Day 3: *Lists in Python*

A long flight is an excellent time for reflection and meditation, and Dot couldn't wait to settle into their seat and rest. The plane was huge, with hundreds of seats, rows and rows of them stretching down the long aisles. Dot glanced down at their ticket to see which of the seats belonged to them. Just at that moment, another passenger in the aisle stumbled and knocked right into Dot. "Oh dear, I apologize, fellow flyer!" the passenger says, "sometimes I just can't seem to get my bearings, stumbling every which way! I hope I did not disturb you too intensely." Dot smiled and reassured the odd woman, clarifying that they weren't harmed.

Dot looked back towards their ticket — oh no! It seemed that Dot spilled some of their coffee on the slip of paper during the hubbub. There was a dark wet spot right over where the seat number was supposed to be. Dot looked up and down the plane and saw that most of the seats were already occupied by passengers. They need to find their seat quickly before the plane takes off — let's use our data skills to help them verify their seat number.

Tutorial

In Python, **lists** are a type of variable that can be sequenced. Lists can contain many items that can be indexed by integers, with items being kept separate by the ','.

```
list = [] #lists are called using the square brackets

list_of_int = [1,2,3,4,5] #list of integers
list_of_string = ['string1','string2', 'string3'] #list of strings
list_of_anything = [1, 'string', 3.2] #list with an integer, string, and
float.
```

To learn more about lists, check out this article here.

Indexing or Slicing Lists

In the previous challenge we introduced you to lists. Lists are Python data structures that can hold many types of data in a sequence. The act of calling specific data in a list based on their position within the list is referred to as **indexing or slicing**.

```
list_of_int = [1,2,3,4,5] # list of integers
list_of_string = ['string1','string2', 'string3'] # list of strings
list_of_anything = [1, 'string', 3.2] # list with an integer, string, and
float.

list_of_string[0] # will print out 'string1'

list_of_int[0:2] # will print out [1,2,3]
```

To read more on list indexing and slicing, read this article

Note: Python is a zero-based indexing language, meaning that the first element of a sequence will be at index = 0. In some other languages, the first element of a sequence is at index = 1 instead.

Appending and Replacing Values in Lists

Python lists are mutable structures, meaning a list's size or values can be changed. Below are some of the common methods of changing values in a list.

To learn more about the various Python list methods, read this article

```
list_of_int = [1 , 2 , 3 , 4 ,5] # list of integers

list_of_int.append('value') # Running this line of code will update
list_of_int = [1,2,3,4,5, 'value']

list_of_int.insert(3 , 3.5) # Insert 3.5 in index 3, resulting in list_of_int
= [1,2,3,3.5,4,5]

list_of_int.pop(0) # Removes the value in position 0, list_of_int = [2,3,4,5]
```

Challenge

We need to find Dot an empty seat on the plane. We have logged the layout of the section of the plane as a Python list. This is formatted as a **nested** list, which means that elements of the list are lists as well. Listception.

Note: Our plane section contains three sections of seats in each row, and three seats in each section. Here is what the values in the layout mean:

- `e` = empty seat
- `o` = occupied seat
- None = aisle (you can't sit here!)
- Capital letters = window seats

```
layout = [
    ["O","e","e",None, "e","o","e",None, "o","o","E"], # row 1
    ["O","o","e",None, "e","o","e",None, "e","o","E"], # row 2
    ["O","o","o",None, "o","o","e",None, "o","e","O"], # row 3
    ["E","o","e",None, "e","o","e",None, "o","e","E"], # row 4
    ["E","e","o",None, "e","e","e",None, "o","o","E"], # row 5
    ["O","e","e",None, "e","e","e",None, "e","e","E"]  # row 6
]
```

Dot has two conditions for their sitting:

- It has to be window seat
- The next seat needs to be empty

Identify the right place for Dot to sit. If there is more than one option that matches the conditions above, choose the seat that is closer to the front, i.e. has a lower row number. What is the list index of Dot's seat?

Stretch Question

Use python's pop() and insert() list functions to change Dot's seat from `E` to `O`.

Stretch Questions are not required to be completed to finish the challenge but are recommended to further develop your skills.

```
new_layout = ...
```

In [19]:

```
# SOLUTION
layout = [
    ["O","e","e",None, "e","o","e",None, "o","o","E"], # row 1
    ["O","o","e",None, "e","o","e",None, "e","o","E"], # row 2
    ["O","o","o",None, "o","o","e",None, "o","e","O"], # row 3
    ["E","o","e",None, "e","o","e",None, "o","e","E"], # row 4
    ["E","e","o",None, "e","e","e",None, "o","o","E"], # row 5
    ["O","e","e",None, "e","e","e",None, "e","e","E"]  # row 6
]

print(str(len(layout)) + ' rows')
print('row: ' + str(layout[0]))
print('length of the row: ' + str(len(layout[0])))

print("Correct indexes for the Dot's seat: ")
layout[3][-1]
```

```
6 rows
row: ['O', 'e', 'e', None, 'e', 'o', 'e', None, 'o', 'o', 'E']
length of the row: 11
Correct indexes for the Dot's seat:
```

Out[19]:

```
'E'
```

In [21]:

```
# STRETCH SOLUTION
new_layout = layout
new_layout[3].insert(-1,'O')
new_layout[3].pop(-1)
new_layout
```

Out[21]:

```
[['O', 'e', 'e', None, 'e', 'o', 'e', None, 'o', 'o', 'E'],
 ['O', 'o', 'e', None, 'e', 'o', 'e', None, 'e', 'o', 'E'],
 ['O', 'o', 'o', None, 'o', 'o', 'e', None, 'o', 'e', 'O'],
 ['E', 'o', 'e', None, 'e', 'o', 'e', None, 'o', 'e', 'O'],
 ['E', 'e', 'o', None, 'e', 'e', 'e', None, 'o', 'o', 'E'],
 ['O', 'e', 'e', None, 'e', 'e', 'e', None, 'e', 'e', 'E']]
```