## COM410 Software Development II

## Assignment 2: “Elevens” Card Game

**Submission Deadline: 12:00pm (Noon) on Friday 18th December 2020 (Week 13)**

**This assignment carries 60% of the overall coursework mark for the module.**

### Introduction

***Elevens*** is a single-player card game that uses a standard deck of playing cards. The object of the game is to remove all 52 cards from the playing area by matching them in one of 2 ways as follows:

i) by removing a pair of (non-picture) cards for which the combined rank value is 11 (i.e. a pair of cards where neither is a Jack, Queen or King). The rank value of an Ace is 1 and cards from 2-10 take the rank value of their number. Pairs that can be removed are therefore Ace and Ten, Two and Nine, Three and Eight, Four and Seven, Five and Six.

ii) by removing a King, Queen and Jack (of any suit or mixed suits) at the same time.

The game begins by dealing 9 cards from a shuffled deck, so that the cards are face-up (i.e. the values of the cards are visible). The player examines the 9 cards and, if two or three cards can be seen that satisfy either of the removal conditions above, those cards can be removed. As cards are removed, they are replaced by cards from the remainder of the deck, for as long as the deck has cards remaining.

The game is won when all 52 cards are removed from the game. If a stalemate is reached where neither of the conditions for removal can be satisfied, then the game is lost.

The rules of Elevens can be seen online at https://gamerules.com/rules/elevens/ and a demonstration video can be seen at https://www.youtube.com/watch?v=c-Yi1WlwQAk

Note that your application will be a pure text version – with keyboard input. There is no requirement for a graphical or mouse-based solution.

### Required Functionality

There are 6 levels to which the submission can be completed – attracting increasing rewards.

**Level 1:** The facilities to create a new game are present and the game can be set up in its initial state, with 9 cards dealt face-up from a shuffled deck.

**Level 2:** The basic game mechanism is in place. Users can select legal combinations of cards to be removed and they are replaced by new cards dealt from the deck (for as long as the deck has cards available).

**Level 3:** A full playable game is available. The application is able to prompt the player when the game is won (all cards removed) or lost (stalemate is reached as no cards can be removed).

**Level 4:** The game is able to provide a hint to the player (on request) suggesting a valid move or confirming that no move is possible.

**Level 5:** On completion of a game (whether the player has won or lost) the application is able to replay the game move by move, with the user prompting each replayed move by a keypress.

**Level 6:** The application is able to play a complete game in "demonstration mode", where the only input from the player is to prompt the application to play its next move. The computer should provide a commentary on each move made such as "3 of Spades and 8 of Diamonds removed", "King of Clubs, Queen of Diamonds and Jack of Hearts removed" and so on.

## Deliverables

The deliverables for this assessment are as follows:

1. **"Elevens" Application** – A Zip file containing all code developed for the application. The easiest way to create this is to Zip the contents of the **src** folder of your Java project. Each Java file should also be provided as a PDF document.

2. **Design and Development Document** – A PDF document that contains a written account of the design and development decisions made during the development of your application. You should provide evidence of your consideration of data structures used and algorithms developed, with justification for your decisions and rejection of alternatives.

3. **Testing Document** – A PDF document that describes your testing regime with details of any test classes developed. This document should also describe steps you have taken to ensure the security of your implementation.

4. **Collaboration Summary** – see later.

5. **Demonstration Video** – An individual video of no more than 5 minutes duration, demonstrating a walkthrough of your application running and a walkthrough of your code base with an associated commentary concentrating on the parts of the code for which you were responsible. You do not have to appear on video in person, but a voiceover commentary is mandatory. Videos should preferably be recorded using Panopto (available in the module area on Blackboard), though other packages may be used as long as the video is in MP4 format. Please note that videos with a duration in excess of 5 minutes will be penalised according to the following scheme.

| | |
|---|---|
| **5 minutes + 10%** | no penalty |
| **5 minutes + 10% - 20%** | reduction in total mark by 5% |
| **5 minutes + >20% - 30%** | reduction in total mark by 10% |
| **5 minutes + >30% - 40%** | reduction in total mark by 15% |
| **5 minutes + >40% - 50%** | reduction in total mark by 20% |
| **5 minutes + >50%** | maximum total mark achievable is 40% |

## Organisation

This assignment will be completed by self-selected groups of 1, 2 or 3 students working collaboratively. No account will be taken of the size of the group when assessing the submission.

Students should organise themselves into groups and inform the module team by submission to Blackboard (in the Group Assignment section) no later than **Noon on Friday 13th November 2020** (Week 5). After this time, any students not self-selected into groups will be randomly assigned into groups of 2 or 3 members and the final group list will be published on Blackboard on **Monday 16th November**.

