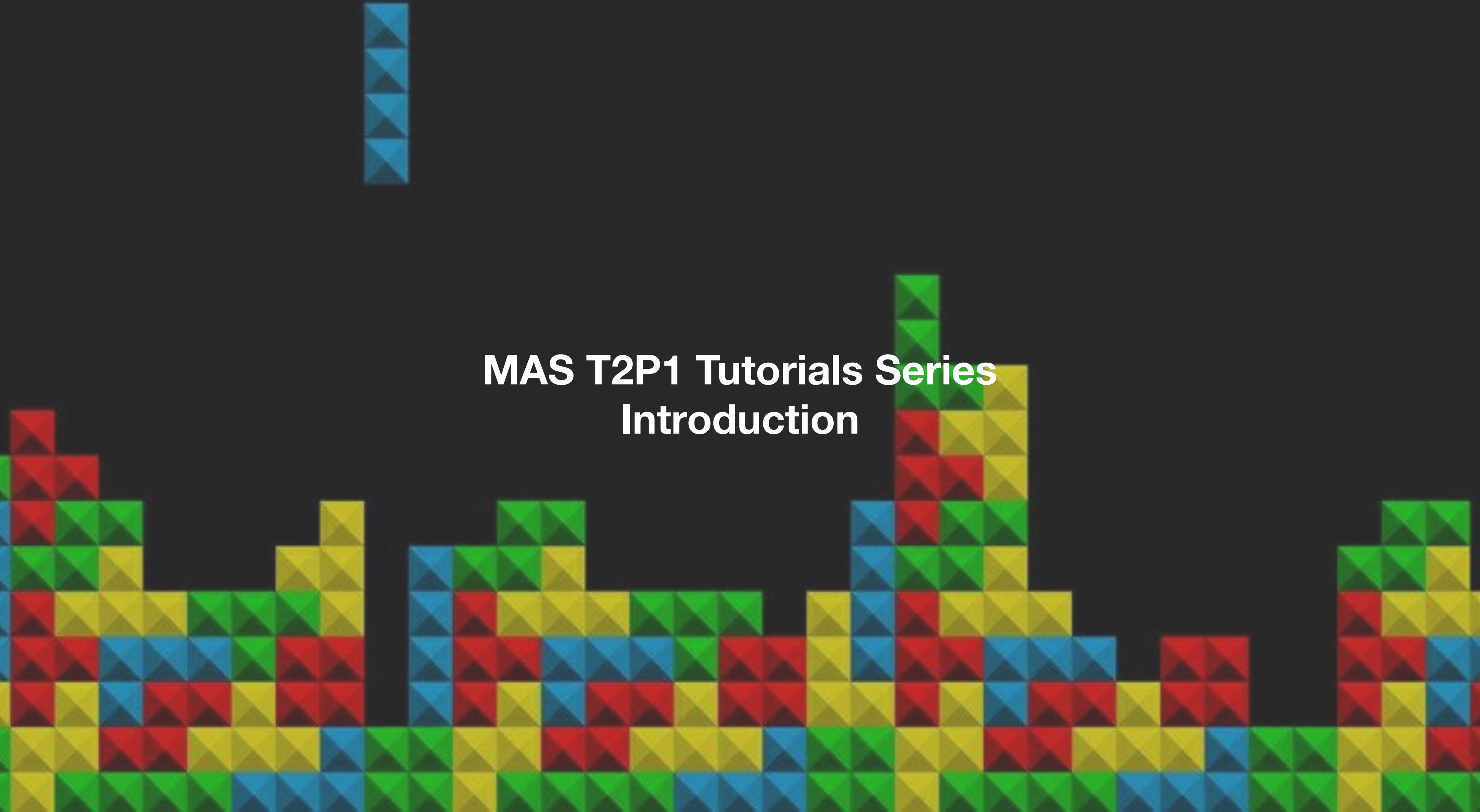
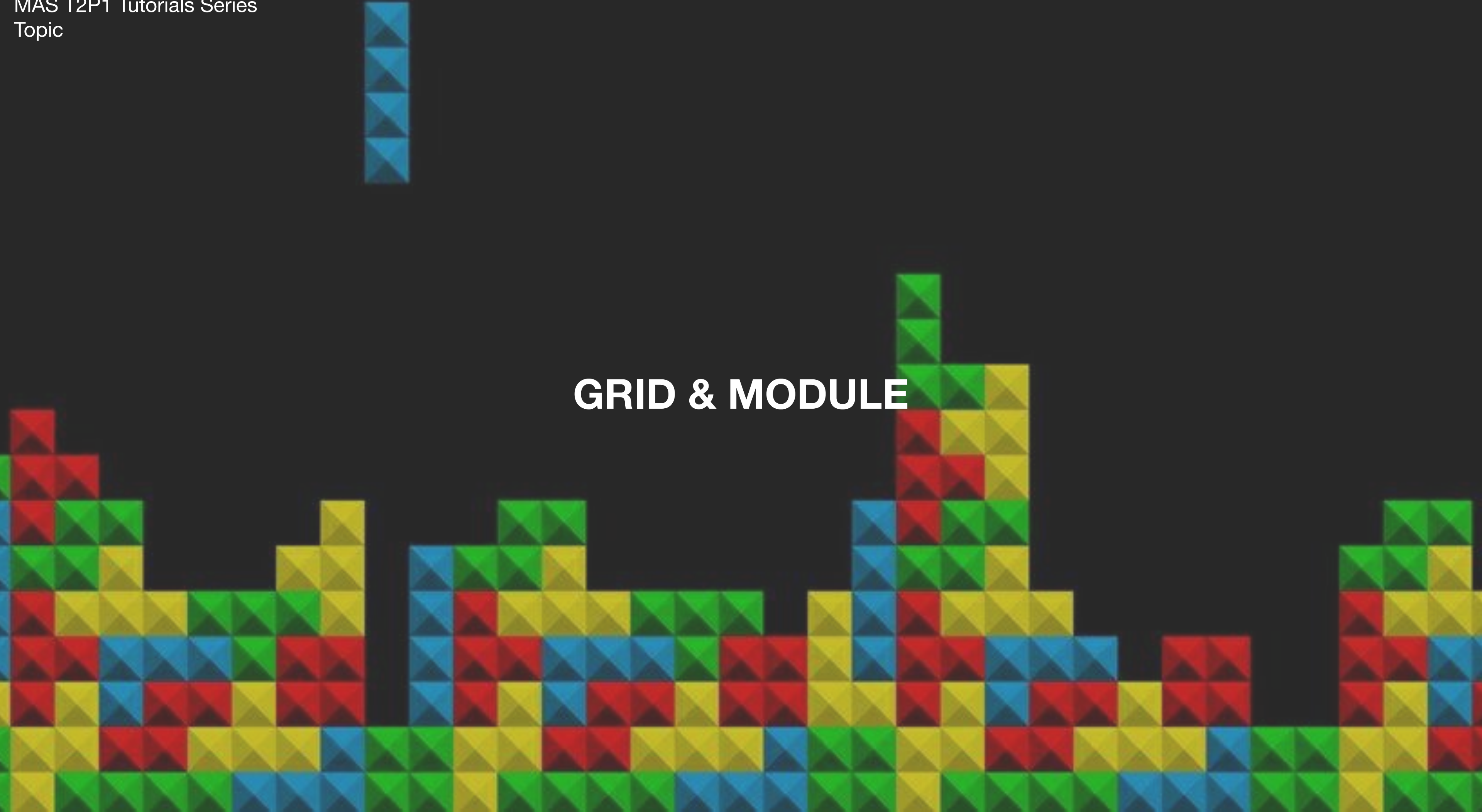


MAS T2P1 Tutorials Series
Introduction



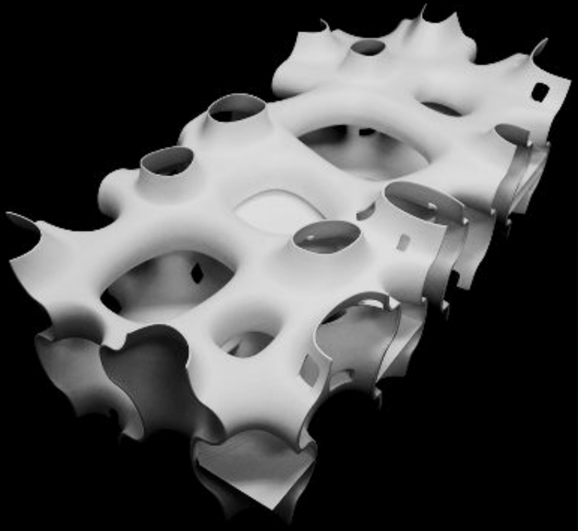
GRID & MODULE

WHY do we care?

