SWEN30006 Software Modelling and Design
Project 2: Cribbage Trainer

School of Computing and Information Systems
University of Melbourne
Semester 1, 2021

Overview

You and your team of independent Software Contractors have been hired by *New, Exciting and Revolutionary Designer Games* (aka *NERD Games*) to provide some much-needed assistance in developing their innovative *Cribbage* card game trainer (see details below) to be market ready.

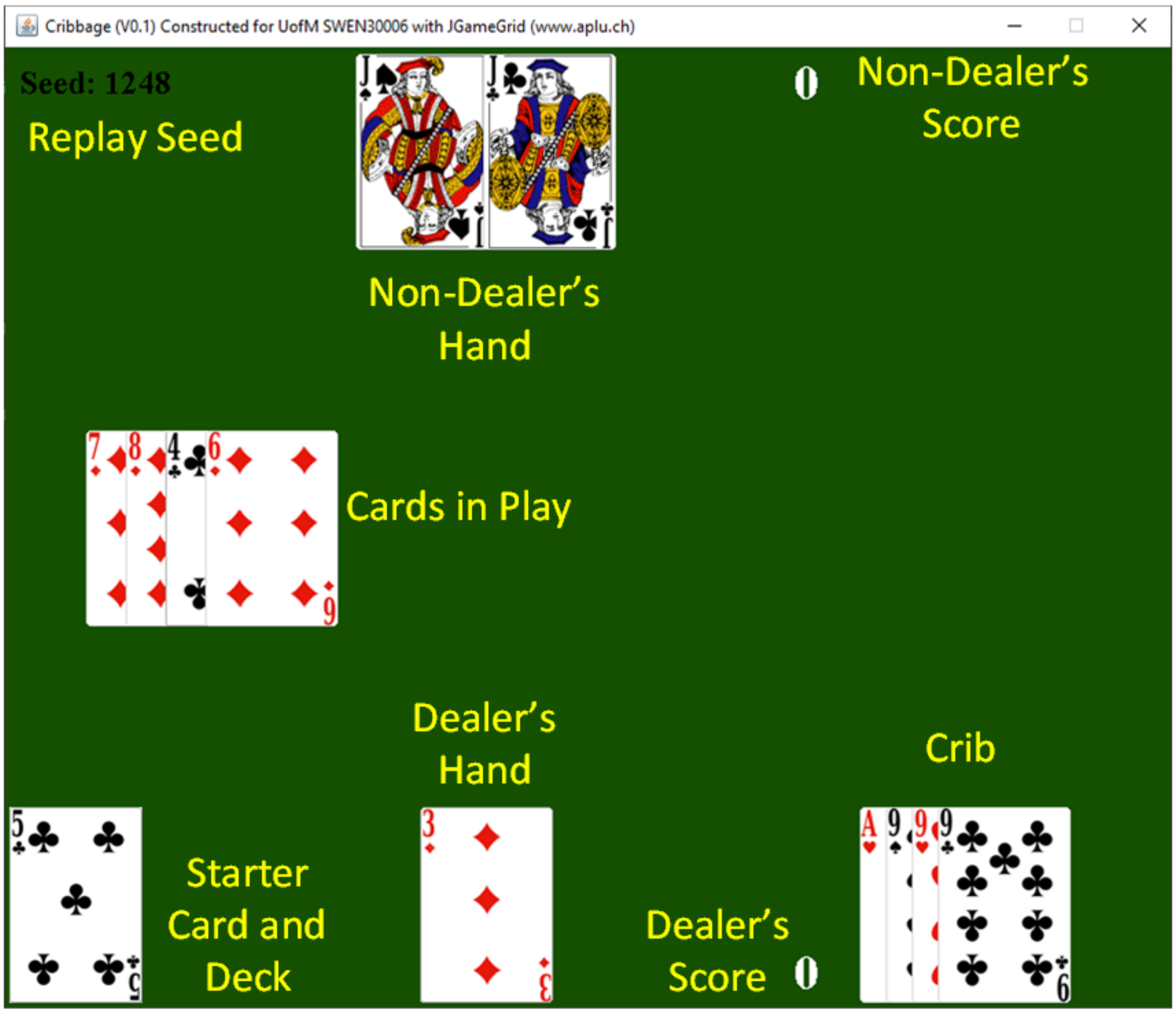

Figure 1: Advanced Cribbage CGI

NERD Games have a configurable, running version of the trainer, but it does not provide any scoring, or log the run sessions.

