Task 9 - Spike: Game State Management

CORE SPIKE

Context: Game state management is a common feature of games.

Knowledge/Skill Gap: The developer is not aware of implementation methods for flexible game state ("stages" of a game) management.

Goals/Deliverables:
[PAPER DESIGN] + [CODE] + [SPIKE REPORT]

You need to create the "Zorkish Adventure" game (Phase 1), as described in the specification document available on the unit website.

You will need to deliver the following items:

1. A simple paper-based plan for your code design. (**REQUIRED!**)
2. Create a simple console program that implements the "Zorkish: Phase I" game (although it has **no** gameplay yet) using a flexible (extensible) game state management method of some kind. The OO State Pattern is the strong suggestion. The implementation must demonstrate the following game stages (states):
    a. "Main Menu" (which allows the user to select other stages…)
    b. "About" (remember to include your own details here)
    c. "Help" (summary of commands – simple hard-coded text is fine)
    d. "Select Adventure" (use a hard-coded list and the title of your test game)
    e. "Gameplay" (test game which only accepts "quit" and "hiscore" commands)
    f. "New High Score" (allows user to enter their name, but doesn't work yet)
    g. "View Hall Of Fame" (view list of name/score. Simple hard-coded text is fine.)

Recommendations:

- Read the complete Zorkish game specification document.
- If not familiar (or you need a reminder) research/read about the state pattern used to represent each "stage" of the game.
- On paper create a design for your implementation. A strong suggestion is a UML class diagram representing a state pattern you would need specific to the Zorkish game.
- Use an "agile" (test often) approach as you implement your design. For example, implement a single state and test, then two states and test changing between the two. Commit to your repo as you go.
- Leave complex issues or issues that might distract from this spike until last.

NOTE: Stay focused on the main points of this spike – state management! Do **NOT** implement a complex gameplay parser, the "Hall of Fame" file IO, scoring features etc. Compare Phase 1 and Phase 2 and read later spikes to see why! Focus on the minimum to get this spike done!