Motivation

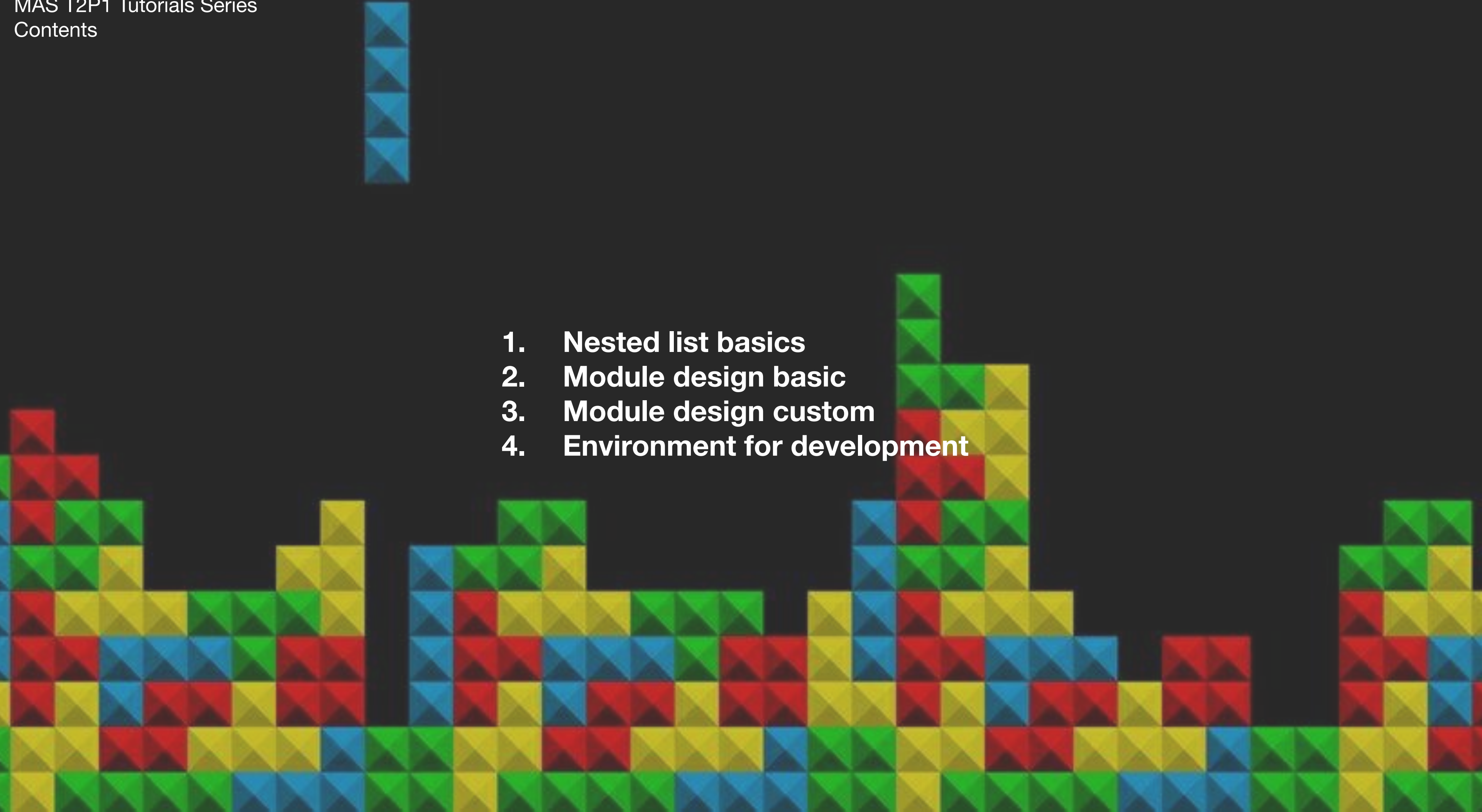
FUTURECRAFT 4D
Adidas + Carbon

Prada Sponge
OMA



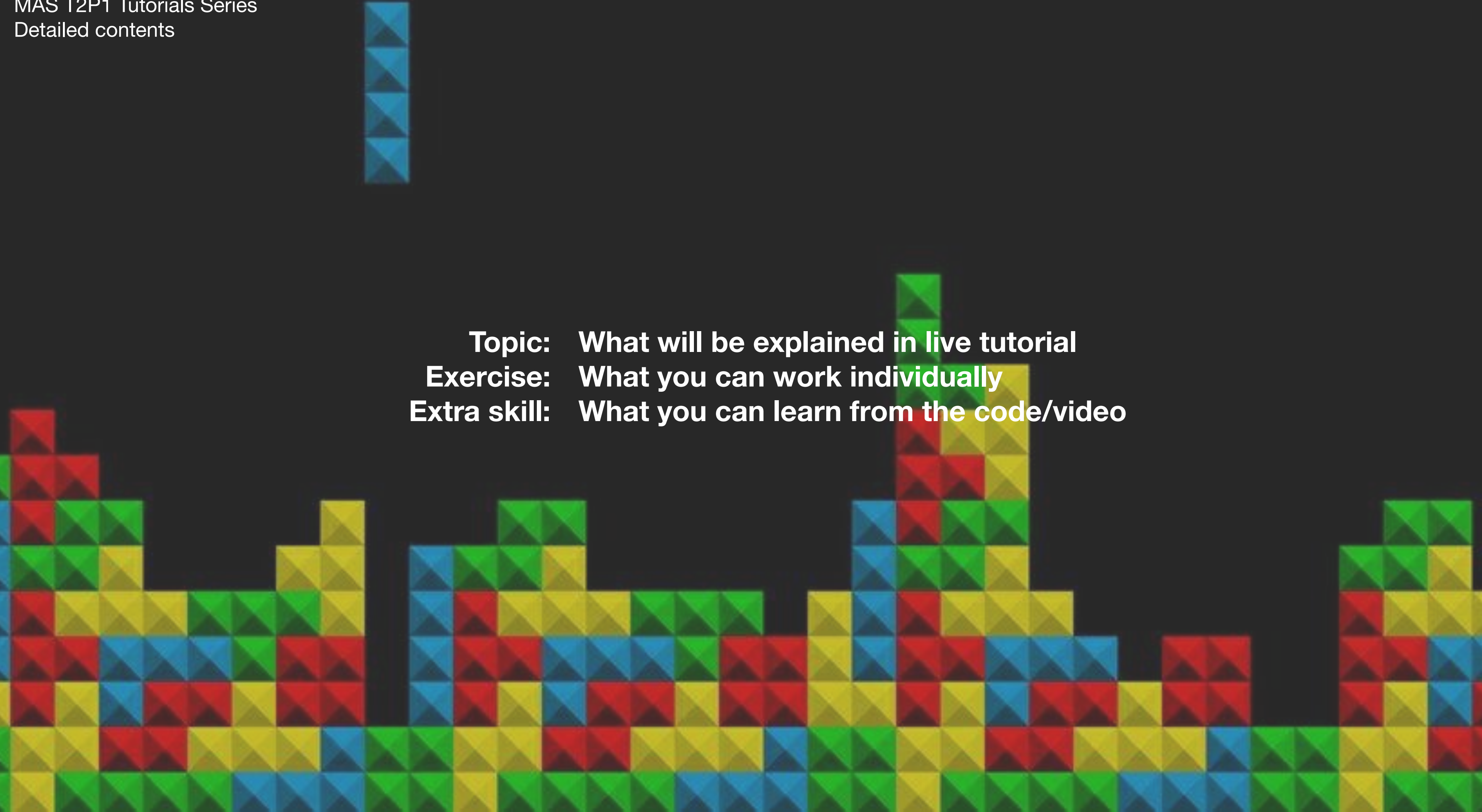
National Taichung Theatre
Toyo Ito

Contents



1. **Nested list basics**
2. **Module design basic**
3. **Module design custom**
4. **Environment for development**

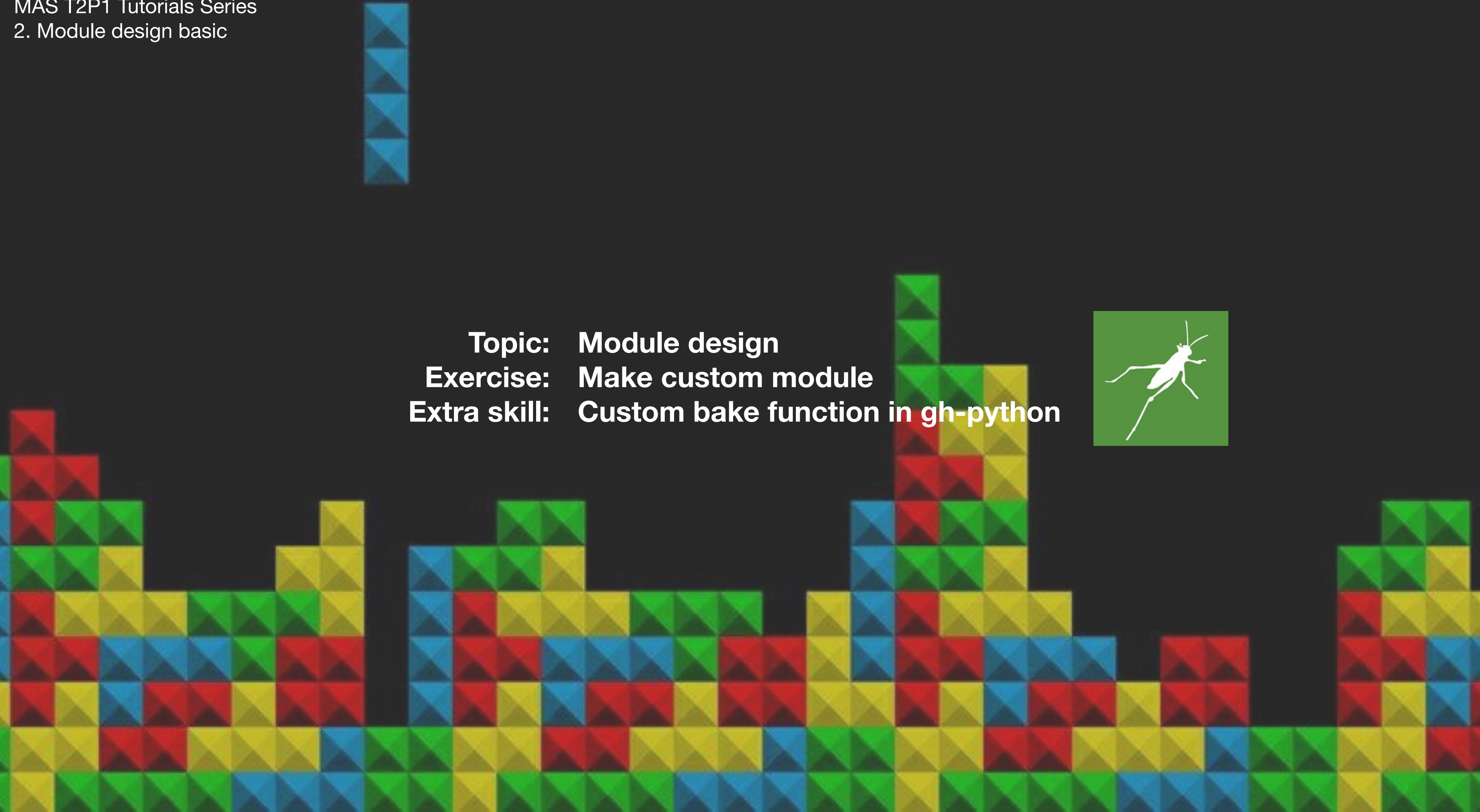
MAS T2P1 Tutorials Series
Detailed contents