Your task is simple: add scoring and logging to the trainer, with a flexible design which will allow for future maintenance and modification by the NERD team. You are free to modify the existing design and implementation to facilitate these changes, with some limitations (see below). You will also need to provide a report to the NERD team explaining and justifying your changes to the design (details below).

Playing the Cribbage Card Game

- Cribbage is played with a standard fifty-two card deck with four *suits* and thirteen *ranks* in each suit.
- The game involves two *players* who play independently. The hand and score for Player 0 appears at the middle-top of the window, and the hand and score for Player 1 (the dealer) appears at the middle-bottom.

Cribbage is usually played until one of the players reaches a score of 121 points, with that player declared the winner. The game consists of a series of *rounds*, however the first player to reach 121 points wins immediately, even if this is in the middle of a round. Each round has clear rules about how play proceeds, and how and when points are scored. For this training system, we are interested only in playing one round, possibly multiple times over, to see how the choices made affect the scores.

A round consists of the following activities:

1. Dealing
2. Discarding to the Crib
3. Selecting the Starter card
4. The Play
5. The Show (show hands for scoring)

In the game of Cribbage, the cards are dealt face down, with most cards being revealed only during the Play.
For our training system, the cards-of-interest are all face up.

The italicised terms in parentheses below, for example *(starter)*, are used for logging which is also discussed below.

Dealing

The game starts with a hand of six cards being dealt to each player by the dealer. The rest of the deck is placed to the left of the dealer's hand.

Discarding to the Crib

Each player then selects and discards two cards to the *Crib* which is to the right of the dealer's hand. This leaves four cards in each player's hand, and four cards in the Crib. The player's hands are used in the Play as described below, and all the hands including the Crib will be scored as part of the Show. Points in the Crib are scored by the dealer so the dealer will try to maximise the points there while the non-dealer will try to minimise the points there.

Selecting the Starter card

The non-dealer then randomly selects a card from the deck and turns it over on top. This card is called the Starter.

- If the Starter is a Jack, the dealer gets two points immediately (*starter*).

The Starter is not used in the Play but is used in the Scoring.

The Play

The game play depends on the face values of the cards. The cards from Ace to Ten have face values 1 to 10 respectively, while Jacks, Queens and Kings are all have a face value of 10. Play proceeds as follows.

The non-dealer lays down a card from their hand. Play then alternates unless a player has no card they can legally play. (If a player has cards but cannot legally place any card on their turn, they say "go" to indicate this to the other player.) Players place cards until a total face value of 31 has been reached, or until no player can place a card without exceeding 31. This segment of the play is now complete. The next segment begins with the player *who did not place the last card* in the previous segment (unless that player has no cards in which case it will revert to the other player).
Scoring takes place during each segment, as play proceeds. Each segment is scored independently; neither earlier segments nor later segments influence the scoring of a segment. Points can be earned during play in a variety of ways, w3hich are listed below.

1. Totals and last cards:
    a. If a player places a card which results in a total face value of 15 (*fifteen*) or 31 (*thirtyone*), they score two points
    b. If a total of 31 has not been reached, and no player can place another card without exceeding 31, the player who placed the last card scores one point (*go*)
2. Runs
    a. three points for completing a run of three cards, regardless of the order in which they are laid: a 6, then a 4, then a 5 is a run of three even though they were not laid in order (*run3*).
    b. four points for completing a run of four (*run4*)
    c. five points for completing a run of five (*run5*)
    d. six points for completing a run of six (*run6*)
    e. seven points for completing a run of seven; e.g. playing 2, 4, 6, A, 3, 5 and 7 (*run7*)
3. Pairs
    a. two points for laying a card of the same rank as the previous card (*pair2*)
    b. six points for laying a third card of the same rank (*pair3*)
    c. twelve points for laying a fourth card of the same rank (*pair4*)

If a card completes more than one scoring combination, then all combinations are scored. For example, if the first three cards played are 5s, the third one scores eight points: two for *fifteen*, and six for *pair3*. During this phase of play run combinations cannot span a pair; in a play of 2, **3, 3**, 4 the pair of three's disrupts the run so only the pair is counted for points.

