\*\*\*\* PART I: Project Planning (50 points) \*\*\*\*

You must submit a design documentation explaining your program's design and intended features. This will include notes of some kind to show your brainstorming process as well as UML documentation of your end result.

\*\*\*\* PART II: Programming (100 points) \*\*\*\*

You may choose one project on anything you'd like. As long as it has at minimum the following:

* 2 Classes
* Variable
* Struct or Enum
* Array
* Function
* Control structure (for loop, while loop, if statement, or switch statement)

Final Project Hangman

Description

Hangman is a game where a player guess letters in a word until the word is revealed. For each missed letter a part of a hanging man is showed. If the player completes the hanging man before the word is revealed, the player loses. Each letter has a value associated with it.

ToDo (Problem set)

1. 1. Read list of words from a file. There should be 3 files from which the words will be originated. Easy, Med, and Hard.
   * 1. Read a file and import the words and values to a map.
2. 2. Create a player class. This should be a base class the should two child classes are created; Player and computer.
   * 1. The player class:
        1. Keeps track of the letters the player has used.
        2. Keeps track of the players score
        3. Holds the players name.
        4. Holds the number of games the player had played.
3. Create Game Engine
   1. Prompts for player name
   2. Prompts for difficulty: Easy, Med, Hard
   3. Retrieves word
   4. create hangman
   5. Draws hangman's noose
   6. Output current score
   7. Output word under scores
   8. Prompts player for letter guess
   9. compares guess letter to letters in word
   10. If a letter is found, replace underscore with letter
   11. If a letter is not found draw a hangman part in this order:
       1. head
       2. torso
       3. left arm
       4. right arm
       5. left leg
       6. right leg

Classes needed:

* Player class
  + Child of Base player class
  + Holds number of guesses
  + Holds player lever (calculate level from score)
  + holds max attempts = 6
  + Holds player name
  + Holds player Score
  + Holds guessed letters
  + Holds incorrect guessed letters
  + Implements functions in IPlayer class.
* GameManager class
  + Gets word from Word manager
  + holds difficulty
  + holds player
  + uses Word management class
* File Management class
  + read word files (semicolon delimited list)
  + write player name and score (high score list, comma seperated)
  + read words into map
* Word management class
  + holds a file manager
  + reads in words and maps the letters to letter value (hold in map. Letter is the key and point value is the letter value)
  + read in file of words based on file with words that match difficulty
* Game Engine – main.cpp
  + Prompts for player name
  + Prompts for difficulty
  + Draw Hangman’s noose
  + Contains the game loop
  + holds max attempts (6)
  + make guess
  + display game status
  + determine if play has won the game
  + determine if player has lost the game