CPSC 101 Team Term Project

Winter 2019

2023-01-09 v1 **Dr David Casperson**

This document provides the problem statement a description of the game of Cribbage.

1. A detailed list of tasks and deadlines,
2. a document describing the design process,
3. a document on what is generally required of formal reports, and
4. a sample design document

are provided on the web-page

`https://web.unbc.ca/~casper/Semesters/2023-01W/101-project.php`.

1 Problem statement

The goal of this project is to write programs to play the two-person variant of the card game called cribbage. The game itself is described elsewhere.

1.1 Programming tasks

Your team needs to write a cribbage program that consists of a "referee", some kind of user interface, a human player interface, and a computer opponent.

The Referee The referee program minimally allows a person to play cribbage against the computer. It keeps track of whether either player has won the current game; lets the human player quit or restart the game at any time (s)he chooses; and draws (or otherwise displays) the board so that the human player can see the current game situation.

The Display Ascii-graphics such as those shown in Figure 1 are acceptable for displaying the board. A Graphical User Interface (GUI) is preferred, but you should plan on developing both ascii-graphics and a GUI. The ascii-graphics allow you to test parts of your project while you are sitll developing the GUI.

The Computer Player The exact mechanism that the referee program uses for interacting with the computer player is left unspecified for now.

However, a longer term goal is to have the various teams' computer players play each other in a tournament. In order for this to happen the Computer Player needs to have minimal coupling with its environment, and this should be a design goal.

For the computer player program, correctness is far more important than cleverness. The computer opponent program must work correctly. However, intelligent play by the computer opponent is not necessary, and should not be a priority when completing the team term project.

The Human User Human player input and output happens at some level through the User Interface. However, it helps a lot if the human player and the computer player are represented by similar classes.

```
    Your cards:  AS 2H 6D 10S JS QS

.. ..... ..O.. ..... .O... ..... ..... ..... ..... ..... ..... ..... .....
.. ..... .O... ..O.. ..... ..... ..... ..... ..... ..... ..... ..... .....

Cut card:  ??                 Crib: ?? ??

Computer cards:  ?? ?? ?? ??
```

Figure 1: ASCII rendition of a cribbage board.

1.2 Interface mechanisms and tournaments

In previous years I have insisted that the computer opponent be a stand-alone prorgram that communicated via `System.in/out`. My goals were two-fold:

1. to force teams to re-use classes in multiple programs, and
2. to make it possible to have an automated tournament where the AI's played against each other.

This didn't work well in practice. I won't require that for this year's project. However, I may yet specify an interface that each AI program must implement, as this will allow me to create a program where the various project AIs play against each other.

```
import interfaces.WhichHand;
public interface ComputerPlayer
{
  abstract public void startNewGame       (boolean amDealer) ;
  abstract public void startRound         (boolean amDealer) ;
  abstract public void startPeggingRound  () ;
  abstract public void cutCardIs          (CardView c) ;
  abstract public void cutCardScores      (boolean countsForYou, int score) ;
  abstract public void peggingIsOver      () ;
  abstract public void roundIsOver        () ;
  abstract public void handScores         (WhichHand w, int score) ;
  abstract public void gameIsOver         () ;

  abstract public void youGetDealt        (CardView [] initHand) ;
  abstract public CardView [] yourDiscards(boolean amDealer) ;

  abstract public CardView yourPlay       () ;
  abstract public void opponentPlays      (CardView aCard) ;

  abstract public void youSayGo           () ;
  abstract public void opponentSaysGo     () ;
  abstract public void lastCardScores     (boolean wasYou, int score) ;
}
```

Figure 2: Possible format of an interace for AI players.