Players choose when to lay each card to maximise their score according to the scoring scheme above.

The Show

During the Show, the hands are scored in the order: the non-dealer's hand, the dealer's hand, and the crib (with the points going to the dealer). The hands are scored based on five cards: their four cards plus the Starter card.

Points are scored by a hand during the show in a variety of ways:

1. Fifteens: two points for each separate combination of two or more cards totalling exactly fifteen (*fifteen*)
2. Runs
    a. three points for a run of three consecutive cards, regardless of suit (*run3*)
    b. four points for a run of four (*run4*)
    c. five points for a run of five (*run5*)
3. Pairs
    a. two points for a pair of cards of a kind (*pair2*)
    b. six points for three cards of a kind (*pair3*)
    c. twelve points for four cards of a kind (*pair4*)
4. Flush: four points for a flush, where all four cards in the hand are of the same suit (*flush4*), with an additional point if the starter card is also of that suit (*flush5*).
5. Jack of the same suit as the Starter card: one point (*jack*)

Various aspects of the game are configurable in the properties file "cribbage.properties": the random seed, the animation, and the players. The seed can be re-used to replay a round to potentially try different choices. The seed can also be overridden on the command line.

The Cribbage trainer currently supports two types of players: *human* interactive players who select the card to play through a double-left-mouse-click, and one type of NPC, *random* players who select a legal card to play from their hand randomly without regard for the score.

Required Changes and Reporting

Scoring

The Cribbage trainer does not provide any scoring. You need to add scoring as per the rules above.

Note that there are regional and other variations to the rules above, for example, not all rules allow scoring for a Flush, and not all variations have the same point scores. While you do not need to implement any variations, your design should account for the situation, in future, where the system may be extended to be configurable to handle scoring variations.

Logging

To help players study the impact of their choices on the play and score[1], the activity of the Cribbage Trainer needs to be logged to a file called "*cribbage.log*". The events to be logged and format of this log file are described below. Note that the log file should be written for the current run, not appended to previous runs.

Cards are to be referred to by two letters, the first for rank (one of: K,Q,J,T,9,8,7,6,5,4,3,2,A) and the second for suit (one of: C,D,H,S). A hand is shown by listing comma separated cards between square brackets in descending order of rank and alpha order of suit within rank (see examples below). You have been provided with code for generating this *canonical* form for representing hands.

Events are logged, one per line in the order they occur, in comma separated format without whitespace as per the example below of the expected log file contents for 2 players of type "cribbage.RandomPlayer" with seed "55510":

```
seed,55510
cribbage.RandomPlayer,P0
cribbage.RandomPlayer,P1
deal,P0,[2C,2H,7C,QC,QD,KH]
deal,P1,[AD,AH,TD,JS,QH,KC]
discard,P0,[2C,QC]
discard,P1,[JS,KC]
starter,8H
play,P0,2,2H
play,P1,3,AD
play,P0,13,QD
play,P1,23,QH
score,P1,2,2,pair2
play,P0,30,7C
play,P1,31,AH
score,P1,4,2,thirtyone
play,P0,10,KH
play,P1,20,TD
score,P1,5,1,go
show,P0,8H+[2H,7C,QD,KH]
score,P0,2,2,fifteen,[7C,8H]
show,P1,8H+[AD,AH,TD,QH]
score,P1,7,2,pair2,[AD,AH]
show,P1,8H+[2C,JS,QC,KC]
score,P1,10,3,run3,[JS,QC,KC]
```

[1] And to help us assess the correctness of your implementation by automatically processing and comparing the output your code produces.

