



COS30008 Data Structures and Patterns: Programming Project 1 (25%)

Due Date: Week 10

Instructions

1. The aim of this assessment is to examine your understanding of Data Structures and Patterns in both design and implementation on an actual application.
2. This assessment consists of a **Software Prototype** (10%), a **Project Report** (10%), and a **Video Demonstration** (5%).
3. A list of Data Structures and Patterns concepts is supplied (covered during Week 1-9 Modules). You are required to develop a prototype software whose implementation demonstrates all the required concepts.
4. **Software Prototype Application:** An open **Project Theme** will be given, so you are free to develop your own version of the prototype, under these conditions:
 - a. The title and content of your application must be specific and unique, else you will infringe upon the work of other students.
 - b. The program must be developed as a C++ Console Application.
 - c. The creation, use and management of Entities/Objects are manually built (not relying on ready-made structures or libraries).
 - d. Provide comments in your code to identify and explain each implemented concept.
 - e. The prototype must be at least minimally functional (complete runtime cycles and carry out functions that demonstrate real-time operations of demonstrated concepts).
 - f. Code the application fully by yourself to avoid plagiarism.
 - g. The full application project folder is to be zipped and submitted in a separate Assignment Page. There is a dedicated Assignment Page for submitting the Report.

5. **Project Report:** The submission must include a Report (in PDF format) with the following content:

- a. Assignment Cover Page (Find this in the Report Submission Page).
- b. Table of Contents
- c. Link to Video Demonstration
- d. Introduction and Description of your Software Prototype Application (Include screen captures and diagrams where necessary).
- e. Write a **Report Section for each Concept**, containing:
 - i. **Application:** What area will you apply this concept to?
e.g. Inventory, fight sequence, loading maps, etc.

What is the desired operation? Describe in detail.

e.g. Each cell that makes up the map must allow selection of the next cell in the UP, DOWN, LEFT and RIGHT position...

ii. **Concept:**

- How does it match this application's desired operation?
- Are there any other alternative Structures can be used here? Why are they less preferred?
- **Draw a diagram to represent how this Structure/Concept is used in this part of the program.**

iii. **Implementation & Output:**

- Include a screen-capture of this code implementation. Ensure that your code has comments to show how it works, step-by-step.
- If the subsystem does not have an output, add temporary testing code to show that it works properly via Command Prompt output.

iv. **Troubleshooting Summary:**

- Did you run into any issues while implementing this concept?
- Did you refer to any resources (tutorials/guides) to find the solution?
- How did you solve the problem?
- **Did you use GenAI to help you figure out how to complete this concept? What prompts did you use? Include Snapshots of key responses from the GPT.**
- Cite your resources/references

f. Appendix: **Paste the full text content of all source files (only from .h and .cpp). Raw code only, no screencaps allowed.**

6. **Video Demonstration:** A 15–45-minute presentation video is required to prove that your developed system is complete and working properly. Ensure you cover the following:

- a. Introduce your program and how to use/play it.
- b. Live Demonstration of all its subsystems working. Indicate which Concept/Structure is involved with each subsystem.
- c. Upload this as an **Unlisted YouTube Video** and add the Link to the front of the Report.

7. Deliver this assignment in **both Canvas Submissions**:

- a. **Report Submission Page:** Report in .PDF format (including link to Presentation Video).
- b. **Prototype Submission Page:** Zipped Folder containing full VS project.

Examined Concepts

Read the Entire List before starting work. Your project must demonstrate and explain the application of the following 10 concepts (each must have a dedicated demonstration):

1. Object-Oriented Programming:
 - a. **Inheritance and Derived Classes**
 - b. **Polymorphism**
2. Composite Data Structures:
 - a. **Hash Tables** (Independent Research)
 - b. **Singly Linked-List**
 - c. **Doubly Linked -List**
3. Abstract Data Type:
 - a. **Stack**
 - b. **Queue**
 - c. **Tree**
4. Design Patterns
 - a. **Iterator**
 - b. Research and choose **one other Design Pattern** appropriate for your application.

Marking Rubric

For each Concept	2	1.5	1.0	0.5	0
4 Requirements: <ul style="list-style-type: none"> a) The purpose and operation of the target subsystem are described clearly. b) An appropriate Concept/Structure is chosen, based on the intended operation of the target subsystem. A suitable diagram is supplied to illustrate how this Concept/Structure is applied to the subsystem. c) Code implementations are well commented and/or explained clearly. Supplied screen-captures of Console Output shows proof that the implemented concept is working. d) Troubleshooting summary indicates self-reflection and/or include citations for crediting reference sources. 	Satisfy 4 Req. or eq.	Satisfy 3 Req. or eq.	Satisfy 2 Req. or eq.	Satisfy 1 Req. or eq.	No Attempt.