Although the assignment is a group activity, ALL students are required to submit an individual Collaboration Summary - an account of the contribution made by each member of the group. A template for the Collaboration Summary will be available on Blackboard. **By default, all members of the group will be awarded the same mark UNLESS these collaboration summaries indicate an inequality of effort.** In this case, the group members will be interviewed to ascertain the appropriate distribution of marks.

## Restriction

The Collection classes from the Java Class Library **MUST NOT** be used within a submission. Any lists, stacks, queues, bags or other data structures must be implemented from scratch as part of the submission.

## Submission

All students must make a submission of the complete set of deliverables to the Assignment 2 link on Blackboard. If a group member does not make a submission, it will be assumed that they played no part in the development process.

## Feedback

Feedback on each final submission will be provided in written form under the categories presented in the marking criteria that follows this specification. All results and feedback will be returned by **Friday 8th January 2021**.

## Plagiarism Declaration

By making a submission, you will be deemed to have made the following declaration of ownership. Source: http://www.ulster.ac.uk/academicservices/student/plagiarism.pdf.

*"I declare that this is my own work and that any material I have referred to has been accurately and consistently referenced. I have read the University's policy on plagiarism and understand the definition of plagiarism as given in the [course/subject] handbook. If it is shown that material has been plagiarised, or I have otherwise attempted to obtain an unfair advantage for myself or others, I understand that I may face sanctions in accordance with the policies and procedures of the University. A mark of zero may be awarded and the reason for that mark will be recorded on my file."*

## Submission Assessment Grid

| Group/Student: | | | | | |
|---|---|---|---|---|---|
| | Poor (<40%) | Satisfactory (40-59%) | Good (60-69%) | Excellent (70%+) | Mark |
| Design and planning (15 marks) | Limited understanding of fundamental ADT design concepts shown. Minimum set of operations defined Lack of discussion of ADT design and/or poor justification for ADT design choices made | Some understanding of fundamental ADT design concepts shown. Adequate set of operations defined. Adequate data storage representation design demonstrated. | Majority of ADT operations defined with appropriate data storage representation design demonstrated. Detailed discussion of ADT design with good justification for design choices made | Demonstrates excellent awareness of alternatives for ADT selection and good justification for decisions made. Most efficient representations for ADTs selected and clearly justified. | |
| Implementation (15 marks) | Limited knowledge of ADT development techniques shown. Minimum set of ADT operations demonstrated. Major issues with implementation of data storage representation. | Some knowledge of ADT development techniques. Adequate set of ADT operations clearly demonstrated. Some issues with implementation of data storage representation. | Good knowledge of ADT development techniques. Majority of ADT operations clearly demonstrated. Minor issues with implementation of data storage representation. | Excellent knowledge of ADT development techniques shown. Complete set of ADT operations clearly demonstrated.<br>Insightful discussion of most technically challenging aspect | |
| Functionality (40 marks) | **Less than L1.** A game that is not playable with significant components missing or not working. | **L1.** The facilities to create a new game are present and the game can be set up in its initial state, with 9 cards dealt face-up from a shuffled deck.<br>**L2.** The basic game mechanism is in place. Users can select legal combinations of cards to be removed and they are replaced by new cards dealt from the deck (while the deck has cards available). | **L3.** A full playable game is available. The application is able to prompt the player when the game is won (all cards removed) or lost (stalemate is reached).<br>**L4.** The game is able to provide a hint to the player (on request) suggesting a valid move or confirming that no move is possible. | **L5.** On completion of a game (whether the player has won or lost) the application is able to replay the game move by move, with the user prompting each replayed move by a keypress.<br>**L6.** The application is able to play a complete game in "demonstration mode", where the only input from the player is to prompt the application to play its next move. The computer should provide a commentary on each move made. | |
| Testing (20 marks) | No evidence of testing or no evidence that testing has been effective. | Details of basic testing are present but no evidence of a structured approach. Limited documentation of the testing process. | Some test classes have been provided, but with less than full coverage. Incomplete testing documentation. | All main classes have test classes that give proper attention to the range of potential input values including edge cases. A comprehensive write-up that fully documents the testing process. | |
| Video (10 marks) | No video or no voiceover. | Video is present but is short or is missing either a demonstration of the application or a walkthrough of the main code elements. | Video demonstrates functionality and discusses the main code elements – but does not provide any insight beyond what can be seen by running the software. | Video provides additional insight into the development process. Excellent demonstration of working functionality. | |
| | | Initial Mark: | Deductions: | Total Mark: | |