***Algorithm:***

~ **Class Connect 4 Gui**

* Set up the grid 6 by 7 with a 2D array
* Initialize the grid to 0 which is empty
* Have a boolean variable that determines who goes first (static)
  + If true then player 1 goes first
  + If false then player 2 goes first
* Have a random number generator that will randomly choose a number between 1 and 2
* Interface Game State

~ **Class Game State**

* Connect board in Game State
* Have static boolean variable in the connect 4 gui class in game state class
* Have static boolean variable in the win condition in the game state on when the game is finished
* Have a boolean variable that determine when the game is finished
* While loop that will stop when the boolean variable is true
  + Have a int variable that will add 1 each time the loop is run
  + Have the user enter a column they want their piece to go in.
    - If the boolean variable is true
      * Player 1 goes first
      * While the int variable < 6 && the piece under is empty which 0
        + If j == 5 || the piece under is not 0

Put 1 (player 1) in the spot

* + - * + Int variable +1
      * Boolean variable which to false
    - If the boolean variable is false
      * Player 2 goes first
      * While the int variable < 6 && the piece under is empty which 0
        + If j == 5 || the piece under is not 0

Put 2 (player 2) in the spot

* + - * + Int variable +1
      * Boolean variable which to true
  + Interface Win condition

~ **Class Win condition**

* Connect board into the win condition
* Have 2 int counter variables that will count the points
* Have a nested for loop that will check from the bottom left corner of the board
  + If 1(player 1) was found on the board
  + Go to the method that checks the neighbours of the piece (right, up, left up and right up)
  + If neighbour is found
    - A recursive method is called of the direction of the neighbour ex.( right - recursive method that keeps going right) recursion would stop if counter for player 1 = 4 || there is no neighbour next to the piece. Each time it finds a neighbour in the same direction counter +1.
  + If 2(player 2) was found on the board
  + Go to the method that checks the neighbours of the piece (right, up, left up and right up)
  + If neighbour is found
    - A recursive method is called of the direction of the neighbour ex.( right - recursive method that keeps going right) recursion would stop if counter for player 1 = 4 || there is no neighbour next to the piece. Each time it finds a neighbour in the same direction counter +1.
  + If counter ==1 then player 1 wins and boolean variable == true
  + If counter ==2 then player 2 wins and boolean variable == false

~ **AI easy Class**

* NOTE: (AI is player 2)
* Choses a random number. (1 -7)
* Interface Game State
* Change boolean variable to true

~**AI medium Class**

* NOTE: (AI is player 2)
* Check the win conditions on player 1
* If the counter ==3 for whatever direction
  + Place a piece blocking player 1 win
* Else
  + Choses a random number. (1 -7)
* Change boolean variable to true

~**AI Hard Class**

* NOTE: (AI is player 2)
* If the counter ==3 for whatever directions
  + Place a piece blocking player 1 win
* Else
  + Check the grid and determine the best piece to place in
* If counter2 == 3 for whatever directions
  + Place a piece to win the game.