Score for Presentation Video	5	4	3	2	1	0
5 Requirements: <ul style="list-style-type: none"> a) The background and design of the program is explained clearly. b) Live demonstration to show that the program is fully functional. c) The various functions of the program are demonstrated, highlighting which concepts were involved in their implementation. d) Discussed reflection over Data Structures awareness in the development process, and areas for future improvement. e) Meet all Genre requirements as stated in the Project Theme. 	Satisfy 5 Req. or eq.	Satisfy 4 Req. or eq.	Satisfy 3 Req. or eq.	Satisfy 2 Req. or eq.	Satisfy 1 Req. or eq.	No Attempt.

Policy on late submission

Late submissions will be penalized with a penalty @ 10% per day and will be marked zero if the submission is done after 5 days. Only in some special cases such as medical emergency, late submission or extension will be provided if a medical certificate is presented.

Policy on plagiarism

Before you submit the document, it is your responsibility to ensure that you follow the University plagiarism policy (click on the link below to read about plagiarism in detail) and do not breach it.

Plagiarism is the practice of submitting or presenting the ideas, writing or other work of someone else, in whole or in part, as though it is your own work. That is, without proper acknowledgement of the source(s). Paraphrasing another person's work without attribution is also plagiarism.

<https://www.swinburne.edu.au/current-students/manage-course/exams-results-assessment/plagiarism-academic-integrity/plagiarism-misconduct>

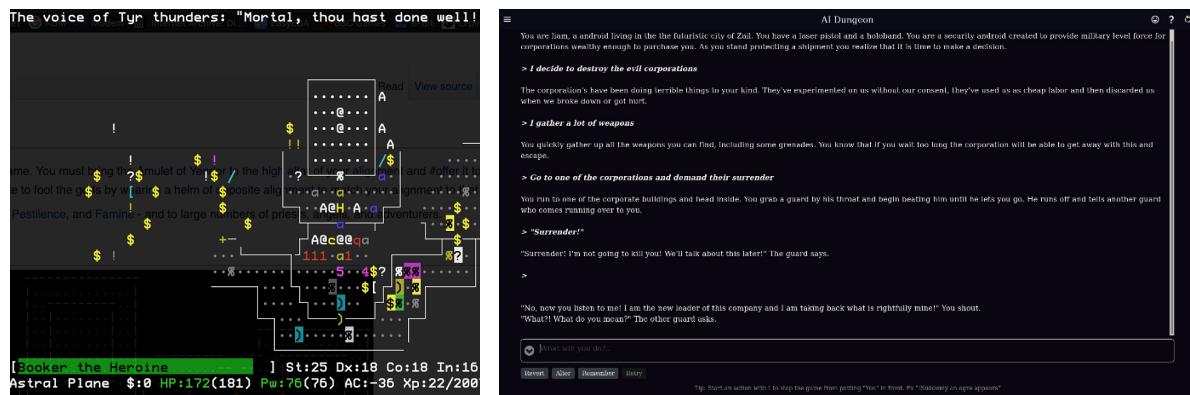
Project Theme: Procedural Storytelling

Procedural Storytelling is a relatively new term to describe a classic gameplay mechanic known as Game Masters. Since the late 90's, tabletop roleplaying games like Dungeons & Dragons, Shadowrun, and Vampire: The Masquerade relied on a human taking the role of Game Master, crafting the scenario, arranging encounters and facilitating interactions between other human players, based on the tools, classes, history and settings available in a particular franchise. Lately, PC games have automated this across various titles like RimWorld, The Sims, FTL: Faster Than Light, Project Zomboid, Shadow of Mordor, and Aliens: Isolation. In these titles, an AI Storyteller operates behind the scenes, spawning crisis, monster attacks, drama and all kinds of challenges depending on your current progress. The aim is to create a dynamic gameplay experience which changes according to your pace and is different each time you start a new game.



AI Storytellers play the role of [Gamemasters](#) across a wide variety of genres, including Colony Builders (like RimWorld, Dwarf Fortress, Frostpunk), Grand Strategy (like Europa Universalis, Total War series), Roguelike & Survival (like Don't Starve, FTL), and many Simulators (like The Sims).

Some notable examples of Text-Based Games equipped with AI Storytellers include [AI Dungeon](#), [Dwarf Fortress](#), and [NetHack](#).



You are tasked to build any genre of game in C++ you wish (e.g. adventure, simulator, colony builder, shooter, RPG, etc.) which integrates some form of AI Storyteller or Gamemaster, as long as you implement the 10 listed concepts as appropriate program features or game mechanics. It is recommended for you to build a Text-Based game, but you are free to implement 2D or 3D platforms if you know how to. Your game must meet the following requirements:

- AI Storyteller must influence at least 3 aspects of the game (e.g. attacks, drops, scenarios, events, etc.) based on at least 2 criteria (e.g. timer, health, progress, e.g.).
- Use an Audio Library ([SFML](#), DirectX, Vulcan etc.) to utilize Music and/or Sound effects.
- File operations to keep track of Player Progress, Save/Load Game or etc.