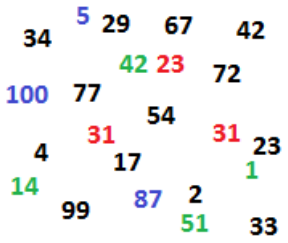
Gue55ing Game



Introduction

This assignment is due by 11pm Friday of Week 7 (20th April, 2018). It is worth 15% of the marks for your final assessment in this unit. **Heavy penalties will apply for late submission.** This is an individual assignment and must be your own work. You must attribute the source of any part of your code which you have not written yourself. Please note the section on plagiarism in this document.

The assignment must be done using the BlueJ environment.

The Java source code for this assignment must be implemented according to the **Java Coding Standards for this unit.**

Any points needing clarification may be discussed with your tutor in the lab classes.

Specification

For this assignment you will write a program which will allow a person to play a number guessing game against the computer. This section specifies the required functionality of the program. **Only a text interface is required for this program**; however, more marks will be gained for a game that is easy to follow with clear information/error messages to the player.

The aim of *Gue55ing Game* is for a person and the computer to compete against other to correctly guess a hidden number.

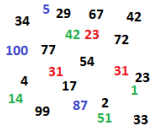
A game consists of four rounds. For each round a number between 1 and 100 (inclusive) is randomly generated and the players (person and computer) take turns to guess the number. The round ends when the correct guess is given or each player has had three guesses.

If a player guesses the number correctly then they are awarded points according to how many attempts were taken to guess the number. If the round ends without either player guessing correctly then the points are awarded to each player according to how close they were to the hidden number.

At the end of the four rounds the player with the highest cumulative score wins the game.

Game play

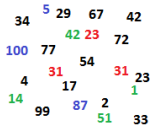
The *Gue55ing Game* game begins with a message inviting the human player to enter their name. The name can only contain characters (except all spaces, including leading and trailing spaces) and must be between 1 and 8 characters in length (inclusive). The other player will be the computer. A number with a value from 1 to 100 (inclusive) is randomly generated but hidden from the players. The player who will have the first turn at guessing the number is then randomly chosen by the computer. The round then progresses with the players taking turns until the correct number is guessed. Note that your program will generate a number guess for the computer player.



The following are the game rules:

- If the number is not guessed correctly then a message is displayed indicating whether the entered number was higher or lower than the hidden number and the other player takes a turn. Note this means that after each turn the range of possible numbers is reduced. For example:
    - When the game starts the original range is 1 – 100. The hidden number generated is 75
    - First turn, the player guesses 50, this is lower. So now the new range for the computer to guess is between 51 and 100.
    - Second turn, the other player guesses 90, this is higher. So now the new range for the player to guess is between 51 and 89.
    - This goes on for each guess.
- If the player enters a number between 1 and 100 but not within the range of possible numbers (as explained above), then the player is given a warning message but is not given a chance to renter the number.
- If the player enters a number less than 1 or greater than 100, then a warning message is displayed and the player is invited to enter another number (with no penalty).
- If the player enters non-numeric characters, then a warning message is displayed and the player is invited to enter another number (with no penalty).
- If the human player enters 999 this indicates that they have decided to abandon the round. To simulate this for the computer player, at the start of each round, generate a random number between 1 and 20. This number will be known as the ‘abandon round indicator’. Then before each guess the computer player makes, generate a random number between 1 and 20. If this number is the same as the ‘abandon round indicator’ then the round is abandoned. If these numbers are not the same, the computer continues with its guess of the hidden number.
    - Abandoning a round means no scores are assigned to either player for that round.
- If a player correctly guesses the number then the round ends, points are awarded to this player according to how many attempts have been made. Note the total number of attempts includes attempts by both players. The other player scores zero for the game.

Number of attempts	Score
1	20
2	15
3	11
4	8
5	6
6	5



- If the round ends and no player has guessed the number then each player is awarded a score according to the proximity of their last guess to the hidden number, as follows:

  score = 10 – proximity-of-last-guess
  (Note if proximity-of-last-guess is 10 or more then the score is 0 (zero))

  For example, if the hidden number was 63 and the last guess for player1 was 53 and the last guess for player2 was 68, then player1 would score 0 (zero) and player2 would score 5.

  Note: the 'last guess' here is referring to the third guess of each player in that particular round.

Program design

Your program should consist of at least three classes: Player, Game and RandomNumber. The following two sections give details of these classes. Students are advised to follow good programming practices and to use loops and appropriate field where required to ensure good program design.

