**Project I - Guess The City**

**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 city name given the number of letters in it (pretty much like hangman but with cities)!**

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

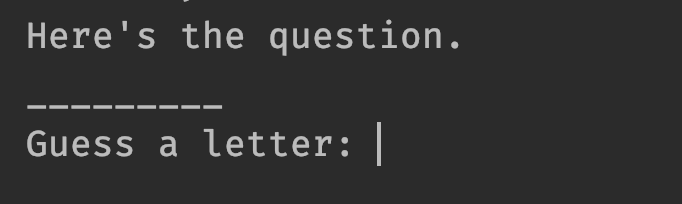
**If a letter is indeed in the city name the computer will reveal its correct position in the word, if not, you lose a point. If you lose 10 points, game over!**

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

**The program will randomly pick a city from a text file that contains a large list of cities.**

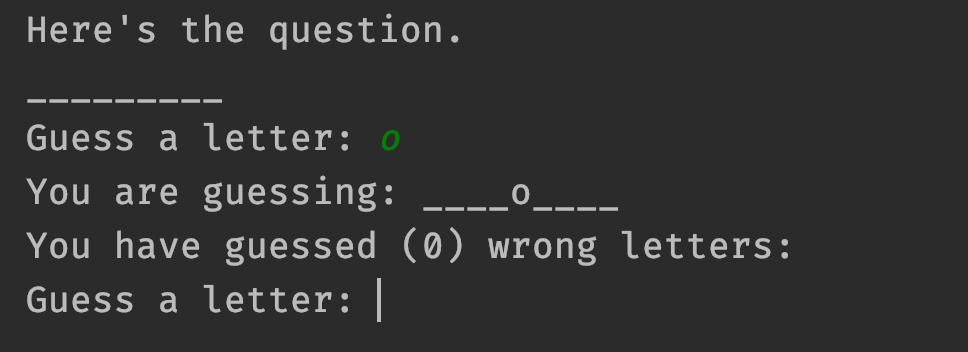
**You can download a sample text file to play with from the resources tab or create your own list of cities.**

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

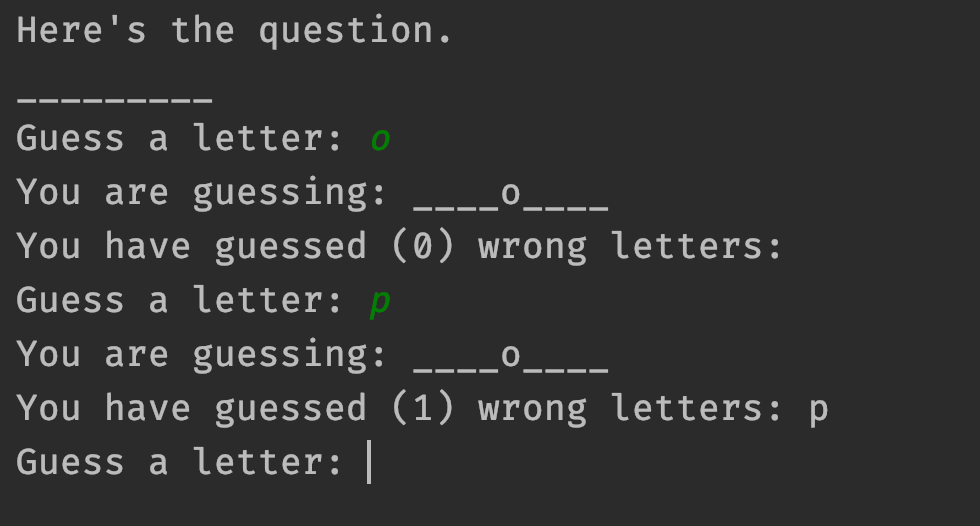
****

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

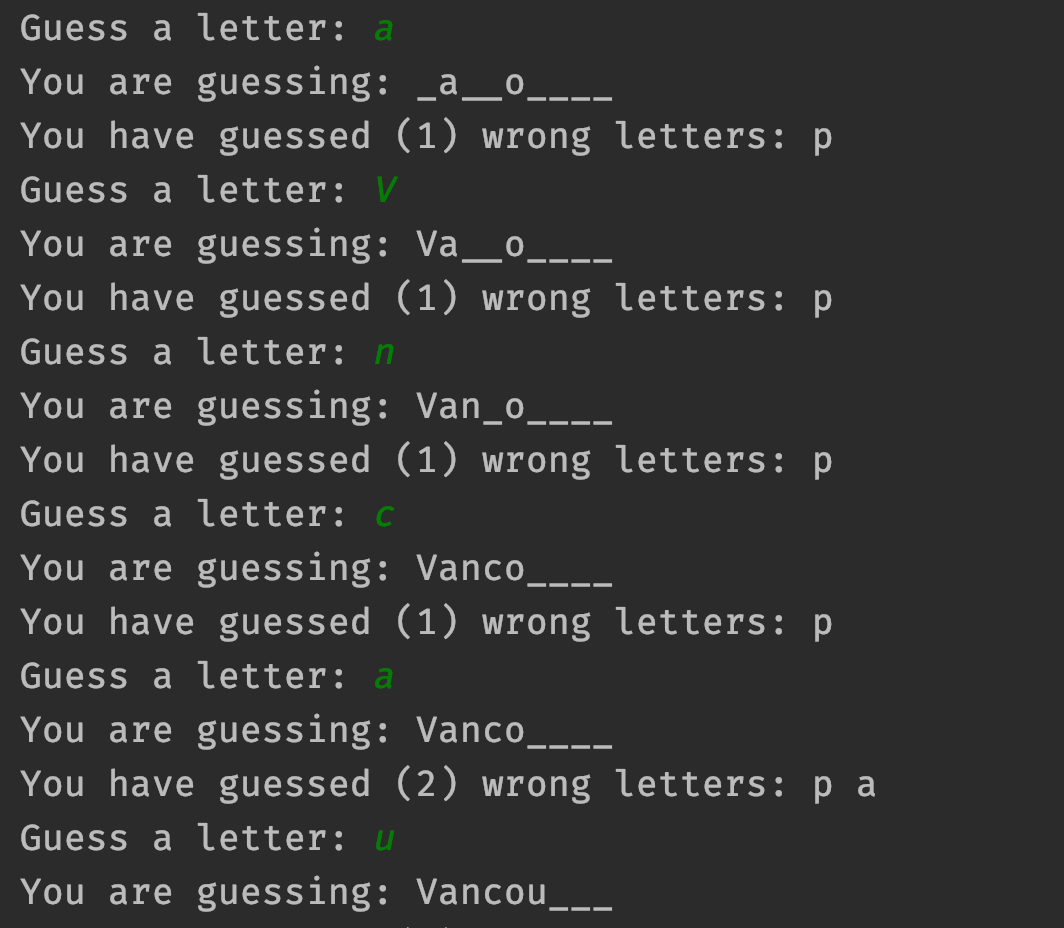
**If the letter was indeed in the word, the underscores "\_" that match that letter will be replaced with the correct letter revealing how many letters have matched their guess and where they are.**

