**Higher Diploma in Science in Computing**

**Software Development – Problem Based Learning Project (20%)**

**Team members**

|  |  |
| --- | --- |
| ***Student name*** | ***Student number*** |
| Alan O'Neill | 16134427 |
|  |  |
| Jonas Stein | 16136306 |
| Mark Hanlon | 16135571 |

**Problem Description**

“Morra” is a hand game usually played for entertainment or to settle a disagreement. The game has many variations and can be played by two or more players.

**Morra Odds and Evens Variation**

In this variation of the game, one player is going to be the “Odds” player and the other player is the “Evens” player. In each round of the game, the players will simultaneously hold out between 1 and 10 ﬁngers. The winner of the round is decided based on the sum of ﬁngers shown by both players, namely if the sum is an even number then the “Evens” player wins, otherwise if the sum is an odd number then the “Odds” player wins. The winner of the round receives two points. In addition, the player whose number of fingers is closer to the sum, receives one extra point.

The winner of the game is the ﬁrst player who accumulates six points.

Develop an application to allow a user to play repeatedly the game “Morra Odds and Evens” with a computer. At the beginning of each game the user will be prompted to choose whether he/she would like to be the “Odds” or “Even” player. In each round of the game the user must decide the number of ﬁngers to show (i.e. between 1 and 10). Similarly, in each round of the game the computer will randomly pick one number between 1 and 10. In each round, the game displays the computer’s choice. After each round the game displays the number of points each player has, and whether the user or the computer won the round.

A game ﬁnishes when one of the players accumulates 6 points. At the end of a game, the game displays who the winner is, and a history of the numbers of ﬁngers shown by both the user and the computer per round. Once a game has ﬁnished the application asks the player if he/she would like to play another game. At the end of all games, display a history of games played. The history shows for each game the number of rounds won and lost by the human player, and how many even and odd numbers have been chosen by each player, and the extra points received by the player per game. All the history elements of the game should be coded using Arrays.

**Application Development:**

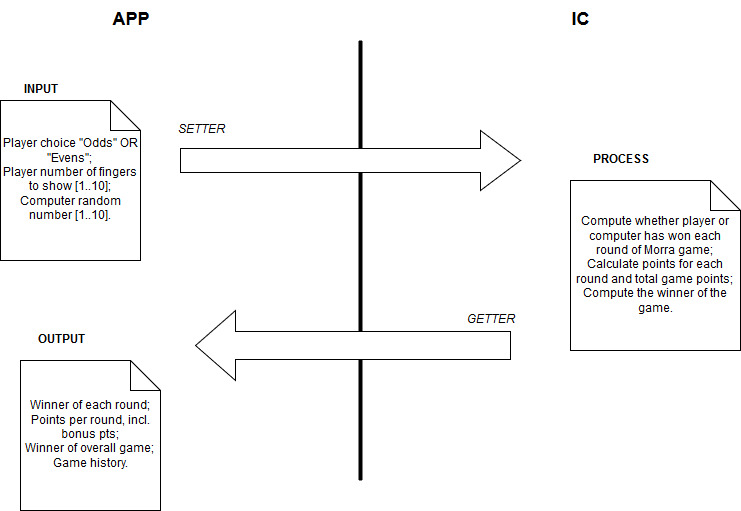
To solve this problem, you shall form a group of 3 people.

**Project Submission:**

* The project submission should include:
  1. A project report detailing who has done what in the code. It should be very clear who has coded which part of the code. The code should be commented to explain what the code is doing. As part of the report, please specify the Input, process and output.

**Project report**

***Input, process and output:***



***Code development:***Having examined the description and worked on the IPO, our team determined that to solve this problem of the Morra game required the use of several Java concepts. These included use of select statements (if/else/switch); repetition statements (for/while loops); and Arrays.

We decided that we would develop three instantiable classes (Morra, Game, NoGenerator) along with the main method class (MorraApp), hence implementing many examples of getter & setter methods and constructors. Also making use from Java of the Scanner class (for user input) and Random class (to generate random numbers).

Note, we designed our game using multiple instantiable classes of which we defined a single-dimensional array of objects, we did not use multi-dimensional arrays.

Coding work was divided up amongst the team as follows:

**JONAS -**

Game.java   
(Instantiable class that implements Player and Computer game stats and features)

NoGenerator.java  
(Instantiable class to generate random number to be used by Computer in Morra game)

**ALAN -**  
Morra.java

(Instantiable class that interfaces with the user in a game)

**MARK -**

MorraApp.java  
(Main class that interacts with the user when initiating the game)  
  
However we ensured that all of the team collaborated in real time to maintain uniformity between names of variables, methods, etc. and each team member shared any findings they had which could benefit the overall solution.

***Additional comments:***

The team realised that the requirements made no provision for two game scenarios which can occur:

a) There is a draw when both players accumulate 6 points at the same time.  
b) Both players select the same number of fingers in a round, so both are eligible for bonus point.  
  
We have made enhancements to our solution code to resolve both of these scenarios.