Authors are David and Wanjiru.

Project Outline

Our project will focus on creating an algorithm that will solve a 9 by 9 sudoku puzzles, and we will do this through use of different classes and methods.
We will do this through the process of backtracking(reverting back to the previous steps, or solutions, as we determine that our current solution cannot be contained into a complete one).

Brief overview:

The program should be able to:

1. Pick an empty square
2. Try all numbers
3. Find one that works
4. Repeat
5. Backtrack

But first we need to make our program draw the sudoku in a more user friendly interface. We would need to draw vertical lines to make our sudoku more distinct. To do that, we'll create our **first** class: class SudokuBoard.
In that class, we'll have a constructor which would take in the parameters: self and board.
Our board would be a **nested list**, each inner list containing the **row** of the board.

```
#Pseudocode
Class SudokuBoard:
    def__init__(self, board):
        self. board = board
    def draw_horizontal():
        #we'll do a for loop of the rows of the board, then print an indented horizontal line
        # after every 3 rows. The code would be similar to:
        #  for i in range of len(board):
            # if i%3==0 and i!=0:
                #print(-------) #our indented horizontal line.
    def draw_vertical():
        # it would be similar to the horizontal only difference would be that we would pick the
        # first row and draw the | line after every 3 numbers.
        #for i in range of len(board)[0]:
            # if i%3==0 and i!=0:
                #print("|", end="") #our vertical line.

board1= SudokuBoard()  #creating a new instance of class

Class NumberSolution:
  def__init__(self,board, num,pos)
    self.board= board
```

```
      self.num = num
      self. pos = pos
    def find empty(self):
   #using both the for loop and if conditional statement to check if the number has 0
#if not it inserts a number, and returns a tuple.
   def valid(self, board, num,pos): #takes in the board number and position as parameters
#uses for loop to confirm that the number is not repeated in the row and column.
#we'll see detailed code later
```

Unit testing table (testing the methods in our class)

Test case number	Test case description	Test data	Expected result	Actual result	Pass/Fail

Class SudokuBoard					
1	Check that the function draws horizontal lines	draw_horizontal() function	horizontal lines drawn that divide the board into three rows which have three 9x9 grids each	-	-
2	Checks that the function draws vertical lines	draw_vertical() function	vertical lines drawn that divide the board into three columns which have three 9x9 grids each	-	-
3	Checks that the number of vertical lines are three	draw_vertical() function	The number of vertical lines are three	-	-
4	Checks that the number of horizontal lines are three	draw_horizontal() function	The number of horizontal lines are three		
5	Checks that the total number of 9x9 grids are 9	draw_horizontal() and draw_vertical() function	The total number of 9x9 grids is 9	-	-
Class NumberSolution					
6	Checks that the find empty function works (positive test)	find-empty() function	The number output is equal to the number of empty number spaces (where there is a zero) (passes)	-	-
7	Checks that the	validate_number() function	The numbers that are put in the	-	-

	valid function is working correctly and is able to put numbers in the correct place, that works for the whole board		finished board correspond to all of the rules of sudoku		
8	Checks that the find empty function works by running the program so the grid is filled so that the function returns zero (negative test)	find_empty() function	Output is zero because all the boxes are filled with numbers that aren't zeros (fails)	-	-
9	Checks that the validates number function works by manually putting a number in	validates_number() function	Function should identify that the number that was put is incorrect and that the board will not be able to be solved	-	-

	the wrong position in the initial grid (negative test)				
10	Checks that the validates number function and the find empty function work together by testing them on a filled out sudoku board	validates_number() and find_empty() functions	The validates number function should say that the all the numbers generated by the program are valid and correct and the find empty function should return zero as all the number spaces should be filled		