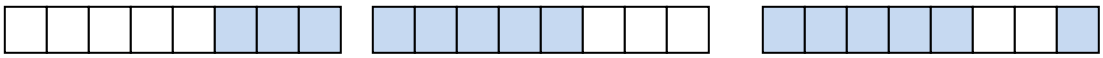
AP Computer Science

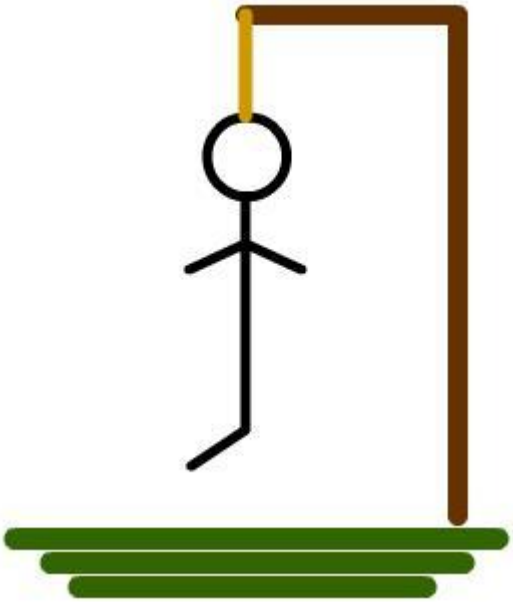
mr Hanley

Assignment 7/111_2/7_8/7_{16}

Binary **Ones Comp** **Twos Comp**



Hangman



1. The hangman game consists of a player attempting to guess the letters of a word before they miss too many guesses and lose. This assignment will involve creating a class called HangmanProcessor. HangmanProcessor will control the internals of keeping track of the word to be guessed, the display that is used to hide part of the word and the number of correct and incorrect guesses. Your class will also provide methods to determine if the player has solved the word or lost the game.

   You are required to write out the logic of the Hangman Processor on paper. When approved, you can go to the computer to key in your code.

   You must also create a HangTester class. This class should create a

HangManProcessor and a hard coded or typed in word and allow the user to play the game hangman. You don't have to draw a hangman in the console, just indicate when they have lost the game.

2. You are to create a Hangman Game using Swing. Your game must utilize a graphical window to allow the player to guess at a hidden word. Your game should utilize a list of words and present them in **random order**. Read the words in from a text file and store in an ArrayList. Make the words part of an overall theme, ie java terms, engineering terms, Clifton Park area restaurants, etc.

   Utilize an JLabel and Image Icons to display the hangman.
   Or use a JPanel and the paintComponent method
   Add a JTextField to allow the player to solve it quick

Project Name	**SwingHangman**
Class 1 Name	**HangApp**
Class 2 Name	**HangFrame**
Class 3 Name	**HangmanProcessor(Create this first with Hanley's Skeleton)**
Class 4 Name	**HangTester(A quick and dirty console tester, Should we use JUnit?)**
Textfile	**Terms.txt (Should be in project folder)**

Rubric	
Basic functionality	**50**
Reading words from a file(at least 25)	**20**
Random sequence (no repeats)	**20**
Solve it quick	**10**
Comments	**10**
TOTAL	**110**

Recursion*Linear Search*Binary Search*Grid World Case Study*File Processing *nlogn*Hangman