Player class

The Player class will specify the attributes and behaviours of a player. An object of the Player class will have the following fields (at least):

Name – the name of the player.

Score – the cumulative game score

Guesses – the last number guessed for the current round

The data type of each field must be chosen carefully and you must be able to justify the choice of the data type of the fields. You may want to include comments in the class to state the assumption made. The class must also have a default constructor and a non-default constructor that accepts a value for the name of the player.

Player class no longer needs validations

The Player class should also have *appropriate* accessor and mutator methods for its fields. You should not allow an object of class Player to be set to an invalid state. There should be no input from the terminal or output to the screen. A Player object should also be able to return its state in the form of a String.

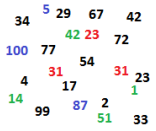
Game class

The Game class will be in the same BlueJ project that contains your Player class. The Game class will manage the playing of a game. It will have the following fields (at least):

Player1 (an object of type Player)

Player2 (an object of type Player)

Note that one of these players will be the computer.



The Game class will have methods to manage the playing of the game. These should include (at least) the following behaviours:

- Display a welcome message on the screen.
- Request the player to enter their name.
- Request the player to enter a number.
- Compare the number entered by a player with the hidden number.
- Display the result of the attempt at guessing the number.
- Display the result for the end of a round (including the value of the hidden number).
- Display the game result.

RandomNumber class

An object of the RandomNumber class will generate a random number from 1 to a maximum value specified.

Assessment

Assessment for this assignment will be done via an interview with your tutor. The marks will be allocated as follows:

- 30% - Object-oriented design quality. This will be assessed on appropriate implementation of classes, fields, constructors, methods and validation of the object's state.
- 10% - Adherence to FIT9131 Java coding standards.
- 60% - Program functionality in accordance to the requirements.

You must submit your work by the submission deadline on the due date (a late penalty of 20% per day, inclusive of weekends, of the possible marks will apply - up to a maximum of 100%). There will be no extensions - so start working on it early.

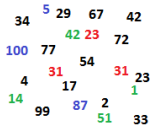
Marks will be deducted for untidy/incomplete submissions, and non-conformances to the FIT9131 Java Coding Standards.

Interview

You will be asked to demonstrate your program at an "interview" following the submission date. At the interview, you will be asked to explain your code/design, modify your code, and discuss your design decisions and alternatives. Marks will not be awarded for any section of code/design/functionality that you cannot explain satisfactorily (the marker may also delete excessive in-code comments before you are asked to explain that code).

In other words, you will be assessed on your understanding of the code, and not on the actual code itself.

For **on-campus students**, interview times will be arranged in the tutorial labs in Week 7. It is your responsibility to attend the lab and arrange an interview time with your tutor. **Any student who does**



not attend an interview will receive a mark of 0 for the assignment. The actual interviews will take place during the normal lab times in Week 8.

For **off-campus learning (OCL)** students, the interviews will be organised during week 7 and will take place online via Skype or other video facility during Week 8. It is your responsibility to make yourself available for an interview time. **Any student who does not attend an interview will receive a mark of 0 for the assignment.**

Submission Requirements

The assignment must be uploaded to Moodle by 11pm Friday of Week 7 (20th April, 2018).

The submission requirements for Assignment 1 are as follows:

A .zip file uploaded to Moodle containing the following components:

- the BlueJ project you created to implement your assignment.
- a completed **Assignment Cover Sheet**. This will be available for download from the unit's Moodle site before the submission deadline. You simply complete the editable sections of the document, save it, and include it in your .zip file for submission.

The .zip file should be named with your Student ID Number. For example, if your id is 12345678, then the file should be named 12345678_A1.zip. Do not name your zip file with any other name.

It is your responsibility to check that your ZIP file contains all the correct files, and is not corrupted, before you submit it. If you tutor cannot open your zip file, or if it does not contain the correct files, you will not be assessed.

Marks will be deducted for any of these requirements that are not complied with.

Warning: there will be no extensions to the due date. Any late submission will incur the 20% per day penalty. It is strongly suggested that you submit the assignment well before the deadline, in case there are some unexpected complications on the day (e.g. interruptions to your home internet connection).

Plagiarism

Cheating and plagiarism are viewed as serious offences. In cases where cheating has been confirmed, students have been severely penalised, from losing all marks for an assignment, to facing disciplinary action at the Faculty level. Monash has several policies in relation to these offences and it is your responsibility to acquaint yourself with these.

Plagiarism (http://www.policy.monash.edu/policy-bank/academic/education/conduct/plagiarism-policy.html)