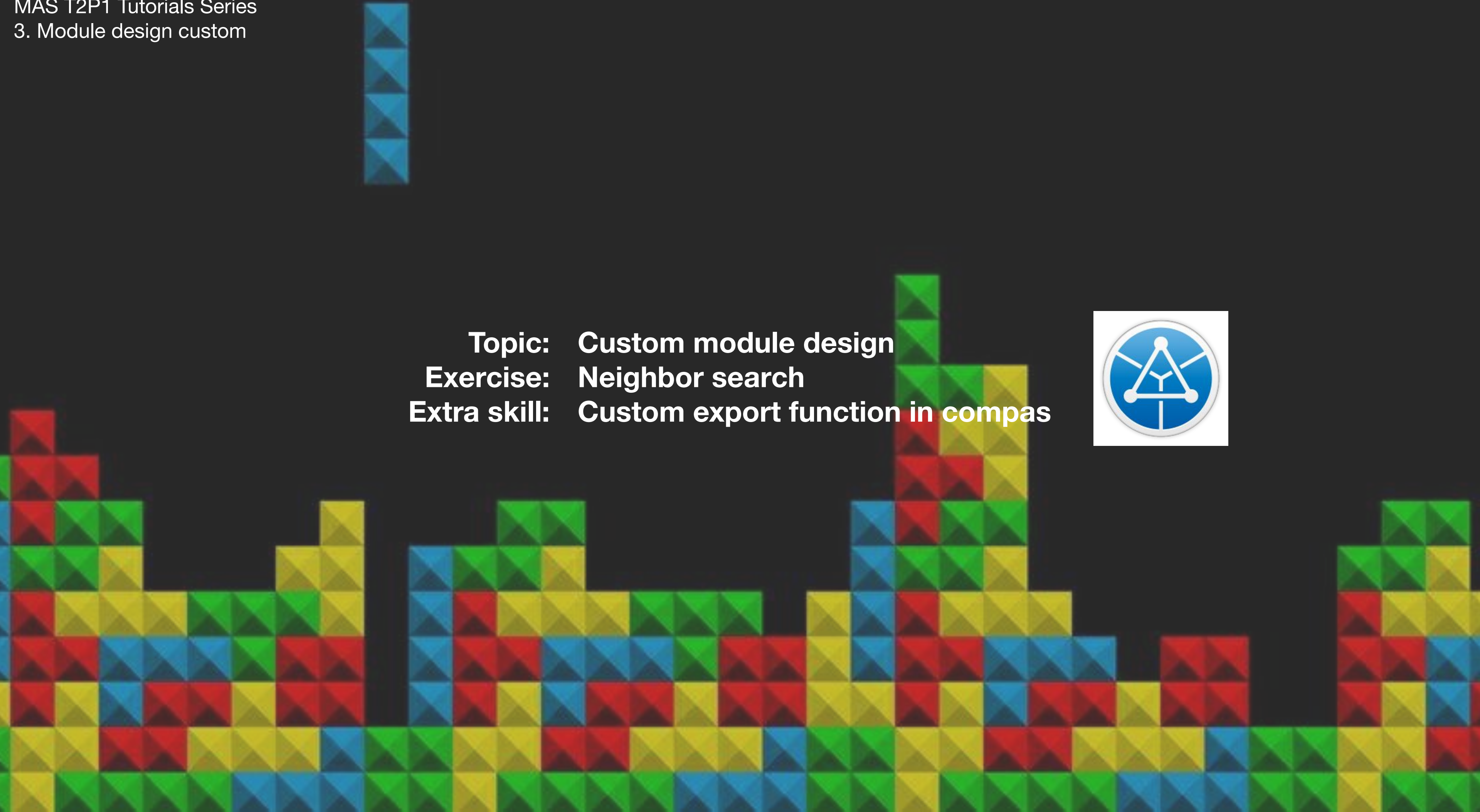
Topic:	**What will be explained in live tutorial**
Exercise:	**What you can work individually**
Extra skill:	**What you can learn from the code/video**

Topic:	**Nested list basics**
Exercise:	**List management / Operation**
Extra skill:	**Make your own branch in github**

MAS T2P1 Tutorials Series
2. Module design basic



Topic: Module design
Exercise: Make custom module
Extra skill: Custom bake function in gh-python



Topic: Custom module design

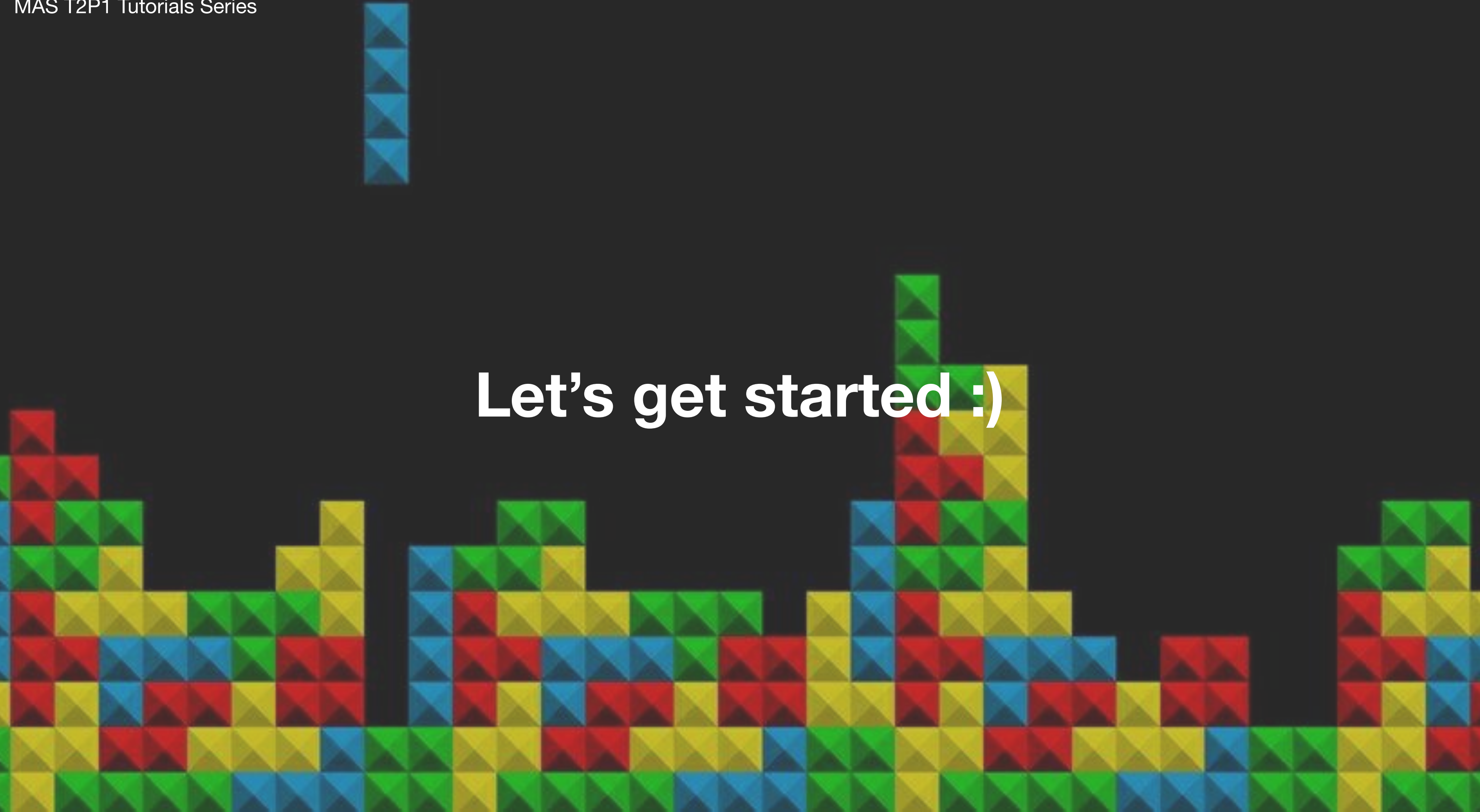
Exercise: Neighbor search

Extra skill: Custom export function in compas

Topic:	**Environment for development**
Exercise:	**Create own function**
Extra skill:	**Custom library / Environment control**

Basic knowledge of grid/module design with OOP

(Basic understanding of development workflow)



Let’s get started :)

Go to the link and make a clone

github.com/trtku/MAS2122-T2P1

Topic: **Nested list basics**

Exercise: **List management / Operation**

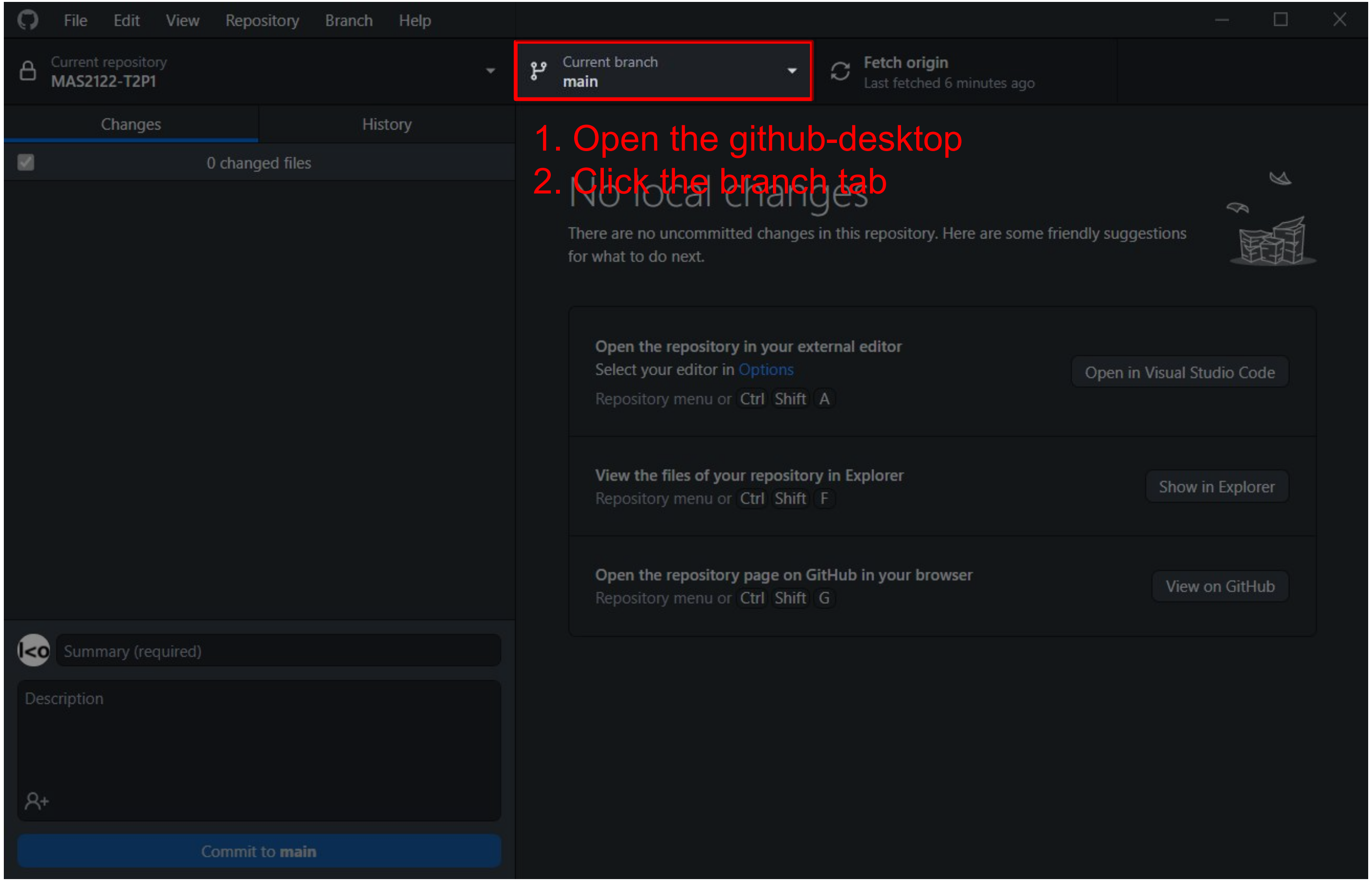
Extra skill: **Make your own branch in github**

