Problem 1.

Consider the following list of list:

```
L = [[1, 2, 3],
     [4, 5, 6],
     [7, 8, 9]]
```

Write a function that modifies the list `L`, modifying both `L[0]` and `L[0][0]`. Observe and explain the results to the TA.

Problem 2.

Write a function that takes in a list formatted like `L`, then changes the first "row" of `L`, then prints out the result, but without modifying the global variable `L`.

Make sure that the global variable is unchanged after the function is called by printing its value.

Problem 3. Getting starting with Gomoku

In this question, you will get started on Project 2.

Part (a)

Write the function `is_sq_in_board(board, y, x)` which returns `True` iff the square `(y, x)` is a valid square in the Gomoku board `board`.

Part (b)

In the question, you use code similar to what's given to you in `put_sequece_on_board`. This is to help you get started with the Gomoku project.

First, read the code of `put_sequece_on_board`, and make sure you understand it. Talk to a TA if necessary.

Write a function `is_sequence_complete(board, col, y_start, x_start, length, d_y, d_x)`.

The function should return `True` if there is a sequence of exactly `length` stones starting at location `y_start, x_start` of colour `col`. If there is a stone of colour `col` either immediately before or immediately after the sequence, the function should return `False`. If there is no sequence of length `length` starting at location `(start_y, start_x)`, the function should return `False`.

Problem 4.

Note that MATLAB automatically creates a deep copy of a matrix passed to it only if the matrix is modified inside the function, but not otherwise.

We will now implement some of what's required for this behaviour to be possible.

You can obtain the source code of a function this way:

```
def f():
    print("hi")
    print("hello")
```

```
import inspect
lines = inspect.getsource(f).split("\n")
```

You can obtain a list of arguments from a string using

```
>> "def f(a, bc, d)".split("(")
['def f', 'a, bc, d)']

>> "def f(a, bc, d)".split("(")[1]
'a, bc, d)'

>> "def f(a, bc, d)".split("(")[1][:-1]
'a, bc, d'

>> "def f(a, bc, d)".split("(")[1][:-1].split(",")
['a', ' bc', ' d']
```

Use `exec` to:

- "Back up" the value of all parameters using deep copy (assume all parameters are lists of lists of integers)
- If executing a line causes a change in the value of a parameter, restore the parameter to its original value

Note that this does not completely do what MATLAB does – it's just a proof of concept.