PROJECT ACCOMPLISHMENTS

Design

- game Thread keeps the program running consistently and smoothly

  GamePanel.run()

- code is minimized through inheritance and secured with encapsulation

  Formation – EnemyFormation; Entity – Samurai

- Java conventions maintained

  use of getters and setters, proper naming, correct identifiers

- efficient planning of algorithms

  no use of recursion, no algorithm more complex than O(n log(n))

  Samurai.move(), Samurai.rotateRight(), Samurai.rotateLeft()

- use of pixel array rendering, not paint() and paintComponent(), images are drawn through arrangement of pixel values and individual changes instead of refreshed graphics each time

  GamePanel.render();

- appropriate comments

  comments for each method with function, parameters, and return variables explained

Project Supplements

- project plan with entire design process explained and recorded

  Project Plan.pdf

- data flow diagram and waterfall diagram

  included in Project Plan, sections 10.1 and 2

- sheet music for composed theme

  music score.pdf

- Javadocs

  Javadoc_index

- UML Class Diagram

  UML Class Diagram.pdf

Game Features

- unique player input with mouse and keyboard control
- story accompanying game, with animated enemies advancing smoothly pixel by pixel instead of block by block such as Tetris and Dr. Mario
- two way movement and left and right rotation at all times
- advanced scoring system that rewards intelligent plays and fitting pieces together appropriately
- damage and health system that gives another facet to game and forces player to be engaged with multiple aspects of the game
- increasing levels of difficulty as time progresses; enemies move faster and faster and the game requires more dexterity and strategy
- multiple types of enemies
- various formation of enemies
- enemies can be cleared in two ways, color combinations and filling rows
- bonuses are given for eliminating enemies (slow time, increase score, health regeneration, and visible placement grid)
- in-game tutorial that teaches how to play

Error Logging and Saving

- crash report generated on SEVERE level errors

  Logging

- error handling through logging errors and debug notes; errors logged to file

  Logging.debug(Throwable exception, String msg);

  data/debug.log & data/error.log

- data saved to text file to record high scores and player names

  data/scores.txt