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

Due Date: Week ~~8~~ 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. *For* **each Concept**, have a section 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?
            - Cite your resources/references

    f. Appendix: **Paste the full text content of all source files (only from .h and .cpp).**

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. **Array**
    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: 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	5	4	3	2	1	0
5 Requirements: a) The background and design of the program is explained clearly.	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.

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) The quality and presentation of the video is concise and creative.						

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

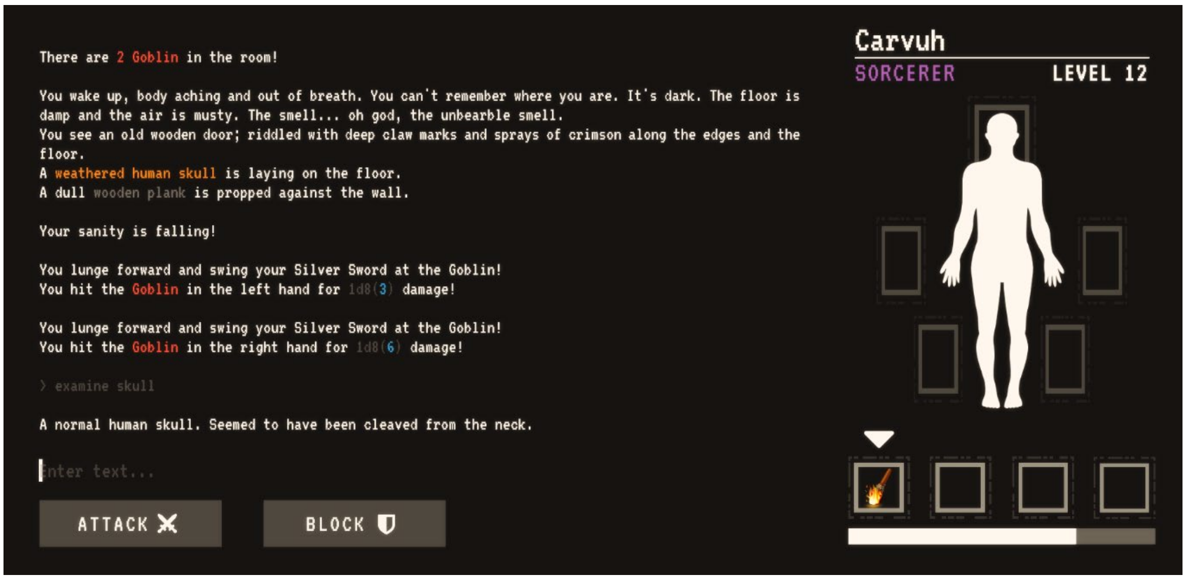
Before you submit the document, it is your responsibility to ensure that you follow the University plagiarism policy (click on the link below to read about plagiarism in detail) and do not breach it.

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.

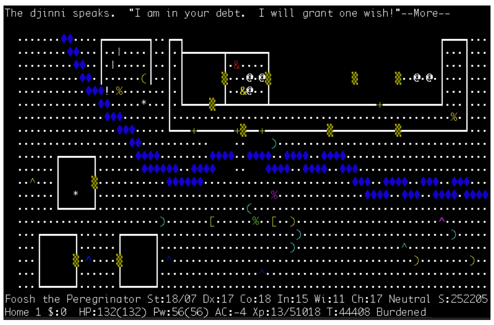
https://www.swinburne.edu.au/current-students/manage-course/exams-results-assessment/plagiarism-academic-integrity/plagiarism-misconduct