A score event such the last one is read as “Player P1 scored 3 points for the run of 3 cards [JS,QC,KC] taking their score to 10”. All event types are depicted in this example. Not all possible ways of scoring are shown in the example. If a Jack is turned up as the Starter, the dealer gets two points immediately; this would be shown as a scoring event like:

```
score,P1,2,2,starter,[JH]
```

If a hand scored in the Show gets points in more than one category, scores should be shown in category order (order listed above in the subsection “The Show”). If the hand gets points on more than one set of cards in a category, the scores should be shown in hand order (alphabetic order of the canonical representation). For example, the log could include a sequence of scores like:

```
score,P1,10,2,fifteen,[5H,JC]
score,P1,12,2,fifteen,[5H,JH]
score,P1,14,2,pair2,[JC,JH]
score,P1,15,1,jack,[JC]
```

If these instructions are not completely unambiguous about the format required, you will need to seek clarification from the client.[2]

Report

You are required to produce a **report** describing the changes you have made to the system and providing a rationale for these changes **using design patterns and principles**. Note that to do this well, you should include discussion of other reasonable options which you considered, and why you did not choose these options. You should clearly identify the points of evolution/variation supported by your design with appropriate justification.

Around 4 pages should be enough to provide a concise rationale. You should include and refer to diagrams in your report (as part of the report or separately) to help illustrate aspects of the changes.

The Sample Package

You have been provided with an Eclipse project export representing the current system.

To begin:

1. Import the project zip file as you did for Workshop 1.
2. Add the JGameGrid.jar file (in dist/lib) to the Java Build Path for the project: Project > Properties > Java Build Path > Add Jars …. You may need to remove it from the build path and re-add it if the path is not correct.
3. Try running by right clicking on the project and selecting **Run as... Java Application**.
4. You should see a window like that in Figure 1; you can then play the game as the interactive player.

The current system should be used as a starting point. Please carefully study it and ensure that you are confident you understand how it is set up and functions before continuing. You will need to preserve the game play behaviour, including the use of randomisation. However, you are free to make whatever changes to the code you wish, given the game play behaviour is preserved. If you have any questions, please make use of the discussion board; it is assumed that you will be comfortable with the package provided.

Note: By default, the simulation will run without a fixed seed, meaning the results will be random on every run. To have consistent test outcome, you need to specify the seed in the configuration file or on the command line.

Note: There are three required tasks here – design, code, and report – which are interrelated. You should work on these tasks **as a team**, not separately. Every team member needs to be able to demonstrate their understanding of and contributions to all deliverables.

[2] Via the project discussion forum or project FAQ.

Testing Your Solution
We need to be able to build and run your program without using an integrated development environment. We will be running your application via script with appropriate property files and applying automated checks to the output. The entry point must remain as "Cribbage.main()". Other than JGameGrid, you should use only standard Java libraries (using other libraries will result in a failure of your system to build).

Note: It is **your responsibility** to ensure that you have thoroughly tested your software before you submit it.

Marking Criteria
This project will account for 20 marks out of the total 100 available for this subject. The rubric for the project is available on the LMS.

Note: We reserve the right to award or deduct marks for clever or very poor code quality on a case-by-case basis outside of the prescribed marking scheme. Further, we expect to see good variable names, well commented functions, inline comments for complicated code. We also expect good object-oriented design principles and functional decomposition. For UML diagrams you are expected to use UML 2+ notation.

On Plagiarism: We take plagiarism very seriously in this subject. You are not permitted to submit the work of others under your own name. This is a **team** project. More information can be found here: https://academichonesty.unimelb.edu.au/advice.html.

Submission
Detailed submission instructions will be posted on the LMS. You should submit all items shown in the checklist below. You must include your team number in all your pdf submissions, and as a comment in all changed or new source code files provided as part of your submission.

Submission Checklist
1. Your complete updated source (only, not the images and other elements of the provided package).
2. Design Analysis Report (pdf) and any external diagrams (pdf or png).

Note: Only one member of the team to submit.

Submission Date
This project is due at **11:59pm on Friday May 28th**. Any late submissions will incur a 1-mark penalty per day unless you have supporting documents. If you have any issues with submission, please email Green at green.vo@unimelb.edu.au **before** the submission deadline.