CYPRUS INTERNATIONAL UNIVERSITY
FACULTY of ENGINEERING Fall 2020-2021
CMPE501 – (Advanced) Programming Languages
PROJECT – Random Letter guessing game

Given Date: 03.11.2020 Tuesday
Submission Deadline: 01.12.2020 Tuesday

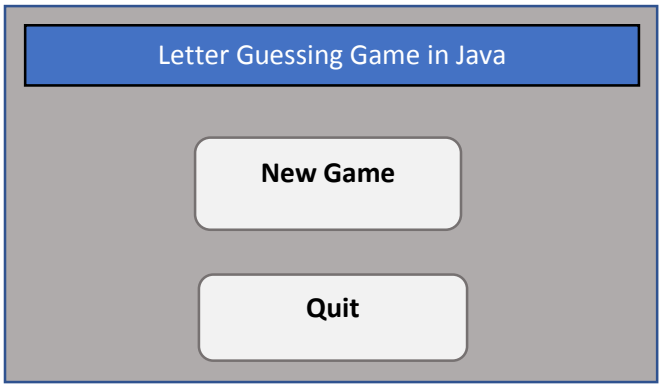
Your submission must be on via Moodle (emails are not accepted).
Weight: %20

Random Letter Guessing Game in Java

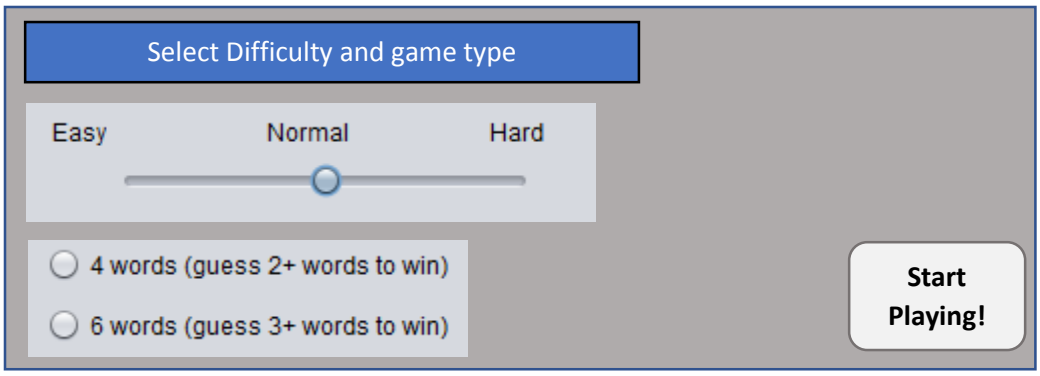
In this assignment you are asked to develop a letter guess game in Java which would select words randomly from a txt file within the project directory. The layout of the game is given below, and you will need to follow this layout to create the game. You are however, encouraged to go above and beyond the basic requirements of the project so that your knowledge can improve.