Topic: Nested list basics

Exercise: List management / Operation

Extra skill: Make your own branch in github

MAS T2P1 Tutorials Series
Make your own branch in github



1. Open the github-desktop
2. Click the branch tab

No local changes

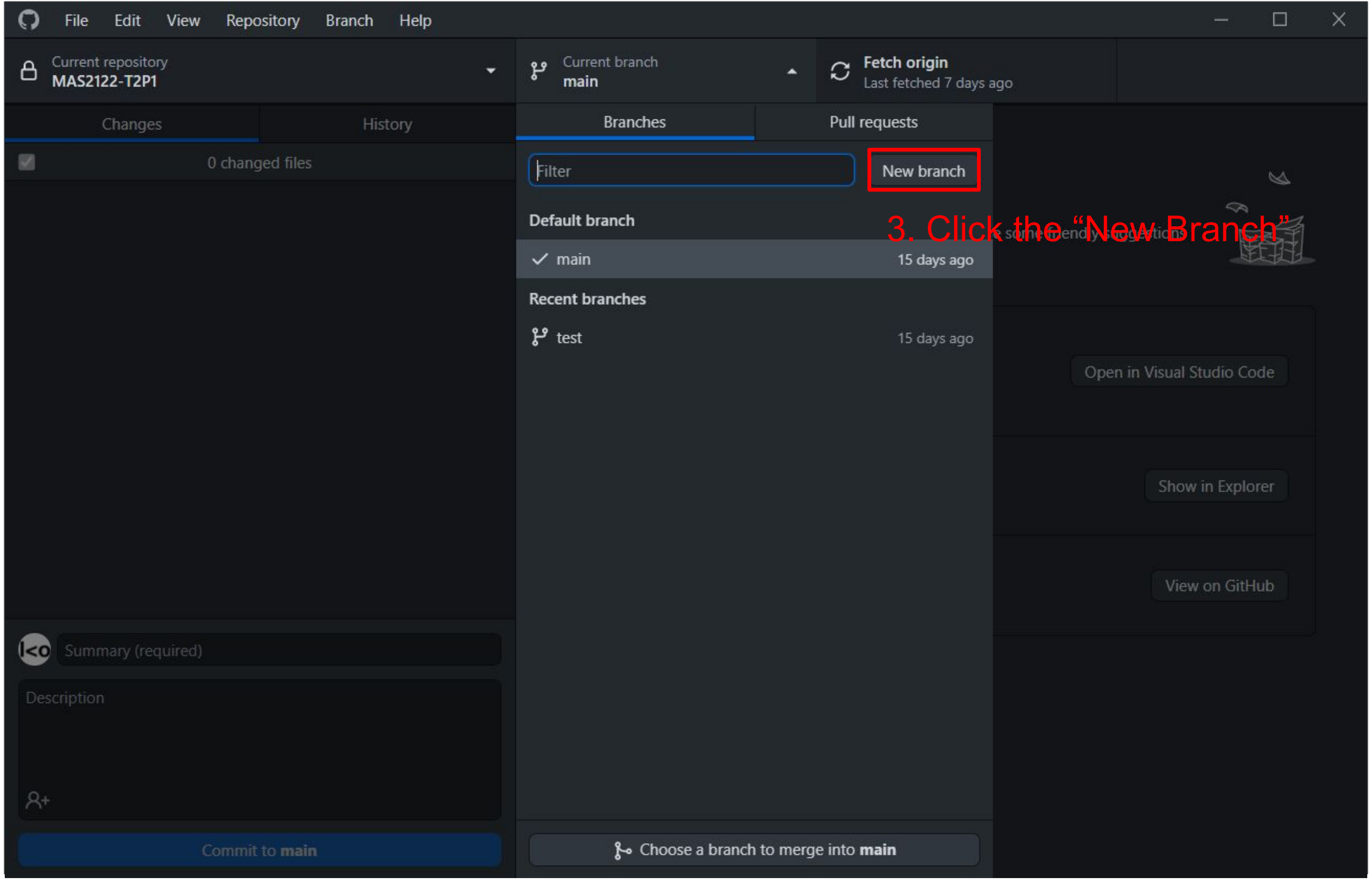
There are no uncommitted changes in this repository. Here are some friendly suggestions for what to do next.

Open the repository in your external editor
Select your editor in Options
Repository menu or Ctrl Shift A

View the files of your repository in Explorer
Repository menu or Ctrl Shift F

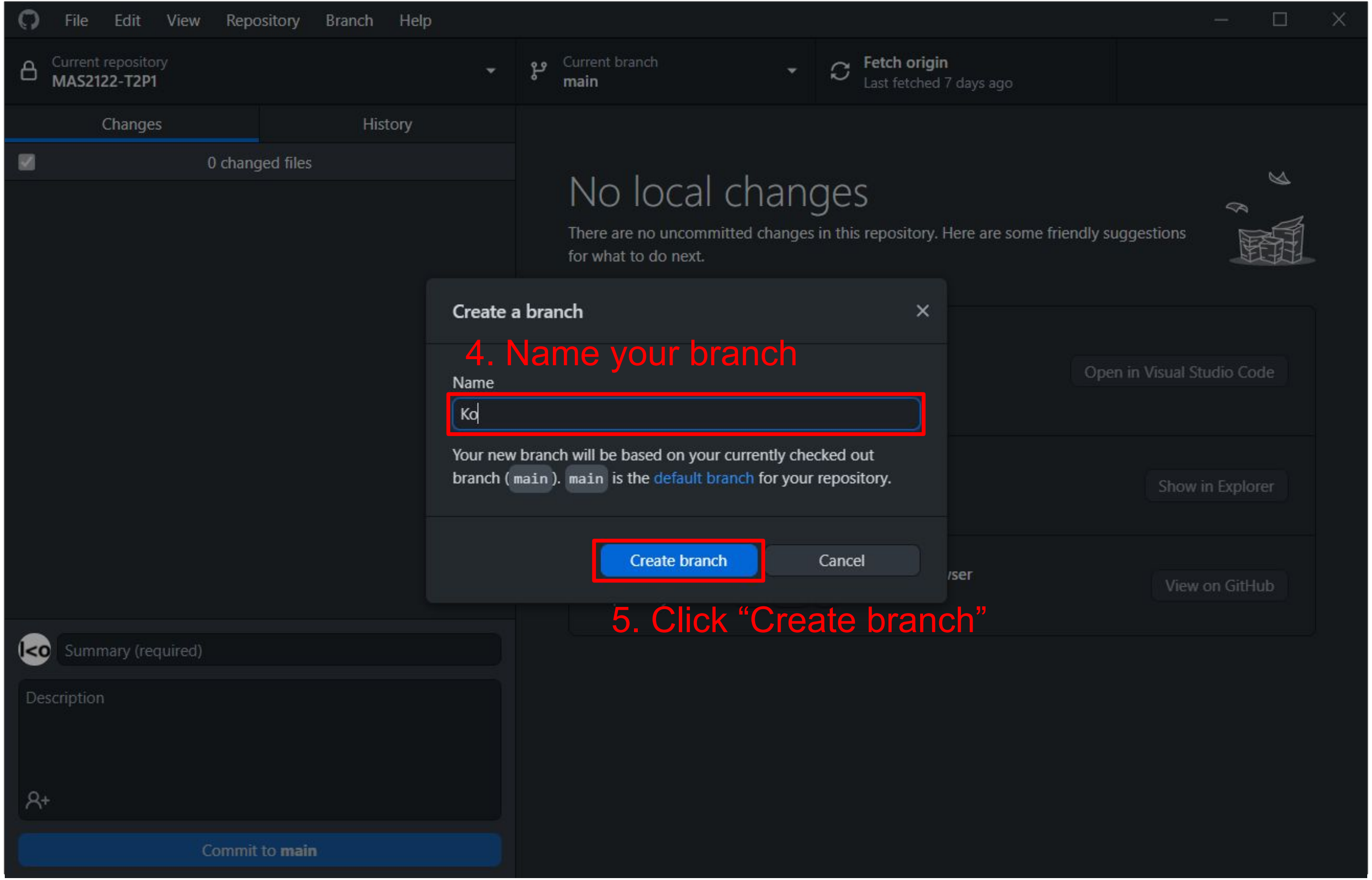
Open the repository page on GitHub in your browser
Repository menu or Ctrl Shift G

MAS T2P1 Tutorials Series
Make your own branch in github



3. Click the “New Branch”

MAS T2P1 Tutorials Series
Make your own branch in github



No local changes

There are no uncommitted changes in this repository. Here are some friendly suggestions for what to do next.

