Assignment 3

Here you will begin to work towards building a tic-tac-toe game.

1. 25 points

Given a tuple \( t \), and an index \( i \), and a value \( v \), produce a new tuple which is the old tuple updated to have value \( v \) at index \( i \).

That is, finish the following:

\[
def assign(t, i, v):
    #your code here
\]

For example, \( \text{assign}((40, 30, 20), 2, 15) \) should return \((40, 30, 15)\).

You do not have to handle the case where the index is out of bounds, only the case where \( 0 \leq i < \text{len}(t) \).

2. 45 points

Assume that a position is represented as a tuple of tuples.

This is an example of a position:

\[
(('O', 'X', '-'), ('X', '-', 'X'), ('O', '-', 'O'))
\]

Given an input position, and player (represented as the character 'X' or 'O'), determine if a player has one on a row, column, or diagonal.

You will return True or False.

That is, finish the following functions:

\[
def won_row(position, player):
    #your code here
\]

\[
def won_column(position, player):
    #your code here
\]

\[
def won_diagonal(position, player):
    #your code here
\]
3. 15 points

Write a function which returns a tuple representing the argmax, followed by the max of a list. That is, finish the following:

def argmax_max(x):
    #your code here

If the max isn’t unique, argmax can be any index corresponding to a max.

For instance, argmax_max([0,5,9,4,3,7,6,4,9,2,3,9,4]) could return the tuple (2,9)

4. 15 points

Write the analogous function argmin_min.