ASSIGNMENT 1

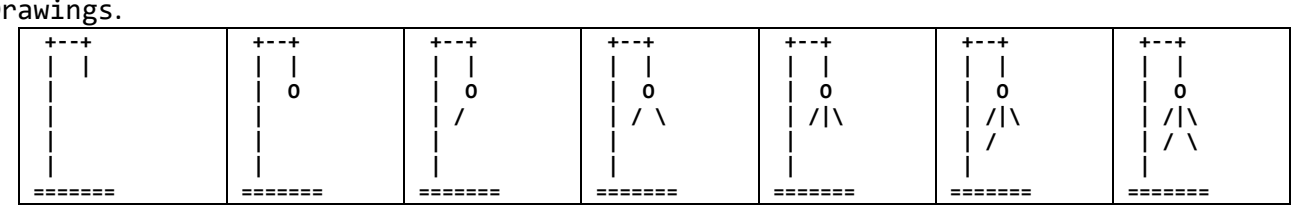
Assignment 1 tests your knowledge of Recursion (Chapter 18), List, Stacks, Queues, and Priority Queues (Chapter 20) and Sets and Maps (chapter 21).

Design a **program/project/driver class** (the program/project/driver class file should be called ***YourName*Assignment1**; replace ***YourName*** with your actual name), that is going to use lists, stacks, queue or priority lists, to implement a console version of the game Hangman.

Your users are going to guess US state capitals by entering one letter at a time. If the user misses 6 times, they lose the game. The user should have the option to give up the game or play again (on another word) until they give up. After each **guess letter attempt**, the program should display, the **guessed word** (the secret/hidden word with the letter guessed so far in it), say CORECT or WRONG if the attempted letter was correct or wrong and display the **Correct and Wrong letters** so far, and the ASCII **Drawing** of the current hangman. After the user wins or loses the game, the program should display a **final message** with the game status "You WON!" or "You LOST!" message. See examples on the second page.

The assignment is about using Recursion/recursive functions, lists, stacks, queue, priority lists, sets, and maps, thus, you should solve the problems using these concepts as much as possible and use them for at least for the following (exact names and functionality):

- Your program should have a *recursive function* call `DisplayWord` to display the guessed word by spelling out the word (separating the letters in the word by spaces), and, replacing empty spaces within the word (letter not guessed yet) with underscore. For example, if the word is "AUSTIN", display "A U S T I N", and, if the guessed word is "A T N" , display "A _ _ T _ N".
- Your program should have a 2 *sets or maps* called `CorrectLetters` and `WrongLetters` to keep track of the Correct letter guessed so fer and wrong/incorrect letter guessed so far.
- Your program should read the state capitals from the attached `ListCapitals.txt` file and store them into a *List* called `Capitals` and shuffle the list.
- Your program should store the 7 hangman drawings (bellow) into a *Stack, Queue, or Priority Queue* called `Drawings`.

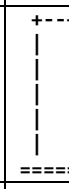
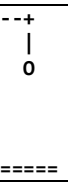
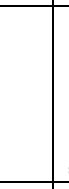
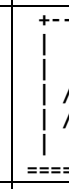


You can add additional methods or add your entire code inside the main method of your class.

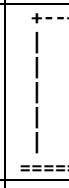
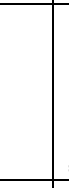
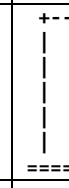
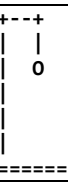
Create a Microsoft Word document called ***YourName*Assignment1-Screenshot**. (replace ***YourName*** with your actual name) that contains **screenshots** of the editor window showing the entire JAVA source code and the entire output. If the entire class JAVA source code or the output does not fit in one screenshot or the screenshots cannot be easily read, create multiple screenshots and add them to the same document.

Submit ***YourName*Assignment1.java JAVA source code file** and ***YourName*-Assignment1-Screenshots.docx Microsoft Word screenshots document** on eCampus under the **Assignment 1.** Do not archive the files (no ZIP, no RAR, etc) or submit other file formats.

Example1 (for secret/hidden word “AUSTIN”):

Guess letter attempt	Guess letter: A	Guess letter: B	Guess letter: C	Guess letter: D	Guess letter: E	Guess letter: F	Guess letter: G
Guessed word	A _ _ _ _ _	A _ _ _ _ _	A _ _ _ _ _	A _ _ _ _ _	A _ _ _ _ _	A _ _ _ _ _	A _ _ _ _ _
Validate	CORRECT!	WRONG!	WRONG!	WRONG!	WRONG!	WRONG!	WRONG!
Correct letters	Correct letters: A	Correct letters: A	Correct letters: A	Correct letters: A	Correct letters: A	Correct letters: A	Correct letters: A
Wrong letters	Wrong letters:	Wrong letters: B	Wrong letters: B C	Wrong letters: B C D	Wrong letters: B C D E	Wrong letters: B C D E	Wrong letters: B C D E F
Drawings							
Final Message							You LOST! The word was AUSTIN!

Example 2 (for secret/hidden word “AUSTIN”):

Guess letter attempt	Guess letter: A	Guess letter: I	Guess letter: U	Guess letter: E	Guess letter: N	Guess letter: S	Guess letter: T
Guessed word	A _ _ _ _ _	A _ _ _ I _	A U _ _ I _	A U _ _ I _	A U _ _ I N	A U S _ I N	A U S T I N
Validate	CORRECT!	CORRECT!	CORRECT!	WRONG!	CORRECT!	CORRECT!	CORRECT!
Correct letters	Correct letters: A	Correct letters: A I	Correct letters: A I U	Correct letters: A I U	Correct letters: A I U N	Correct letters: A I U N S	Correct letters: A I U N S T
Wrong letters	Wrong letters:	Wrong letters:	Wrong letters:	Wrong letters: E	Wrong letters: E	Wrong letters: E	Wrong letters: E
Drawings							
Final Message							You WON! The word was AUSTIN!