Create a branch

4. Name your branch

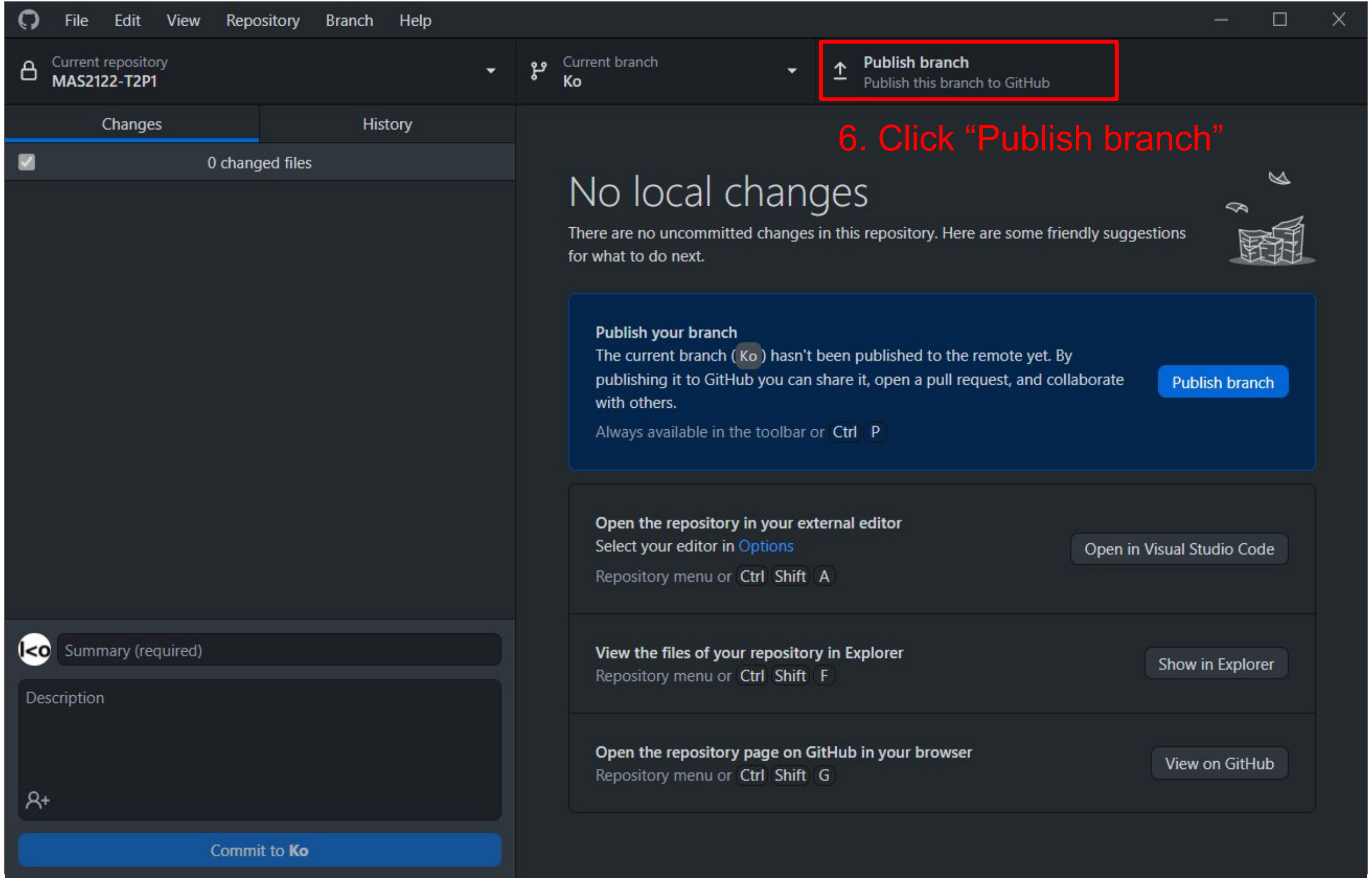
Name

Ko

Your new branch will be based on your currently checked out branch (`main`). `main` is the default branch for your repository.

5. Click “Create branch”

MAS T2P1 Tutorials Series
Make your own branch in github



6. Click “Publish branch”

No local changes

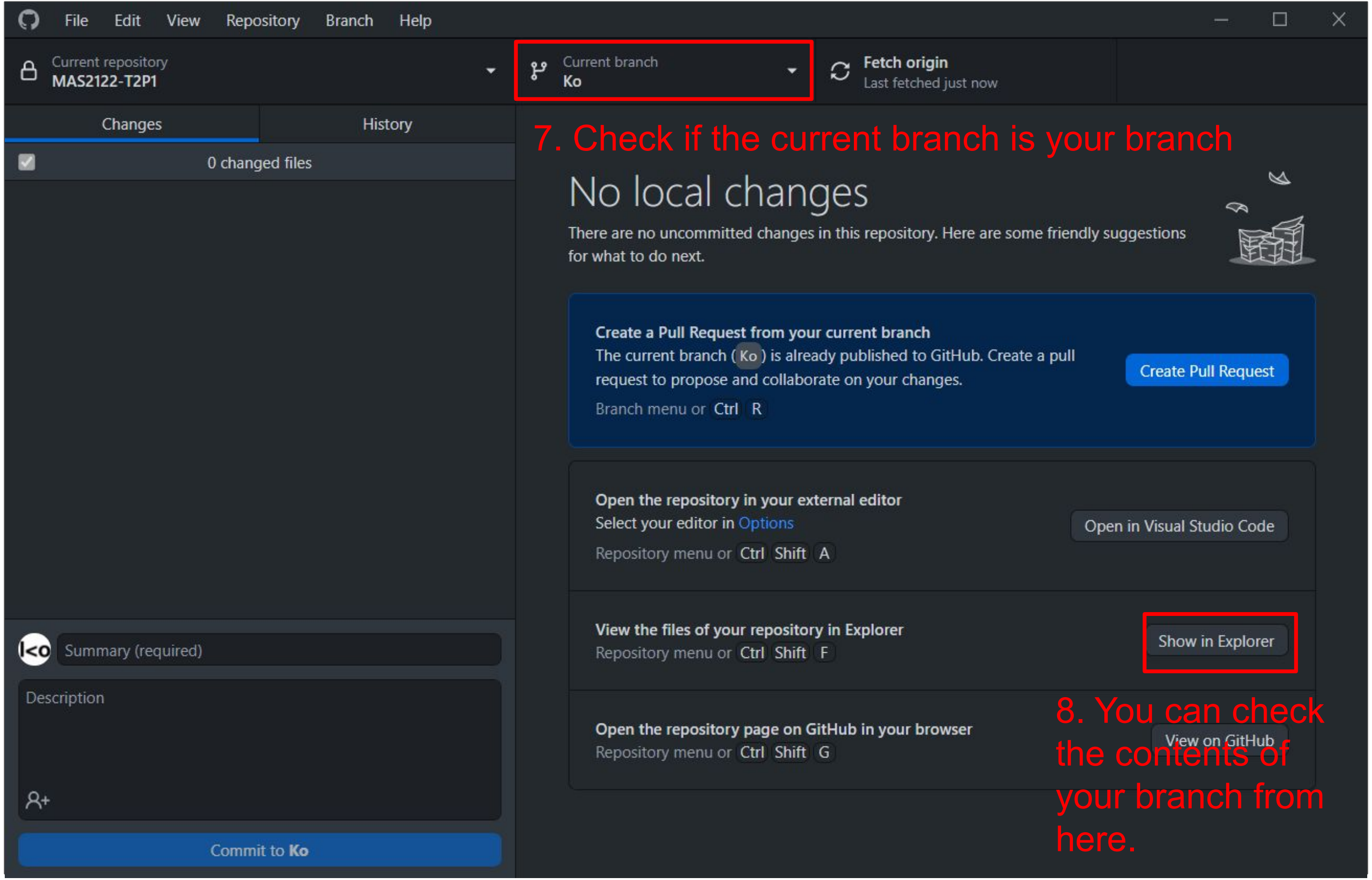
There are no uncommitted changes in this repository. Here are some friendly suggestions for what to do next.

Publish your branch
The current branch (Ko) hasn't been published to the remote yet. By publishing it to GitHub you can share it, open a pull request, and collaborate with others.
Always available in the toolbar or Ctrl P

Open the repository in your external editor
Select your editor in Options
Repository menu or Ctrl Shift A

View the files of your repository in Explorer
Repository menu or Ctrl Shift F

Open the repository page on GitHub in your browser
Repository menu or Ctrl Shift G



7. Check if the current branch is your branch

No local changes

There are no uncommitted changes in this repository. Here are some friendly suggestions for what to do next.

Create a Pull Request from your current branch
The current branch (Ko) is already published to GitHub. Create a pull request to propose and collaborate on your changes.
Branch menu or Ctrl R

Open the repository in your external editor
Select your editor in Options
Repository menu or Ctrl Shift A

View the files of your repository in Explorer
Repository menu or Ctrl Shift F

Open the repository page on GitHub in your browser
Repository menu or Ctrl Shift G

8. You can check the contents of your branch from here.

Congratulation!

❑ Develop, maintain and share your code.
❑ Collaborative workflow with team.
❑ Easy&Clear copyright handling to protect your IP(intellectual property)

