Project 1: Guess The Movie   Dated: 03.03.2020

Due Date: 20.03.2020

Mark : 10 for each Lab and Theory In Semester Examination

Ok, it's time to build your own project in Java, this time you'll be completing a game where the player gets to guess the movie name given the number of letters in it (pretty much like hangman but with movies)!

The rules are simple, the computer randomly picks a movie title, and shows you how many letters it's made up of. Your goal is to try to figure out the movie by guessing one letter at a time.

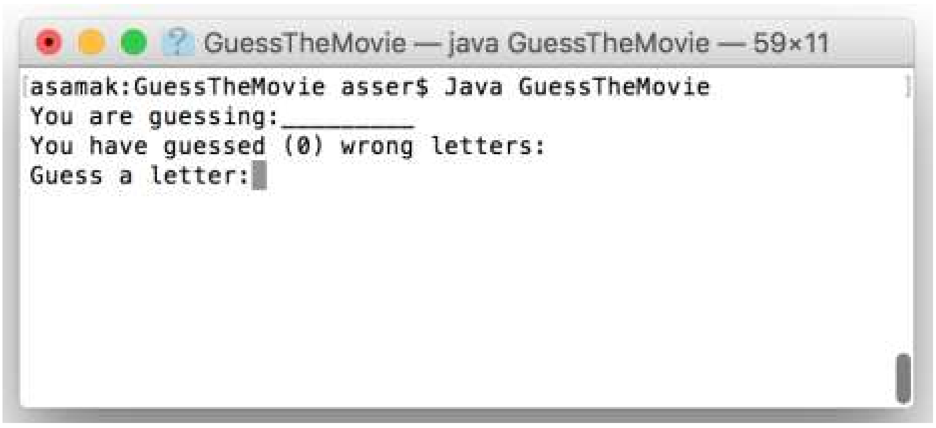
If a letter is indeed in the title the computer will reveal its correct position in the word, if not, you lose a point. If you lose 10 points, game over!

BUT the more correct letters you guess the more obvious the movie becomes and at a certain point you should be able to figure it out.

The program will randomly pick a movie title from a text file that contains a large list of movies.

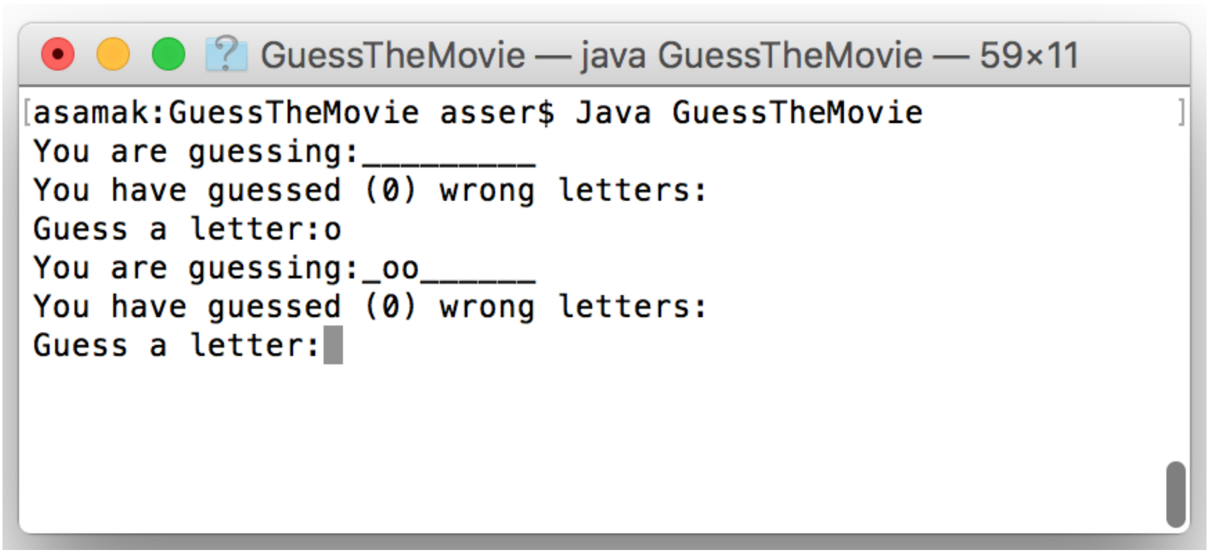
You can download a sample text file to play with from the resources tab or create your own list of movie titles.

Once the computer picks a random title, it will display underscores "_" in place of the real letters, thereby only giving away the number of letters in the movie title.