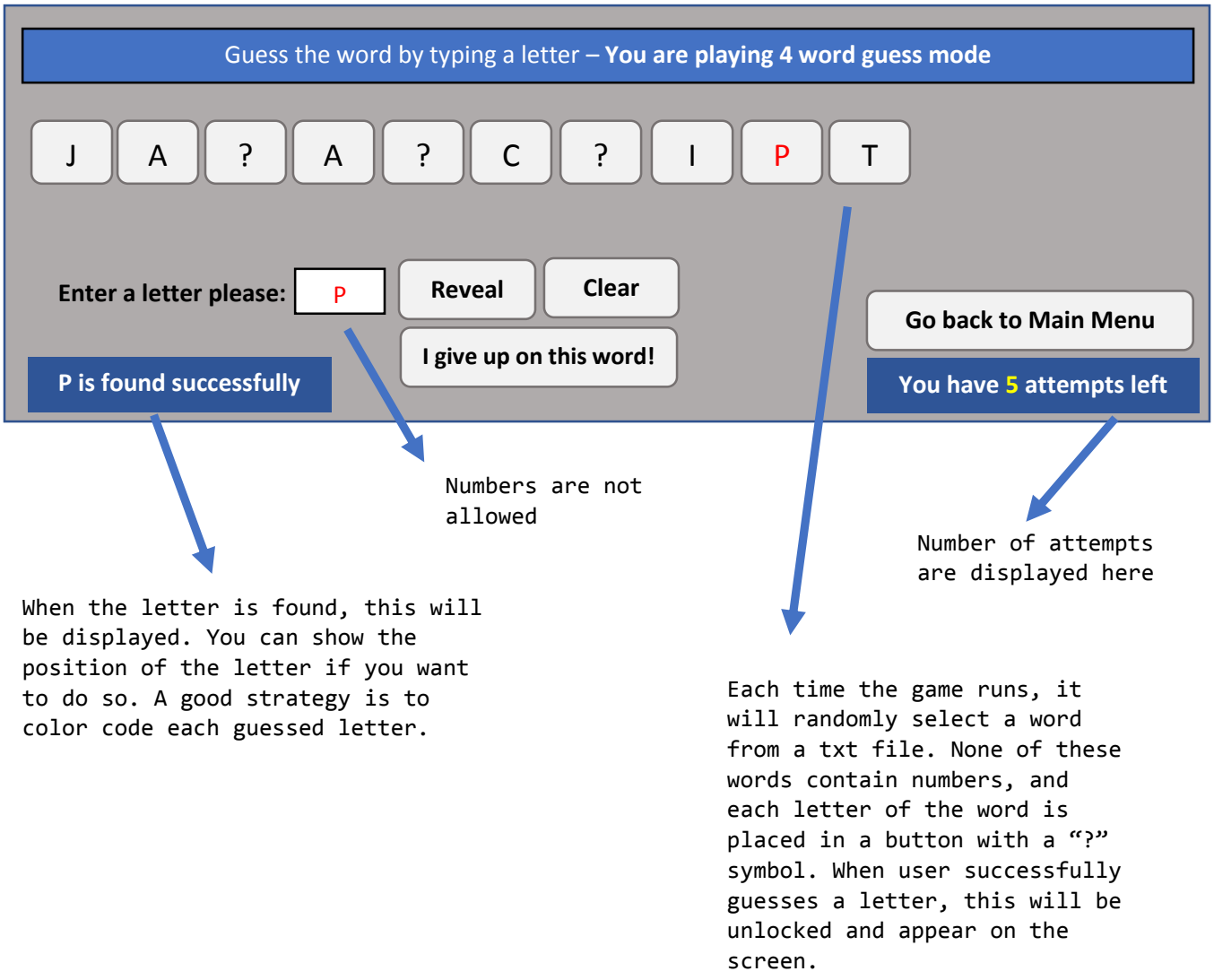
The Game

When the application runs, the game screen will offer two main options. These are new Game – which would start a new game, and quit option which would end the application.

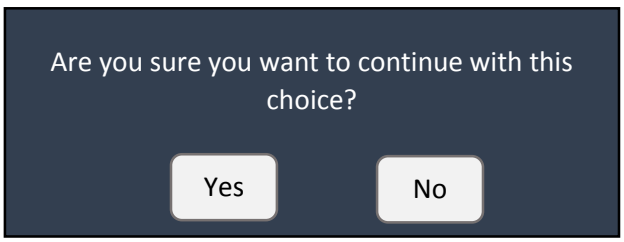


When user selects the New Game option, the next screen will allow a difficulty selection and how long the game is going to run. Followings are the options offered to the user.





The general layout of the game is displayed above. As you can see from the above figure, the users can type a letter and press the reveal button to play the game. The textArea only allows one letter entry in each attempt and it does not allow numbers to be typed (i.e. these should be validated). The clear button clears the entry user typed and also removes the highlight on the last letter. Additionally, when user do not want to play, they can click on **I give up on this word!** button, and this reveals the entire word asked and allows the user to move on to the next word. Please be aware that this button only this word, so they can still continue playing the game. At any point, the user can return back to main menu by clicking on **Go back to Main Menu** option. When user clicks on either **I give up on this word!** button or on **Go back to Main Menu**, a message will be displayed to user to ensure their choice as they could have clicked on any of these buttons accidentally:



The Easy, Normal and Hard options decide how many letters are revealed on the word when the game starts.

In **easy mode**, approximately half of the letters (i.e. 1/2) are revealed at the beginning of the game.

In **normal mode**, approximately one third of the letters (1/3) of the letters are revealed, and

In **hard mode**, only 1 letter is revealed in each word at the beginning of the game.

Similarly, the game can be played with 4 guess or with 6 guesses. When user successfully guesses half of the words correctly, they win the game. The difficulty choice does not affect this outcome.

The words are read from a txt file within the project directly. There should be at least 100 words stored in this file and they are randomly selected each time a new game starts. You can store each letter to an array or to a collection in order to manage the letter guessing. Please make sure that you store each word in a collection. Defend why you used this collection instead of another one, such as if you had used an arrayList, describe why you did not use a LinkList or a Vector.

At the end of the game, if the user successfully guessed half of the words, they will win the game. If they do not, the game is over, and they can replay again.

What to submit? How to do the work?

This is a team project, and you must complete this assignment in pairs (working with someone else who is also taking this course). It is your responsibility to find your pair, and please do not send messages such as "*I cannot find someone to work with*" or "*I want to do this project alone*" because you cannot complete this assignment individually – you must work with someone in the class as software development is a team work.

There are no strictly defined roles for the project so you can decide who is going to work on what, however, be aware that the project must be fully object oriented in a tiered software architecture. This means that there should be a clear separation between the user interface and the business logic. If this is not done, your work will be heavily panelized.

At the end of the work, you must submit: the **source files**, the **runnable .JAR file** – which can be clicked and directly runs, and a **report**. Please create a folder for each of these within your submission which should be a **.ZIP file** (not .RAR, .ZIP!). The report format is uploaded separately, and it should contain name, surname and number of all participants who worked in the project.

Going above and beyond…

You are strongly encouraged to go above and beyond in this project by applying new and modern programming methods, constructs and technologies. Below you will find a series of **innovations** you can consider to improve the fundamental requirements of this project. While **you are under no obligation to complete all of the innovations listed below, you must add at least 1 of them**.

1. Implement a high score system using a JSON file. You can receive a nick name from the user and store this in a JSON file with their score. Add a High Score List button on the main menu, and allow user to view their high score.

2. Instead of a txt file, use an editable XML file to store the words. You can create an admin mode for the game to add new words or even remove existing ones.

3. Create a Cheat Mode in the game. Such as when user enters a certain character (e.g. # sign), the word is revealed on a separate label – making it very easy for the user to guess the word and win the game.

4. Use a timer for each segment and enforce the player to guess the word within a time limit. As an example, you can put 5 mins time limit for guessing each word. You can use a thread to put the timer and if time runs out, the player loses that word.

Last but not least, please make sure to submit the project on the submission date latest by 23:55. If you miss this deadline, you will not be allowed to submit and you will lose the %20 weight of your overall mark.

GOOD LUCK