Python Basic & Plus

Workshop - 4

Subject: Collections - Control Flow Statements - Functions

Coding Challenge -1 : Palindrome

Purpose of the this coding challenge is to solve a control flow statements issue.

Learning Outcomes

At the end of the this coding challenge, students will be able to;

- **understand the use of loops.**
- **understand the importance of alphanumeric string methods.**
- **get a better understanding in manipulating strings.**

Problem Statement

- **Write a function/functions that checks whether the sentence you get from the user is a palindrome. (Do not consider punctuation and special characters. Only consider "*alphanumeric*" characters.)**

```
input : "ey edip adana'da, pide ye!"

output : "ey edip adana'da, pide ye!" is a palindrome
```

Solution :

In [1]:

```
def is_palindrome(string):
    # backwards = string[::-1]
    # return backwards == string
    return string[::-1].casefold() == string.casefold()


def palindrome_sentence(sentence):
    string = ""
    for char in sentence:
```

```
        if char.isalnum():
            string += char
    print(string)
    # return string[::-1].casefold() == string.casefold()
    return is_palindrome(string)

word = input("Please enter a word to check: ")
if palindrome_sentence(word):
    print("'{}' is a palindrome".format(word))
else:
    print("'{}' is not a palindrome".format(word))
```

```
Please enter a word to check: Ey Edip Adana'da, pide ye!
EyEdipAdanadapideye
'Ey Edip Adana'da, pide ye!' is a palindrome
```

Coding Challenge - 2: Sudoku Format Converter&Printer

The purpose of this coding challenge is to write a program that prints the given lists as sudoku looking format.

Learning Outcomes

At the end of this coding challenge, students will be able to;

- **analyze a problem, identify, and apply programming knowledge for appropriate solution.**
- **design, implement `arithmetic operators` and nested loops effectively in Python to solve the given problem.**
- **demonstrate their knowledge of algorithmic design principles by solving the problem effectively.**

Problem Statement

Objective:

- **To improve your control flow statement skills.**

Task: The department you work for has received a project that displays the solved sudoku puzzles in a digital environment.

- **Write a Python code to print out the given `sudoku` puzzle matrix in the following format.**

Given format :

```
sudoku = [
    [0, 0, 0, 0, 6, 4, 0, 0, 0],
    [7, 0, 0, 0, 0, 0, 3, 9, 0],
    [8, 0, 0, 0, 0, 0, 0, 0, 0],
    [0, 0, 0, 5, 0, 2, 0, 6, 0],
    [0, 8, 0, 4, 0, 0, 0, 0, 0],
    [3, 5, 0, 6, 0, 0, 0, 7, 0],
    [0, 0, 2, 0, 0, 0, 1, 0, 3],
    [0, 0, 1, 0, 5, 9, 0, 0, 0],
    [0, 0, 0, 0, 0, 0, 7, 0, 0]
]
```

Desired output format :

```
- - - - - - - - - - - - - - -
0  0  0  | 0  6  4  | 0  0  0
7  0  0  | 0  0  0  | 3  9  0
8  0  0  | 0  0  0  | 0  0  0
- - - - - - - - - - - - - - -
0  0  0  | 5  0  2  | 0  6  0
```

```
0  8  0  | 4  0  0  | 0  0  0
3  5  0  | 6  0  0  | 0  7  0
- - - - - - - - - - - - - - -
0  0  2  | 0  0  0  | 1  0  3
0  0  1  | 0  5  9  | 0  0  0
0  0  0  | 0  0  0  | 7  0  0
- - - - - - - - - - - - - - -
```

Note that;

- ***Use not more than "control flow statement and boolean logic operators" in solving this code problem.***
- ***The output which we expect from you is only a new output format above.***
- ***We don't expect a sudoku puzzle solver from you.***

Solution :

In [3]:

```
sudoku = [
    [0, 0, 0, 0, 6, 4, 0, 0, 0],
    [7, 0, 0, 0, 0, 0, 3, 9, 0],
    [8, 0, 0, 0, 0, 0, 0, 0, 0],
    [0, 0, 0, 5, 0, 2, 0, 6, 0],
    [0, 8, 0, 4, 0, 0, 0, 0, 0],
    [3, 5, 0, 6, 0, 0, 0, 7, 0],
    [0, 0, 2, 0, 0, 0, 1, 0, 3],
    [0, 0, 1, 0, 5, 9, 0, 0, 0],
    [0, 0, 0, 0, 0, 0, 7, 0, 0]
]

# ------------------------------------------- #

count = 0
print("- - - - - - - - - - - - - - - - - ")
for i in sudoku:
    for j in range(9):
        print(i[j], " ", end="")
        if (j+1) == 9 :
            print()
            count+=1
            if count%3==0 and count!=0 :
                print("- - - - - - - - - - - - - - - - - ")
        if (j+1) % 3 == 0 and j != 0 and j!=8:
             print("| ", end="")
```

```
- - - - - - - - - - - - - - - - -
0  0  0  | 0  6  4  | 0  0  0
7  0  0  | 0  0  0  | 3  9  0
8  0  0  | 0  0  0  | 0  0  0
- - - - - - - - - - - - - - - - -
0  0  0  | 5  0  2  | 0  6  0
0  8  0  | 4  0  0  | 0  0  0
3  5  0  | 6  0  0  | 0  7  0
- - - - - - - - - - - - - - - - -
0  0  2  | 0  0  0  | 1  0  3
0  0  1  | 0  5  9  | 0  0  0
0  0  0  | 0  0  0  | 7  0  0
- - - - - - - - - - - - - - - - -
```

In [ ]:

```

```