Topic: Nested list basics

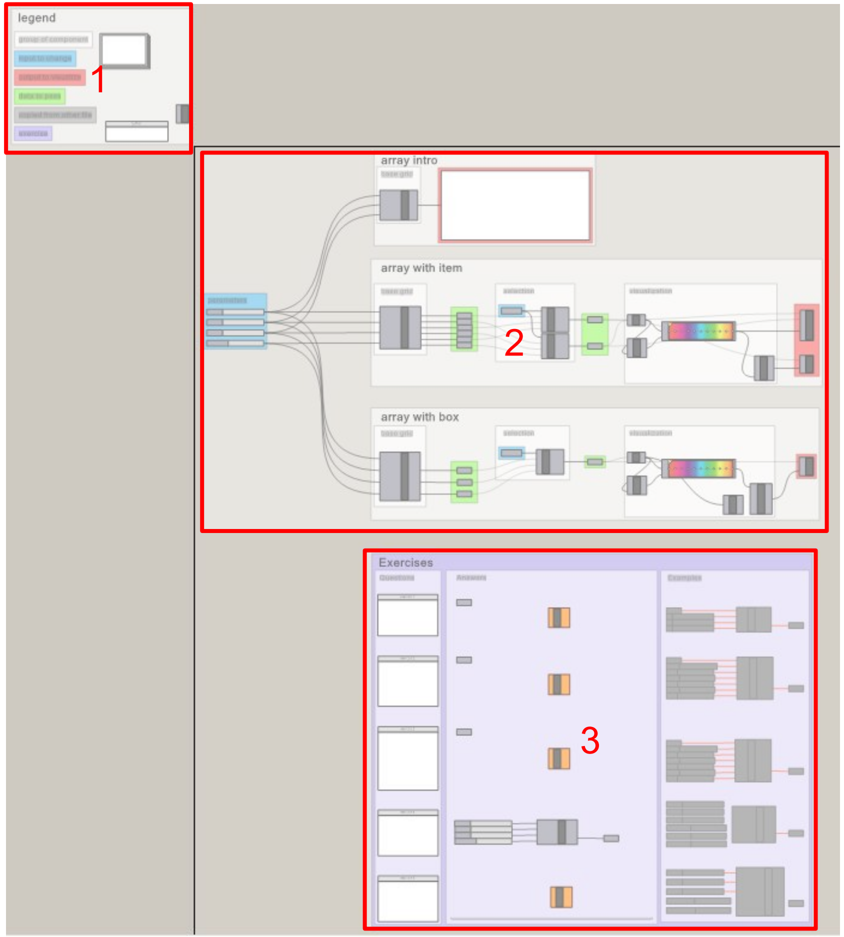
Exercise: List management / Operation

Extra skill: Make your own branch in github

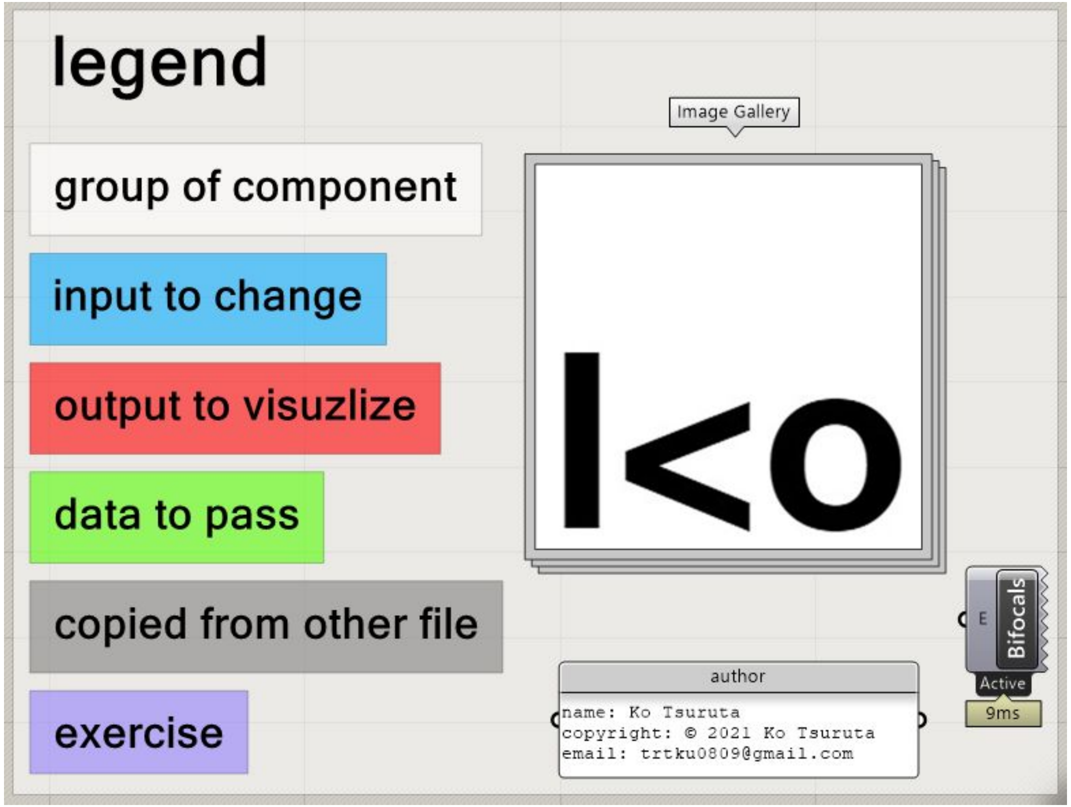
Open RH/GH & the .ghx file below

C:YOUR_LOCAL_DIR/MAS2122-T2P1/docs/tutorials/**00_nested_list_basics.ghx**

MAS T2P1 Tutorials Series
Nested list basics



1. Legend
2. Tutorial
3. Exercise



legend

group of component

input to change

output to visuzlize

data to pass

copied from other file

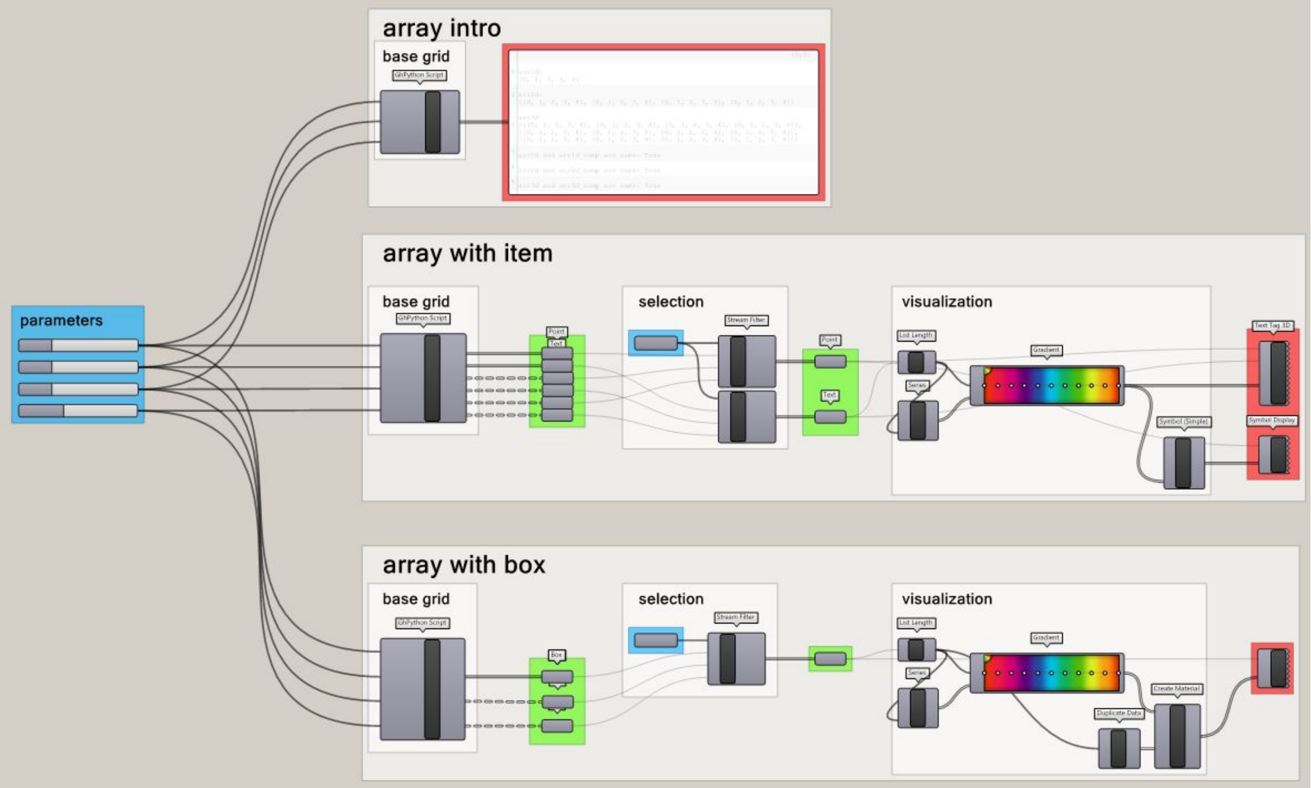
exercise

author

```
name: Ko Tsuruta
copyright: © 2021 Ko Tsuruta
email: trtku0809@gmail.com
```

1. Legend
2. Tutorial
3. Exercise

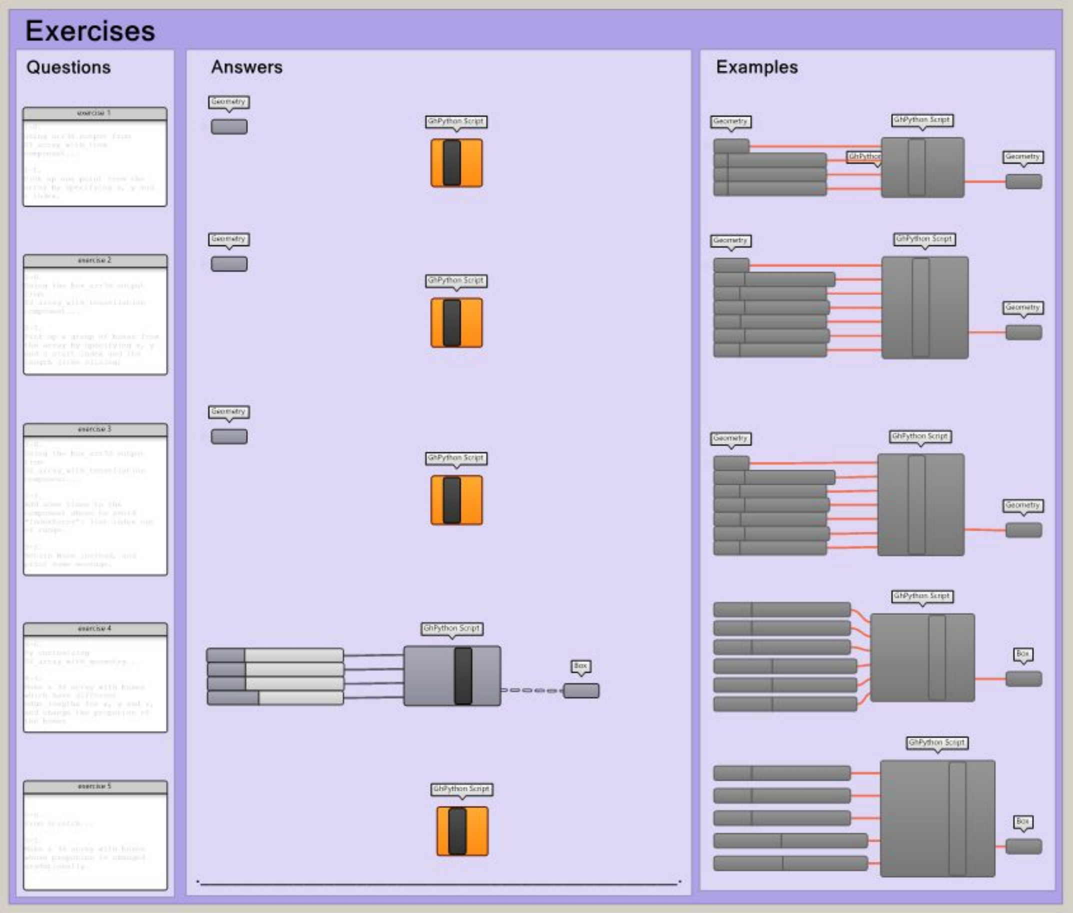
MAS T2P1 Tutorials Series
Nested list basics



```
arr1d:
[0, 1, 2, 3, 4]

arr2d:
[[0, 1, 2, 3, 4], [0, 1, 2, 3, 4], [0, 1, 2, 3, 4], [0, 1, 2, 3, 4]]

arr3d:
[[[0, 1, 2, 3, 4], [0, 1, 2, 3, 4], [0, 1, 2, 3, 4], [0, 1, 2, 3, 4]],
[[0, 1, 2, 3, 4], [0, 1, 2, 3, 4], [0, 1, 2, 3, 4], [0, 1, 2, 3, 4]],
[[0, 1, 2, 3, 4], [0, 1, 2, 3, 4], [0, 1, 2, 3, 4], [0, 1, 2, 3, 4]]]

arr1d and arr1d_comp are same: True

arr2d and arr2d_comp are same: True

arr3d and arr3d_comp are same: True
```

1. Legend
2. Tutorial
3. Exercise

MAS T2P1 Tutorials Series
Nested list basics



Exercises

Questions

exercise 1

1-0.
Using arr3d output from
3d array with item
component...

