Brandon Tonge
ID - 201358937

COMP101 – Assessment 07

Python Code –

```
# 201358937 Tonge_Brandon-CA07.py
# December 2018
# This program is a simple 2 player game. Each player takes turns to choose a position on a 5
by 5
# grid. The winner is the first person to create a 2 by 2 grid on the game board.

import random

print("--- SQUARED ---")
print("Welcome to Squared, the tactical game where making a square makes you a winner!")
print("- Each player gets a turn to select a place in the grid between 1 and 25")
print("- The objective is to create a 2 by 2 square")
print("- The first player to do so is the winner!")
print("Don't worry about picking who goes first, the game will choose for you!")


def choose_player():

    # Get both the players names
    player1 = input(str("\nPlease enter player 1's name: ")).upper()
    player2 = input(str("Please input player 2's name: ")).upper()

    # Randomly select the first player (with an exception to my own name)
    player_selection = random.randint(1, 2)
    if (player1 == "BRANDON"):
        print(f"{player1} is first to go!")
        create_board(player1, player2)
    elif (player2 == "BRANDON"):
        print(f"{player2} is first to go!")
        create_board(player2, player1)
    elif (player_selection == 1):
        print(f"{player1} is first to go!")
        create_board(player1, player2)
    else:
        print(f"{player2} is first to go!")
        create_board(player2, player1)


def create_board(player1, player2):

    # Create list with 5 rows and columns
    numc = 5
    numr = 5

    list_2d = [[0 for row in range(numr)] for col in range(numc)]

    print("")
    for i in range(len(list_2d)):
        print(list_2d[i])

    # Call Game functions
    game(list_2d, player1, player2)


def game(list_2d, player1, player2):

    game_win = False
    game_draw = False

    while(game_win == False):

        # Run the Player 1 move and check for win and draw
        list_2d, position = player1_move(list_2d, player1)
        game_win = win_check(list_2d)
        if (game_win == True):
            print(f"\n{player1} is the winner!")
            break
        game_draw = draw_check(list_2d)
        if (game_draw == True):
            print("\nAll the positions in the grid are full. The game is a draw!")
            break

        # Run Player 2 move and check for win and draw
        list_2d, position = player2_move(list_2d, player2)
        game_win = win_check(list_2d)
        if (game_win == True):
            print(f"\n{player2} is the winner!")
            break
        game_draw = draw_check(list_2d)
        if (game_draw == True):
            print("\nAll the positions in the grid are full. The game is a draw!")
            break

    # Ask if player wants to play again
    while True:
        play_again = input(str(("\nWould you like to play again? Y/N : "))).upper()
        if(play_again == "Y"):
            create_board(player2, player1)
        elif(play_again == "N"):
            print("Thank you for playing!")
            exit()
        else:
            print("Please enter a valid choice!")
            continue


def player1_move(list_2d, player1):

    # Get Player 1 position and validate
    while True:
```

Brandon Tonge
ID - 201358937

```
        position = input(f"\n{player1}, choose your position in the list: ")
        try:
            position = int(position)
            if (position < 1 or position > 25):
                print("Please enter a value between 1 and 25!")
                continue
            else:
                break
        except:
            print("Please enter a valid number!")
            continue

    # Convert to a 1d list
    list_1d = convert_list_1d(list_2d)

    # Check if the position entered is already full
    while(list_1d[position - 1] != 0):
        print("\nPlease enter an empty position!")
        position = int(input(f"\n{player1}, choose your position in the list: "))

    # Add a one to the players selected position
    list_1d[position - 1] = 1

    # Convert to a 2d list
    list_2d = convert_list_2d(list_1d)

    # Return values
    return(list_2d, position)


def player2_move(list_2d, player2):

    # Get Player 2 position and validate
    while True:
        position = input(f"\n{player2}, choose your position in the list: ")
        try:
            position = int(position)
            if (position < 1 or position > 25):
                print("Please enter a value between 1 and 25!")
                continue
            else:
                break
        except:
            print("Please enter a valid number!")
            continue

    # Convert to a 1d list
    list_1d = convert_list_1d(list_2d)

    # Check if the position entered is already full
    while(list_1d[position - 1] != 0):
        print("\nPlease enter an empty position position!")
        position = int(input(f"\n{player2}, choose your position in the list: "))

    # Add a two to the players selected position
    list_1d[position - 1] = 2

    # Convert to a 2d list
    list_2d = convert_list_2d(list_1d)

    # Return values
    return (list_2d, position)


def convert_list_1d(list_2d):

    # Create one list from the number of elements in the 2d list
    list_1d = []
    for i in range(len(list_2d)):
        list_1d += list_2d[i]

    # Return the value
    return(list_1d)


def convert_list_2d(list_1d):

    # Split the 1d list into 5 part and create a 2d list with 5 rows
    rows = 5
    cols = 5
    list_2d = [list_1d[x:x+cols] for x in range(0,len(list_1d),cols)][:rows]

    # Print the list
    print("")
    for i in range(len(list_2d)):
        print(list_2d[i])

    # Return values
    return(list_2d)


def win_check(list_2d):

    a = 1
    b = 1
    c = 0

    # Run the win check for all 16 possible win positions, only break if true or the loop has ran
for all 16 solutions
    for a in range(4):
        for b in range(4):
            c = c + 1
            if(list_2d[a][b] == 1 and list_2d[a + 1][b] == 1 and list_2d[a][b + 1] == 1 and
list_2d[a + 1][b + 1] == 1
```

```
         or list_2d[a][b] == 2 and list_2d[a + 1][b] == 2 and list_2d[a][b + 1] == 2 and
list_2d[a + 1][b + 1] == 2):
            return(True)
         elif(c < 16):
            continue
         else:
            return(False)


def draw_check(list_2d):

   # Convert to a 1d list
   list_1d = convert_list_1d(list_2d)

   # Check if 0 is in the list
   if 0 not in list_1d:
      return(True)
   else:
      return (False)

# Start point
choose_player()
```

Testing Table –

In this table I am going to check 4 of the possible winning situations at random for both player 1 and player 2. This should be enough to confirm my win function is working without testing all 16 possible situations.

Input	Expected Output	Actual Output	Comments
Player 1 – 3, 4, 8, 9 Player 2 – 20, 25, 13	This should result in a Player 1 win situation, thus triggering the game restart.	This triggered the Player 1 win as I expected.	This is the test of one of the 16 win situations.
Player 1 17, 18 , 21, 22 Player 2 – 20, 25, 13	This should result in a Player 1 win situation, thus triggering the game restart.	This triggered the Player 1 win as I expected.	This is the test of one of the 16 win situations.
Player 1 – 1, 2, 22, 23 Player 2 – 7, 6, 11, 12	This should result in a Player 2 win situation, thus triggering the game restart.	This triggered the Player 2 win as I expected.	This is the test of one of the 16 win situations.
Player 1 – 1, 3, 12, 15 Player 2 – 25, 24, 20, 19	This should result in a Player 2 win situation, thus triggering the game restart.	This triggered the Player 2 win as I expected.	This is the test of one of the 16 win situations.

Brandon Tonge
ID - 201358937

All the positions are full with no player winning.	This should trigger the draw function and end the game stating to both player a draw has been reached and no more moves can be made.	This triggered the draw function as expected.	This test proves that the draw function is activated when there is no 0's left in the list.
Player 1 – 3 Player 2 – 3	This should trigger the message telling the player that they cannot occupy a taken position. Prompting them to select another choice.	As I expected this triggered the error asking the player to enter a value that is not currently occupied.	Although this only tests the validation on player two the same code is in place for player 1. I'm happy that the function works and checks that the position is empty.

Input	Expected Output	Actual Output	Comments
Player 1 – 28	I expect the program to give a specific error message saying the input is out of the expected range.	The error message ran as I expected.	This error is inside the try and catch test. This proves that it passes the try but not the check for the range.
Player 1 – One	I expect the program to give the specific error message saying the input is not a number.	The error message ran as I expected.	This proves that the try and except works and won't let anything but and integer pass through.
Player 2 – 34	I expect the program to give a specific error message saying the input is out of the expected range.	The error message ran as I expected.	This error is inside the try and catch test. This proves that it passes the try but not the check for the range.
Player 2 – Two	I expect the program to give the specific error message saying the input is not a number.	The error message ran as I expected.	This proves that the try and except works and won't let anything but and integer pass through.
Play again – yes	I expect the program to ask for of the options to be inputted.	The error message ran as I expected.	This activates the else in the loop as it isn't one of the two expected inputs.
Play again – Y	The program should restart. Asking player 2 to input their choice.	The program runs the create board function as expected.	The program acted as expected and just restarts the game from the beginner.
Play again – N	The program should thank the player for	The program printed the message and	This ends the program entirely.

	playing and then exit the console.	exited the program entirely.	

Pseudocode –

```
OUTPUT Title of the game and instructions

CREATE 5 by 5 list of 0's

LOOP
    OUTPUT Ask player 1 to choose a position between 1-25
    INPUT  Choose position in the list
    CONVERT Input to a 2d list index
    FUNCTION Check win
    FUNCTION Check draw

    OUTPUT Ask player 2 to choose a position between 1-25
    INPUT Choose position in the list
    CONVERT Input to a 2d list index
    FUNCTION Check win
    FUNCTION Check draw

CHECK WIN
    SEARCH List for all 16 possible winning combinations

CHECK DRAW
    SEARCH List to see if there are any positions with 0 left in them
```