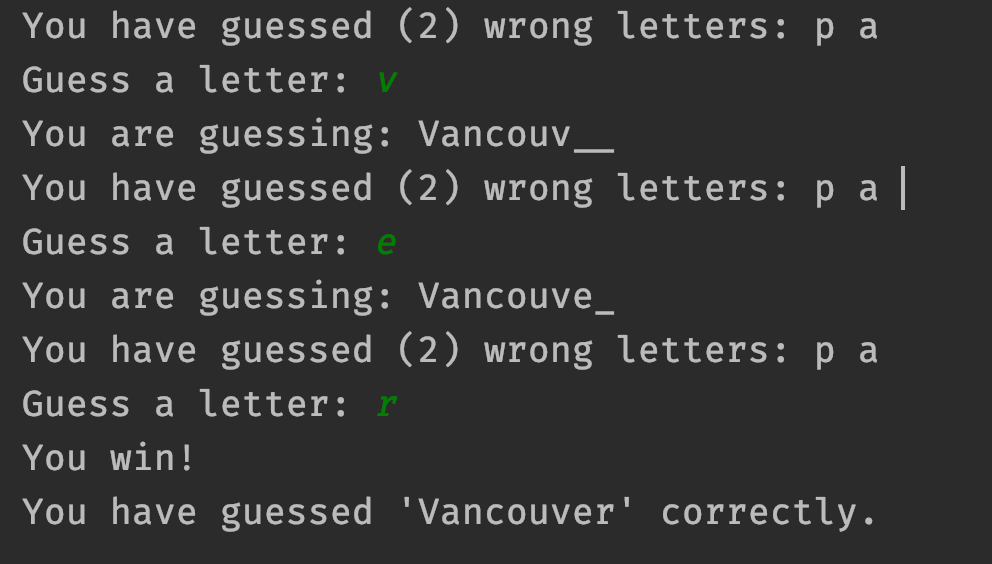
****

**If their guess isn't in the city, then the output will display the same output as the previous step as well as any letters that have been previously guessed wrong.**

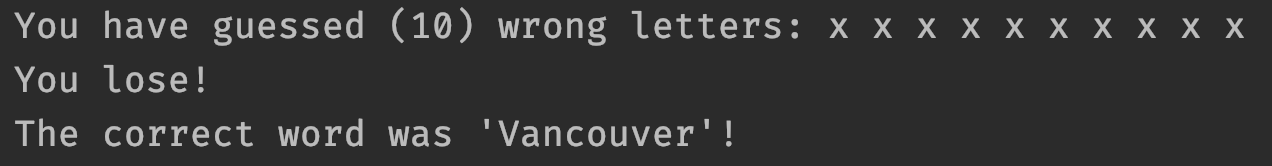
****

**Eventually, if the player manages to guess all the letters in the city correctly before they lost 10 points, they win (and the program terminates).**

****

****

**If the user fails to guess the city, then the following message will be printed out (terminate).**

****

**Everything you need to know to be able to build this game should be covered in the previous classes, 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.**

**Aaaand, whenever you reach a roadblock, head to the slack straight away, there are tons of other students and myself there that will be more than happy to help.**

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

**(You will be doing a small demo in front of the class!)**

# 

# **Hints to help building Guess The Movie game**

# **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**](https://en.wikipedia.org/wiki/Letter_frequency)**.**

## **Use classes**

**This program will have more code than all of the exercises/labs/assignments we've previously done, so it's 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 Question that will include methods responsible for handling a single question or displaying the hidden city name etc.**

**Then have another class that controls the logic of the game, reading the user's input and calling the methods in the Question class.**

**Lastly, you can create a Driver class that contains the main method to run your game.**

## 

## **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 city list and display the whole list.**
2. **Then add to your code to randomly pick one city 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 city.**
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 city 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**](https://docs.oracle.com/javase/8/docs/api/java/lang/String.html)**.**

**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.**

**Edge Cases**

// Mexico City

// \_\_\_i\_\_ \_i\_\_ (updates both i)

// invalid input: asdf - more than one letter (won't count)

// wrong char: \*, &, etc (add to wrong letter count)

// same answer char: (won't count, ask again)

// wrong (same char) : (add to wrong letter count)