1-1.
Pick up one point from the
array by specifying x, y and
z index.

exercise 2

2-0.
Using the box_arr3d output
from
3d array with tessellation
component...

2-1.
Pick up a group of boxes from
the array by specifying x, y
and a start index and its
length (like slicing)

exercise 3

3-0.
Using the box_arr3d output
from
3d array with tessellation
component...

3-1.
Add some lines to the
component above to avoid
"IndexError": list index out
of range.

3-2.
Return None instead, and
print some message.

exercise 4

4-0.
By customizing
3d array with geometry...

4-1.
Make a 3d array with boxes
which have different
edge lengths for x, y and z,
and change the proportion of
the boxes

exercise 5

5-0.
From scratch...

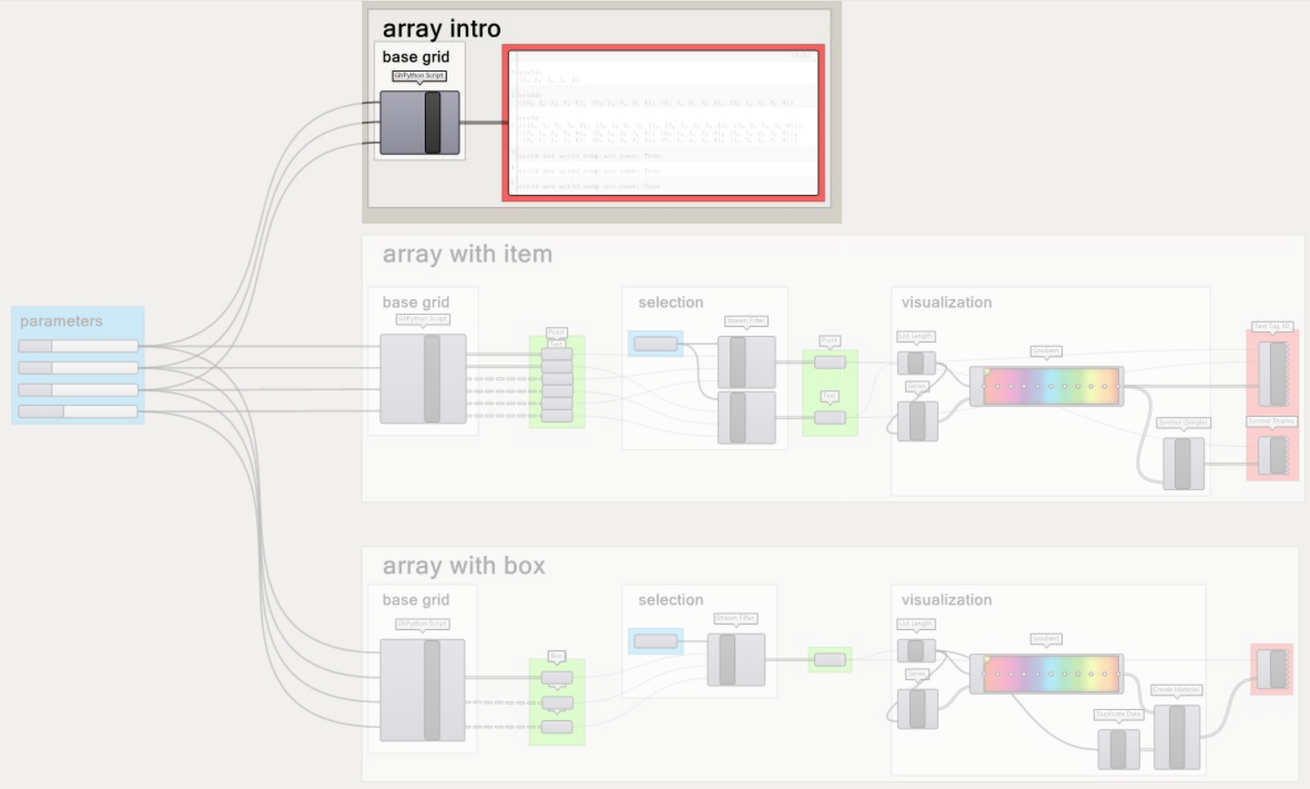
5-1.
Make a 3d array with boxes
whose proportion is changed
probabilistically.

Answers

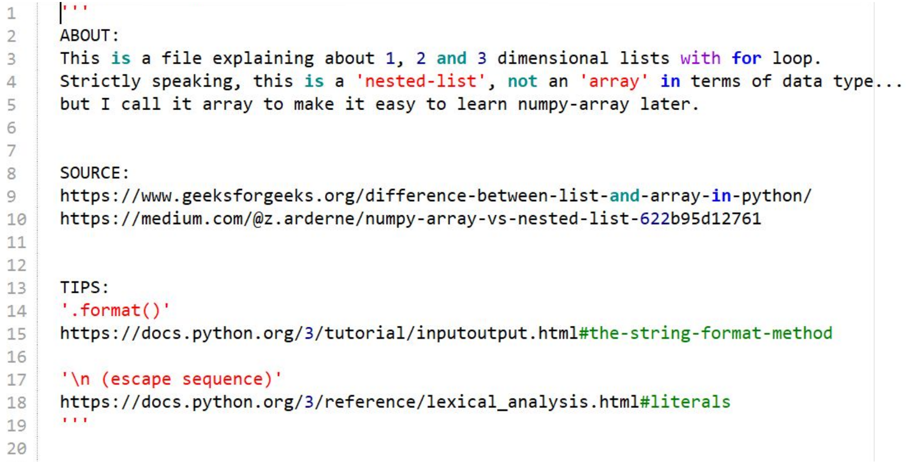
Examples

1. Legend
2. Tutorial
3. Exercise

MAS T2P1 Tutorials Series
Nested list basics



```
arr1d:
[0, 1, 2, 3, 4]

arr2d:
[[0, 1, 2, 3, 4], [0, 1, 2, 3, 4], [0, 1, 2, 3, 4], [0, 1, 2, 3, 4]]

arr3d:
[[[0, 1, 2, 3, 4], [0, 1, 2, 3, 4], [0, 1, 2, 3, 4], [0, 1, 2, 3, 4]],
[[0, 1, 2, 3, 4], [0, 1, 2, 3, 4], [0, 1, 2, 3, 4], [0, 1, 2, 3, 4]],
[[0, 1, 2, 3, 4], [0, 1, 2, 3, 4], [0, 1, 2, 3, 4], [0, 1, 2, 3, 4]]]

arr1d and arr1d_comp are same: True

arr2d and arr2d_comp are same: True

arr3d and arr3d_comp are same: True
```



```
'''
ABOUT:
This is a file explaining about 1, 2 and 3 dimensional lists with for loop.
Strictly speaking, this is a 'nested-list', not an 'array' in terms of data type...
but I call it array to make it easy to learn numpy-array later.


SOURCE:
https://www.geeksforgeeks.org/difference-between-list-and-array-in-python/
https://medium.com/@z.arderne/numpy-array-vs-nested-list-622b95d12761


TIPS:
'.format()'
https://docs.python.org/3/tutorial/inputoutput.html#the-string-format-method

'\n (escape sequence)'
https://docs.python.org/3/reference/lexical_analysis.html#literals
'''

