SWEN30006 Software Modelling and Design
Semester 1 2023

Assignment 1 feedback for **Matthew Pham (1273607)**

Criterion	Available marks	Team submission
Build		
Build Success		TRUE
Tests		
Preserve Behaviour #1		TRUE
Test Behaviour #2		TRUE
Test Behaviour #3		TRUE
Test Behaviour #4		TRUE
Overall mark for Assignment 1	12	**11.5**

Feedback on Team Submission

Implementation
The implementation preserved the original structure according to the design model

Design Report,

Demonstration of good knowledge of GRASP principles and discuss the strengths/weaknesses of alternative designs

The report demonstrates a good understanding of GRASP principles by identifying issues in the original design, such as the lack of polymorphism, indirection, protected variation, and low cohesion (Section 1.1, 1.2, 1.3). The report also discusses the strengths and weaknesses of alternative designs, such as the proposed Item class (Section 2.1) and the Abstract Entity and Abstract Monster classes (Section 2.2).

Discussion of alternative designs
- The report argues that items should be implemented as separate classes to address the lack of polymorphism, high coupling, and other issues with the original design (Section 1.1). The proposed fix involves creating ItemHandlers, Items, ItemEventListeners, and ItemEventCodes (Section 2.1).
- The report does not specifically suggest a separate game logic class but rather proposes refactoring the responsibilities of Game and PacManGameGrid classes to achieve higher cohesion and protected variation (Section 2.3).
- The report does not directly address breaking the monster's state into a separate class. However, it proposes creating Abstract Entity and Abstract Monster classes to address low cohesion and lack of polymorphism (Section 2.2). The report also discusses extending the Monster class with subclasses that override the abstract walkApproach() method (Section 3).

Not expected to incorporate GoF patterns, but will be a plus if you have:
- Seems that the student incorporated observer pattern as ItemEventListeners

Justifying design decisions is much more valuable in comparison with explaining what your code

does
The report effectively justifies the design decisions made, such as the creation of ItemHandlers, Items, ItemEventListeners, and ItemEventCodes (Section 2.1) and the Abstract Entity and Abstract Monster classes (Section 2.2).
Stronger submissions will also discuss future extensibility, including understanding of the codebase by future team members:
The report discusses future extensibility by highlighting how the proposed design changes will make it easier to add new items (Section 2.1) and create new types of monsters (Section 2.2). The report also briefly mentions how the new design will be easier for future team members to understand (Section 2.3, Diagram 3).

Domain Model
- Redudant information: static attribute as well as access modifier.

Design Class Diagram
- Identified correctly the classes and their relationships: inheritance, and general association
- However, Design class diagram is not a domain model:
- If one class creates another but does not store, it will be a dependency (-->).
- It will be composition/aggregation if one class always stores another (--> *).

State Machine Diagram
- [x] Correctly identified the state and transition

Verdict
Appropriate Design (+ Clarity of the report): 5 pts

The student demonstrates a good understanding of GRASP principles and discusses alternative designs, addressing weaknesses in the original design and justifying their design decisions. They also discuss future extensibility and the potential impact on future team members understanding the codebase. The student incorporated the observer pattern in their design.
Diagrams (Domain Model, Static Design Model, Dynamic Design Model): 3.5 pts

Domain Model: Redundant information (static attribute and access modifier). The overall structure is still identifiable.
Design Class Diagram: Correctly identified classes and their relationships (inheritance and general association). However, the diagram is missing the appropriate dependency and composition/ aggregation relationships.
State Machine Diagram: Correctly identified states and transitions.
Implementation: 3 pts

The implementation preserved the original structure according to the design model.
Total score: 11.5 pts out of 12 pts.