SET Game OOP Project Documentation

1. Project Overview

This project is a Java-based implementation of the card game SET. It features a full game logic engine, graphical user interface (GUI), and support for both human and AI players. The game includes core concepts of object-oriented programming such as encapsulation, inheritance, abstraction, and polymorphism.

2. Key Features

\* 81 unique cards, each defined by 4 attributes (color, shape, filling, number)

\* A board that can hold up to 15 cards

\* Game logic to identify valid SETs

\* Human vs AI gameplay mode

\* Graphical interface using Java Swing

\* Threaded AI player with basic strategy

3. Class Descriptions

Card

Represents a single SET card with 4 enum-based attributes: Color, Shape, Filling, and Number.

\* Provides getters and image path mapping

\* Contains a static method isSet() to validate three cards as a SET

Deck

Stores all 81 cards and allows card drawing and shuffling.

\* Supports drawing single or multiple cards

\* Ensures cards are drawn in groups of 3

\* Tracks remaining cards and resets when needed

Board

Holds cards currently in play and allows set validation and removal.

\* Max capacity of 15 cards

\* Offers methods to add/remove cards and check for SETs

\* Handles game rules such as board fullness and clearing

Player (abstract)

Base class for game participants with name and score.

\* Includes abstract play() method and a printInfo() method

HumanPlayer

Handles human interactions via GUI.

\* Overrides play() (left empty since interaction is manual)

AIPlayer

Threaded player that scans the board to find and play a valid set.

\* Uses Runnable and separate thread

\* Interacts with the board and GUI

\* Score increases when a valid set is played

Game

Main logic controller that connects players, board, and deck.

\* Manages initialization and resetting

\* Checks for game-over conditions

GameGUI

Swing-based GUI interface.

\* Displays board with card images

\* Allows human interaction and tracks selected cards

\* Updates score and game state dynamically

\* Starts and stops AI threads

4. UML Diagram

(Include as image or PDF showing Card, Deck, Board, Player, HumanPlayer, AIPlayer, Game, and GameGUI relationships)

---

5. How to Run

1. Compile all files in the set.core and set.gui packages.

2. Place all 81 card images in a folder named images/ inside the classpath.

3. Run `GameGUI.java` from set.gui package to launch the GUI.

6. Design Decisions

\* Used enums to strictly define attribute values for cards

\* AI is kept simple and operates in a single thread for demonstration

\* Board logic handles rule enforcement (e.g., only sets of 3, max 15 cards)

7. Testing and Known Issues

\* Manual testing performed through GUI interaction

\* Basic main() methods exist in some classes for unit testing

\* Known issue: AI might occasionally act on an outdated board state if interrupted

\* Known issue: Some card images are not loading properly, which results in missing cards on the board display

8. Team mates

\* Lika Mirzoyan

\* Liazzat Syzdykova

\* Mariam Ghalechyan