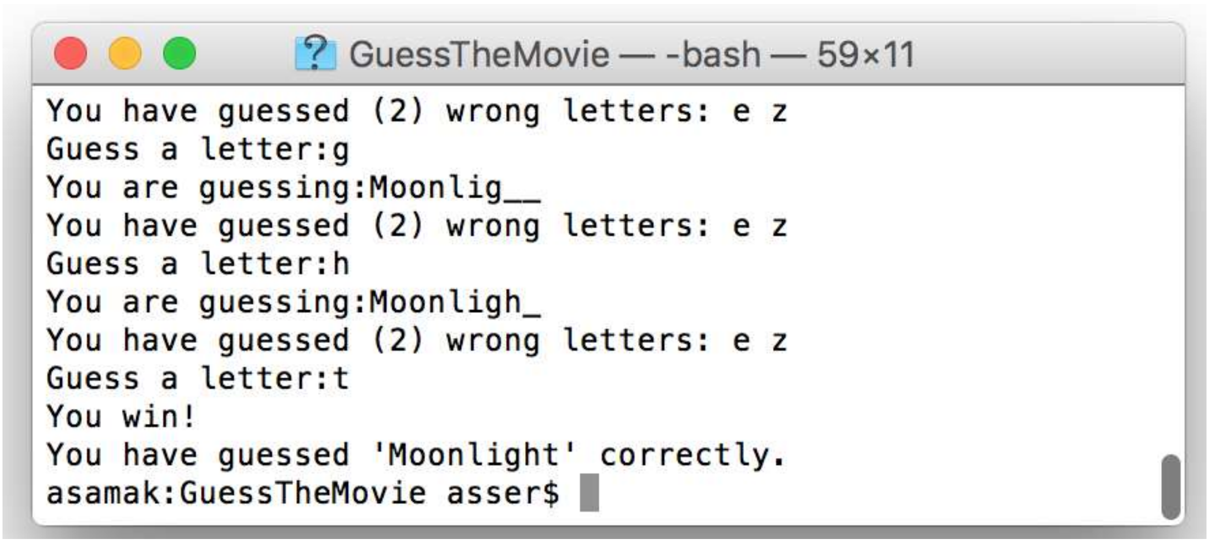

```
asamak:GuessTheMovie asser$ Java GuessTheMovie
You are guessing:_________
You have guessed (0) wrong letters:
Guess a letter:
```

Then it will wait for the player to enter their first letter guess.

If the letter was indeed in the word, the underscores "_" that match that letter will be replaced with the correct letter revealing how many letters have matched their guess and where they are.


```
asamak:GuessTheMovie asser$ Java GuessTheMovie
You are guessing:_________
You have guessed (0) wrong letters:
Guess a letter:o
You are guessing:_oo______
You have guessed (0) wrong letters:
Guess a letter:
```

Eventually, if the player manages to guess all the letters in the movie title correctly before they lost 10 points, they win.



```
You have guessed (2) wrong letters: e z
Guess a letter:g
You are guessing:Moonlig__
You have guessed (2) wrong letters: e z
Guess a letter:h
You are guessing:Moonligh_
You have guessed (2) wrong letters: e z
Guess a letter:t
You win!
You have guessed 'Moonlight' correctly.
asamak:GuessTheMovie asser$
```

Everything you need to know to be able to build this game should be covered in the previous lessons, but of course that doesn't mean it has to be easy! It's ok to get stuck and it's absolutely normal for things to not work from the first time.

Just take it step by step, build a small part of the game first, test it and make sure it works and then continue to add more to it.

Download the movie list from here, and start coding. Good luck :)

Hints to help building Guess The Movie game

Reading File by java program:

File Scanner

Another way of accepting runtime input is through files, these files can be plain text files that the user creates with a very basic text editor (e.g. notepad on windows or TextEdit on macs).

A good example would be a Java program that loads a list of expenses from a text file (or excel sheet) and after some calculations prints a report of the total amount, average spendings, largest purchase etc.

To read a text file in Java you can also use the same `Scanner` class we used to read command line inputs, but instead of passing `System.in` as the argument you pass a `File` object that you can create by typing in the file name:

```
File file = new File("expenses.txt");
Scanner fileScanner = new Scanner(file);
```

Once the file scanner has been created, you read lines the same way we did earlier.

But since you would most likely want to load the entire file at once, you can check if the file still has more lines using `hasNextLine` method and then use this loop to read everything:

```
while (input.hasNextLine()) {
   String line = input.nextLine();
   // Use that line to do any calculations, processing, etc ..
}
```

For other method, you can check here.

Game play hints

In English, the top 5 frequency of letters is `e` `t` `a` `o` `i`. It can help you play this game after you finish it.

It's an important study in Cryptanalysis. More info about this, please read Letter frequency from Wikipedia.

Use classes

This program will have more code than all of the exercises we have previously covered, so it is a good idea to divide your code into classes instead of writing everything in 1 class.

A simple design would be to have at least one more class called `Game` that will include methods responsible for handling a single guess or displaying the hidden movie title etc.

Then have another class that contains the main method and controls the logic of reading the user's input and calling the methods in the `Game` class

Build it step by step

Don't rush into building the entire game at once, start small, for example:

1. Write some code that will simply read the movie list and display the whole list.
2. Then add to your code to randomly pick one movie and display that title only.
3. Then convert its letters to underscores (`_`) and display that instead, and so on.
4. Once you've got that part done start reading the user's input and search for it in the title.
5. Work on revealing the correct letters and displaying them.
6. Add the logic to keep track of wrong letters so they don't lose points for guessing the same letter twice.
7. After that, you can keep track of how many wrong guesses and end the game if they lose.
8. Finally, detect when they have guessed all the letters and let them know they've won!

You can also start by hard coding a single movie title in the code instead of randomly picking one from the file, then add the file reading functionality at the end.

Test often

Every time you add new code that does something new, test it.

The best way to do so is to use `System.out.println()` everytime you add new functionality to test the output of that part.
Make sure when testing to try all possible cases that you can think of (what if the user tries to guess a space character? what if they type in a number? etc)

If you test often while building your code you will end up with fewer bugs as you get closer to finishing it.

String methods

Check out all the powerful methods that Java has already written for you here. Knowing the capabilities of your programming language can save you hours and even days of re-writing code that already exists.

For example:

To find if a letter exists in a String, instead of creating a loop to compare each character you can use the `indexOf()` method which returns the position of such character in the String.