could Machine Learning (e.g. neural networks) have been used?
    - What about whole-game search?
    - Could heriarchical state machines / utility AI / behaviour trees have been used instead?
    - Could content have been hand authored instead of generated or vice versa?
    - Could automated game testing have been used?
    - Could evolutionary algorithms have been used?
  - Give me an idea of how these could potentially be used. Give pros and cons.

The Git Repository

The Git Respository

- Everything goes in your git repository
  - Report
  - Playable Build
  - Source Code
  - Readme
  - License
- You will just submit **the URL** of your git repository via Moodle

- Playable Build
  - Your submission should include a playable game with your AI system included.
    - If the game is not submitted – or cannot be run with reasonable effort – then your mark will be adjusted.
    - Remember the CD/102 computers are not very powerful, and they run Linux
    - I run Linux too, build for Linux
  - Instructions to run and play in README.md

- Source Code
  - All the source and assets I need to build the playable build you submitted
  - Exclude unnecessary files (e.g. cached files) with a .gitignore
  - The root of your repository will be the root of your project folder in e.g. Unity

- README.md
  - Basic information I need to know to build and play your game
- What is a readme?
  - Often the first file someone encounters
  - Gives a detailed description of your project
  - Documentation how to use it
    - How to get it
    - How to run it
    - How to develop it

Git Repository

- Project Title
  - A descriptive name, or the name of the game
- Description
  - What it does
  - What core technologies it uses
  - What key challenges does it solve
  - Important areas where it is incomplete

Git Repository

- How to install and run the project
  - Software and hardware dependances
  - Steps required to build / install / run the project
  - Assume a technically competent reader
    - Standard tools don’t need to be described
      - “Navigate to the root of the repository and run `npm install`”
    - Console commands are fine
    - Anything that can be easily found by a web search

- How to use the project
  - Brief introductory information to how to use the project, e.g.
    - Necessary initial configuration required
    - Game controls
    - Description of game goal, rules, if not obvious
  - Assume a technically competent user
    - This is not the place for detailed instructions about specific tasks – use a tutorial or in-depth documentation for that

- How to develop the project
  - Information about how to continue development
    - Any development dependencies
      - Software I need to compile from source
      - e.g. Unity
    - How to compile
    - How to contribute (if open source)
    - How to run tests

- Credits
  - If you’ve included third-party code or tools under license
  - Assets with a creative commons attribution license
  - If you’ve adapted an existing project

Git Repository

- After reading your README.md I should be able to build and run your game and be able to understand how to play
- Keep it brief and to the point
  - It’s not your job to teach anything
  - You’re making the steps unambiguous for a technically competent user
- Use Markdown (https://www.markdownguide.org/basic-syntax/)
  - A standard way of formatting plain text files

- LICENSE.md
  - What are people allowed to do with your code?
  - Use a standard software license or copyright statement

    e.g. “Copyright (c) 2022 Joe Blogs”
  - https://github.com/licenses/license-templates/tree/master/templates

Copyright (c) {{ year }} {{ organization }}

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Coming up

- Coming up:
  - **Git**