Project Theme: Rogue-Like

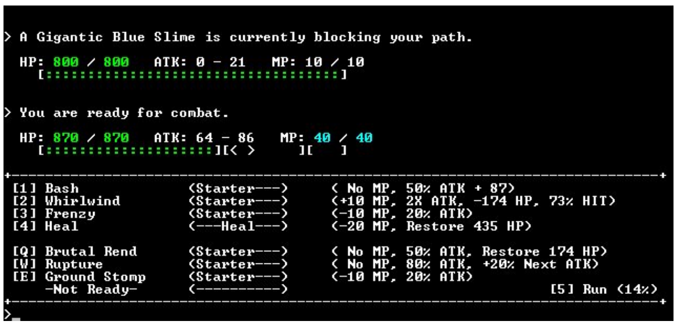
Rogue-Like is a sub-genre that describes computer games involving the player moving through **randomly-generated** environments and overcoming **challenges** (e.g. enemies, traps, puzzles, etc.) with **increasing difficulty**. The game ends either with the player's demise, depleted resources, running out of time or etc. To learn more about this flavour of game, see here: https://en.wikipedia.org/wiki/Roguelike. Some examples of Text-Based Rogue-like games you can try: https://itch.io/games/free/tag-roguelike/tag-text-based. A pretty interesting (and free) one on IOS or Android: https://www.pathofadventure.com/. Here's a top example of (slightly adult) Terminal Based Rogue-Like: https://drl.chaosforge.org/downloads.



```
There are 2 Goblin in the room!

You wake up, body aching and out of breath. You can't remember where you are. It's dark. The floor is
damp and the air is musty. The smell... oh god, the unbearble smell.
You see an old wooden door; riddled with deep claw marks and sprays of crimson along the edges and the
floor.
A weathered human skull is laying on the floor.
A dull wooden plank is propped against the wall.

Your sanity is falling!

You lunge forward and swing your Silver Sword at the Goblin!
You hit the Goblin in the left hand for 1d8(3) damage!

You lunge forward and swing your Silver Sword at the Goblin!
You hit the Goblin in the right hand for 1d8(6) damage!

> examine skull

A normal human skull. Seemed to have been cleaved from the neck.

Enter text...
```



```
The djinni speaks.  "I am in your debt.  I will grant one wish!"--More--

Foosh the Peregrinator St:18/07 Dx:17 Co:18 In:15 Wi:11 Ch:17 Neutral S:252205
Home 1 $:0  HP:132(132) Pw:56(56) AC:-4 Xp:13/51018 T:44408 Burdened
```



```
> A Gigantic Blue Slime is currently blocking your path.

  HP: 800 / 800   ATK: 0 - 21   MP: 10 / 10
    [::::::::::::::::::::::::::::::::::::]

> You are ready for combat.

  HP: 870 / 870   ATK: 64 - 86   MP: 40 / 40
    [::::::::::::::::::::][< >     ][    ]

+------------------------------------------------------------------------------+
 [1] Bash              (Starter---)      ( No MP, 50% ATK + 87)
 [2] Whirlwind         (Starter---)      (+10 MP, 2X ATK, -174 HP, 73% HIT)
 [3] Frenzy            (Starter---)      (-10 MP, 20% ATK)
 [4] Heal              (---Heal---)      (-20 MP, Restore 435 HP)

 [Q] Brutal Rend       (Starter---)      ( No MP, 50% ATK, Restore 174 HP)
 [W] Rupture           (Starter---)      ( No MP, 80% ATK, +20% Next ATK)
 [E] Ground Stomp      (Starter---)      (-10 MP, 20% ATK)
     -Not Ready-       (----------)                               [5] Run (14%)
+------------------------------------------------------------------------------+
>
```

Remember this is a sub-genre. i.e. you can make a racing, puzzle, city-builder, empire-builder, simulator, shooter, hack-n-slash, farming, RPG or etc. game that is Rogue-Like, as long as it has the following characteristics:

a) Randomly generated element (map, challenge, items, etc.) No two games are the same.

b) There must be a finite resource (time, health, etc.) that will run out (ending the game).

c) A save and load game feature.

This project does not require a complete game; just a working prototype is acceptable. You can build a text-based game or you can extend it with external libraries (like SFML) to add graphics, HID inputs and sound to your creation.