Fn: fillMyMatrixPat(myMat)

- Write a recursive function which goes through an empty matrix passed to it which has many levels of inner matrices with varying depths, all of them being empty.
- Input (global values): strVal = "abcdefghijk", $PAT_LEN = someInt$ (some +ve int)
- The program needs to fill in the pattern as per the values in the above identifiers.
- The rules to be followed are the following:
 - 1. Assuming the above value is in strVal, after 'k' is filled into an empty list, it should wrap around from 'a'. **Note**: The string used, needs to be configurable manually.
 - 2. You are free to follow my implementation (outerChkMyMatrix) discussed in the Session 17 or your own as the starting point, but it needs to use inner functions with recursion.
 - 3. You can assume that the list object passed to the function has **sub-lists**, which are all empty.
 - 4. Returns the same list object back which is filled in with the pattern as per configuration values.
 - 5. You can now use the filled in matrix to test the *outerChkMyMatrix()* code too.

Sample inputs and expected outputs are given in the next pages ...

Inputs (empty list) and Outputs (filled list)

1

calling fillMatrixPat

```
strVal: abcdefghijk
PAT_LEN: 3
myMat0: [] myMat0:
['a', 'b', 'c'] myMat1:
[[]]
myMat1: [['a', 'b', 'c']]
myMat2: [[], [], [], []]
myMat2: [['a', 'b', 'c'], ['d', 'e', 'f'],
             ['g', 'h', 'i'], ['i', 'k', 'a']]
strVal: abcdefghijk
PAT LEN: 4
mvMat0: []
myMat0: ['a', 'b', 'c', 'd']
myMat1: [['a', 'b', 'c', 'd']]
myMat2: [[], [], [], []]
myMat2: [['a', 'b', 'c', 'd'], ['e', 'f', 'g', 'h'],
            ['i', 'j', 'k', 'a'], ['b', 'c', 'd', 'e']]
```

```
strVal: abcdefghijk
PAT LEN: 5
myMat0: []
myMat0: ['a', 'b', 'c', 'd', 'e']
myMat1: [[]]
myMat1: [['a', 'b', 'c', 'd', 'e']]
myMat2: [[], [], [], []]
myMat2: [['a', 'b', 'c', 'd', 'e'],
            ['f', 'g', 'h', 'i', 'j'],
            ['k', 'a', 'b', 'c', 'd'],
            ['e', 'f', 'g', 'h', 'i']]
```

Inputs (empty list) and Outputs (filled list)

(python file for testing)

2

strVal: 0123456789

PAT_LEN: 2

myMat0: []

myMat0: ['0', '1']

myMat1: [[]]

myMat1: [['0', '1']]

myMat2: [[], [], [], []]

myMat2: [['0', '1'], ['2', '3'], ['4', '5'], ['6', '7']]

myMatrix2

myMatrix2

[[['0', '1'], ['2', '3']], [['4', '5']], [['6', '7'], ['8', '9'], ['0', '1']], [['2', '3']], ['4', '5'], ['6', '7'], ['8', '9']], ['0', '1']]

Note: Use outerChkMyMatrix() written in the Session 17

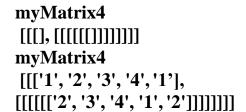
The result of checking the myMatrix2: True

The returned matrix is:

[[['0', '1'], ['2', '3']], [['4', '5']], [['6', '7'], ['8', '9'], ['0', '1']], [['2', '3']], [['4', '5'], ['6', '7'], ['8', '9']], ['0', '1']]

Note: **len(strVal)** < **PAT_LEN** is also valid





Note: Use outerChkMyMatrix() written in the Session 17
The result of checking the myMatrix4:
True
The returned matrix is:
[[['1', '2', '3', '4', '1'],

Assign 3: Inputs (empty list) and Outputs (filled list)

(python file for testing)

3

```
strVal: abcdefghijk PAT LEN: 3
```

```
myMatrix3
[[[],[]],[[],[]],[[],[]],[[],[]],[]]
myMatrix3
[[['a', 'b', 'c'],['d', 'e', 'f']],[['g', 'h', 'i']],[['j', 'k', 'a'],['b', 'c', 'd'],['e', 'f', 'g']],
[['h', 'i', 'j']],[['k', 'a', 'b'],['c', 'd', 'e'],['f', 'g', 'h']],['i', 'j', 'k']]
```

Note: Use outerChkMyMatrix() written in the Session 17

The result of checking the myMatrix3: True

The returned matrix is: