syst 17796 Deliverable 1
design document template

Overview

1. Project Background and Description

Describe the project goals and final vision. Include a brief description of how to play the game you have chosen and a reference to the rules of the game you have chosen. Also describe the current starting base code. Use technical terms to describe the code including what language it is written in, any patterns you can see and any coding conventions used.

The goal of our project is to evenly distribute the workload across all group members. Also, we plan on completing our deliverables early enough so that we have enough time to ask questions if necessary. Our final vision for our project is to create a functioning Go Fish card game. The game begins with seven cards being dealt between two players from a standard 52 card deck. The same rule applies for 3 players, however from 4-5 players each player hand consists of 5 cards. The objective of the game is to collect 4 cards of the same face value (“book”) by asking the opposing player if they have a card of the same value of one of yours. The player that is being asked for cards must have at least one card of that face value otherwise they will have to “Go Fish” and pick up another card from the deck.

Current base code written in Java gives us 4 classes which have 2 associations. The Game class is associated with the Player class as every “game” has a “player”, and the GroupOfCards class is associated with the Card class as every group of cards has a “card”. Encapsulation is brought into 3 of the 4 classes through their data fields, and the Card class is that of an abstract class. There is a very probable chance that our group shall create subclasses from the Card class in order to delegate Value and Suit respectively. If this is the case, considering that Card class is abstract, any abstract methods found will need to subsequently be overridden by these subclasses. Array Lists are present for the amount of players and the amount of cards. We look to utilize the card array list in order to both add and subtract cards from the players hand and solely subtract from the deck.

2. Project Scope

Describe the names and roles of each team member. Describe the technical scope of the project by talking about the interface and how you will know when the project is complete.

Henson Mak - Developer
Sean Jaeger - Developer
Georgy Dhanjal - Project leader, Developer
Mostafa Allahmoradi - Developer

The technical scope of our project is to replicate a game of Go Fish between 2-5 players. In order for this to successfully work we will implement multiple OO methods such as conditional operations in order to compare values between each players hand, multiple loops (one of which is important in that it will discontinue once the total amount of books will equal 13) and a utilization of arrays (particularly an array list in order to populate each players hand and accommodate for “Go Fishing!”). With constant refinement over these implementations over the course of our project timeline, our project shall be functioning properly come submission.

3. High-Level Requirements

[Describe the high level requirements for the project. For example:]

The new system must include the following:

- Ability for each player to register with the game
  - -Before starting the game, players will enter their name and then be assigned to a class consisting of “player 1-5” respectively.
- Ability for the game to communicate a win or loss
  - -For Go Fish, each player aims to create a “book”, where they will then gain one point. Knowing this, once all 13 “books” are collected, the player with the most amount of points will win.
- Ability for players to know their status (score) at all times
  - -A counter which will update and be visible during each player’s turn which will display the total amount of points for each player. Knowing this, each player can keep track of how many “books” have been collected by summing player scores up and subtracting it from the total amount of “books” (13).

4. Implementation Plan

Include your Git repository URL here and a brief description of the expected use (i.e. each developer checks in code at the end of each day/week). Text files are stored under a separate directory, code, UML diagrams have their own folders etc.
Include information on coding standards you intend to follow and tools you expect to use (VP, NetBeans, eclipse, Junit…)

Repository URL:
https://github.com/GeoDh/DeliverableGroup7

Each developer will check the code on Github on a weekly basis. Group members will branch to the repository and we will come together to decide what to merge to the master. Group

members will be in contact with text documents over Google Drive, a WhatsApp group has been made to maintain contact for the project.

Netbeans will be utilized as the JVM and the coding standards we will attempt to properly include are the current OO methods which we have learnt plus some additional learning outside of our given classes.

5. Design Considerations

Talk about how the current code is structured as it relates to the following OO principles. Each principle should have 2 or 3 specific examples from the base code or your intended additional code (i.e. potential for improvement).

- Encapsulation
  -playerID (already in the base code), Card Value (will be added) and Card Suits (will be added) shall be encapsulated so that the players will not be able to adjust the card values or player names. The only data field which we will want to have any player able to adjust is the playerID through mutators.

- Delegation
  -We shall create a main class which will act as the controller of Go Fish. This controller will invoke the Game and Group of Cards classes respectively to construct the basis of whom is playing and the manipulation of the deck (shuffling, dealing, deck size).
  -By having our classes delegated rather than inherited, we are loosely coupling our classes yet the methods brought about each class will maintain high cohesion.
  -The sole probable class which will take on Inheritance looks to be the Card class as we will most likely create subclasses of “value” and “suit” respectively.

- Flexibility/Maintainability
  -Coinciding with the previous usage of delegating classes, the flexibility and maintainability of our program will not be difficult as adjusting our parameters will be adjustable through their respectable classes.