```

Short description of this file

Source link I refer to prepare this file

Python tips in the code

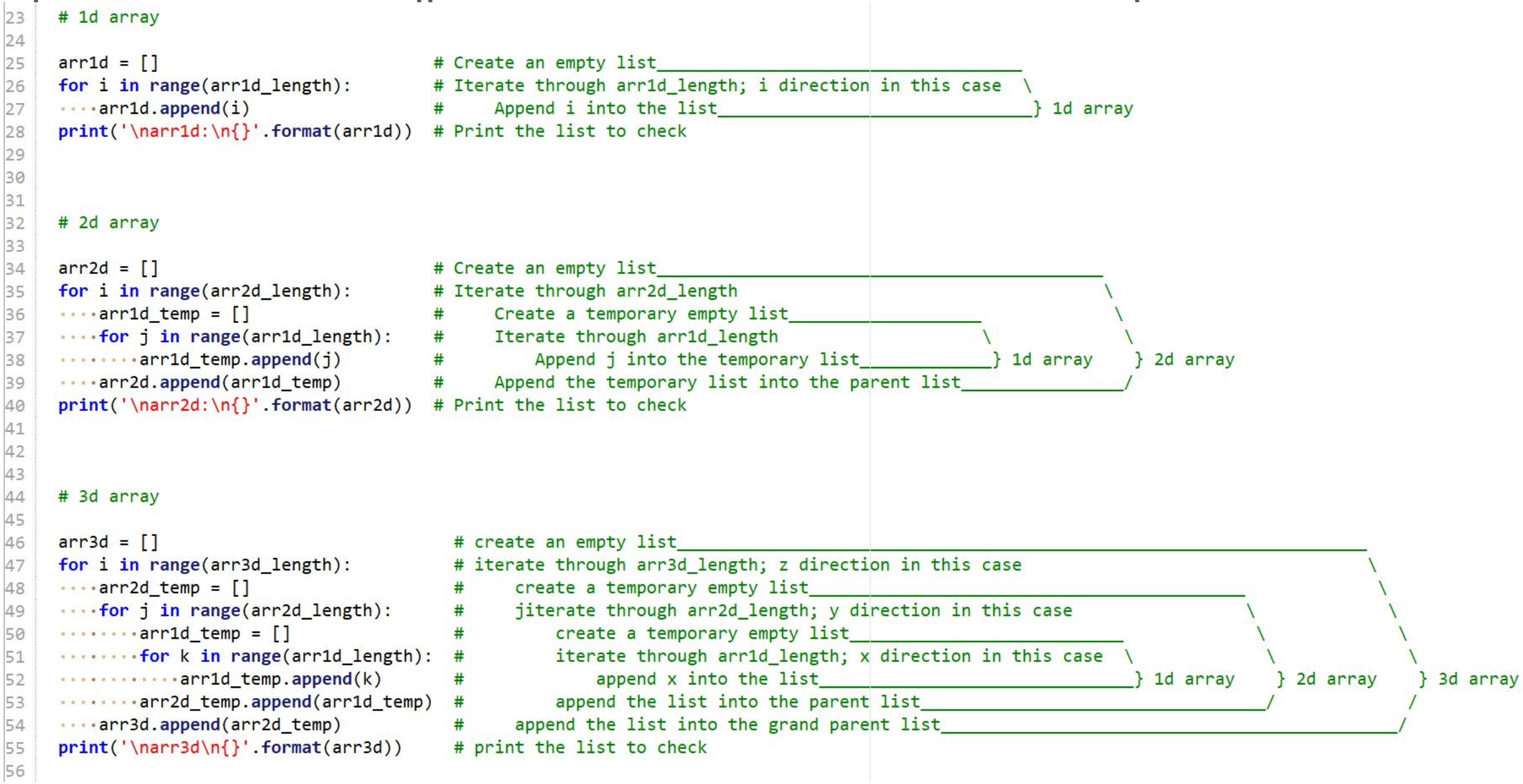
LIST vs ARRAY

docs.python.org/3/library/stdtypes.html
#sequence-types-list-tuple-range

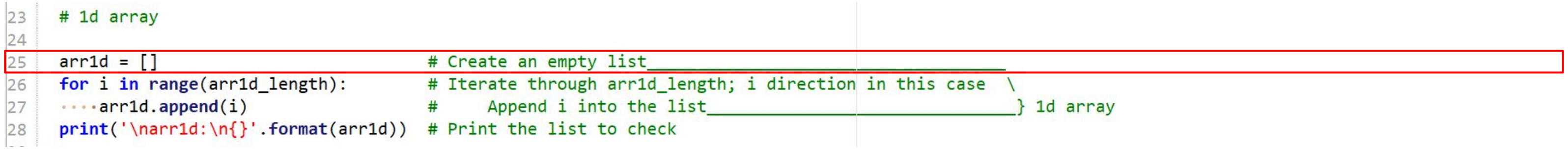
docs.python.org/3/library/array.html

Actual code

Line by line explanation

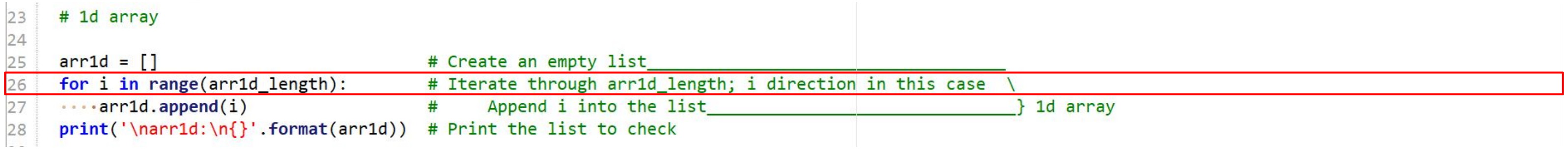


```
# 1d array

arr1d = []                           # Create an empty list____________________________________
for i in range(arr1d_length):        # Iterate through arr1d_length; i direction in this case  \
    arr1d.append(i)                  #     Append i into the list_______________________________} 1d array
print('\narr1d:\n{}'.format(arr1d))   # Print the list to check



# 2d array

arr2d = []                           # Create an empty list____________________________________________
for i in range(arr2d_length):        # Iterate through arr2d_length                                    \
    arr1d_temp = []                  #     Create a temporary empty list___________________             \
    for j in range(arr1d_length):    #     Iterate through arr1d_length                    \             \
        arr1d_temp.append(j)         #         Append j into the temporary list_____________} 1d array    } 2d array
    arr2d.append(arr1d_temp)         #     Append the temporary list into the parent list_________________/
print('\narr2d:\n{}'.format(arr2d))   # Print the list to check



# 3d array

arr3d = []                             # create an empty list____________________________________________________________________
for i in range(arr3d_length):          # iterate through arr3d_length; z direction in this case                                  \
    arr2d_temp = []                    #     create a temporary empty list___________________________________________             \
    for j in range(arr2d_length):      #     jiterate through arr2d_length; y direction in this case                \             \
        arr1d_temp = []                #         create a temporary empty list___________________________             \             \
        for k in range(arr1d_length):  #         iterate through arr1d_length; x direction in this case  \             \             \
            arr1d_temp.append(k)       #             append x into the list_______________________________} 1d array    } 2d array    } 3d array
        arr2d_temp.append(arr1d_temp)  #         append the list into the parent list___________________________________/             /
    arr3d.append(arr2d_temp)           #     append the list into the grand parent list_______________________________________________/
print('\narr3d\n{}'.format(arr3d))      # print the list to check
```



```
# 1d array

arr1d = []                         # Create an empty list____________________________________
for i in range(arr1d_length):      # Iterate through arr1d_length; i direction in this case  \
····arr1d.append(i)                #     Append i into the list_______________________________} 1d array
print('\narr1d:\n{}'.format(arr1d))  # Print the list to check
```

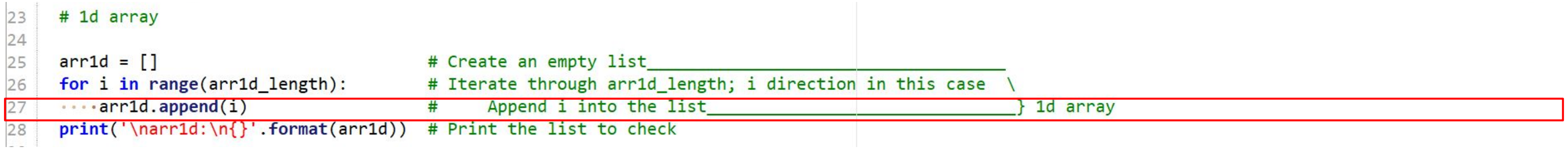
arr1d [ ]



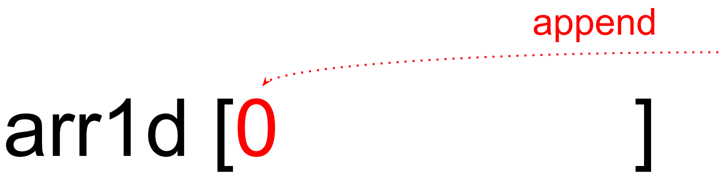
```
# 1d array

arr1d = []                           # Create an empty list____________________________________
for i in range(arr1d_length):        # Iterate through arr1d_length; i direction in this case  \
    arr1d.append(i)                  #     Append i into the list______________________________} 1d array
print('\narr1d:\n{}'.format(arr1d))  # Print the list to check
```

i will iterate from 0 to 4

arr1d [ ]



```
# 1d array

arr1d = []                           # Create an empty list____________________________________
for i in range(arr1d_length):        # Iterate through arr1d_length; i direction in this case  \
    arr1d.append(i)                  #     Append i into the list_______________________________} 1d array
print('\narr1d:\n{}'.format(arr1d))  # Print the list to check
```

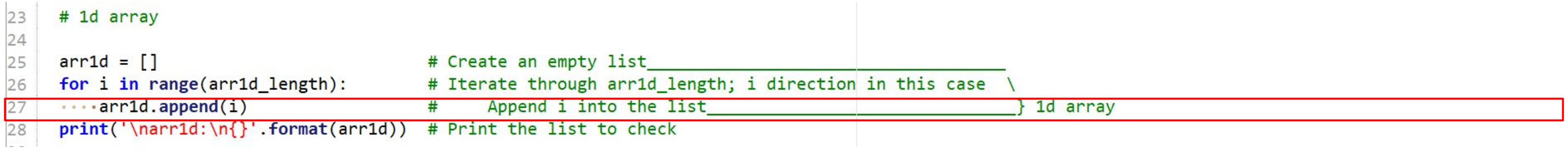


append

arr1d [0]

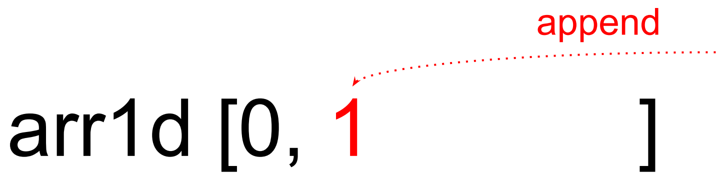
i will iterate from 0 to 4

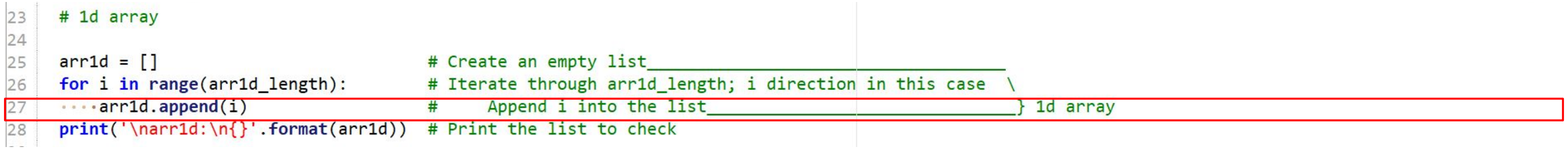
i is 0 in the 1st loop.



```
# 1d array

arr1d = []                          # Create an empty list____________________________________
for i in range(arr1d_length):       # Iterate through arr1d_length; i direction in this case  \
····arr1d.append(i)                 #     Append i into the list______________________________} 1d array
print('\narr1d:\n{}'.format(arr1d)) # Print the list to check
```

i will iterate from 0 to 4
i is 0 in the 1st loop.
i is 1 in the 2nd loop.





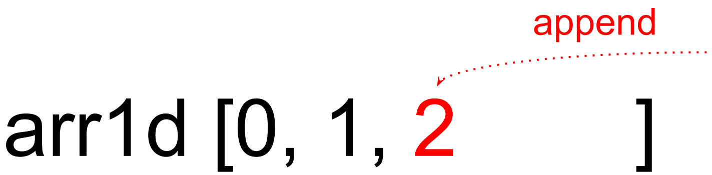
```
# 1d array

arr1d = []                           # Create an empty list____________________________________
for i in range(arr1d_length):        # Iterate through arr1d_length; i direction in this case  \
    arr1d.append(i)                  #     Append i into the list_______________________________} 1d array
print('\narr1d:\n{}'.format(arr1d))  # Print the list to check
```

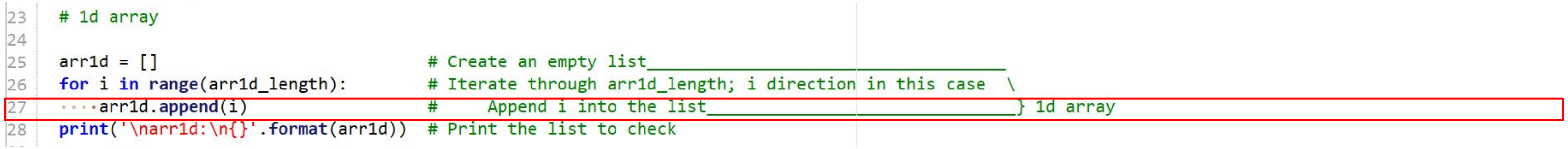
i will iterate from 0 to 4

i is 0 in the 1st loop.

i is 1 in the 2nd loop.

i is 2 in the 3rd loop.



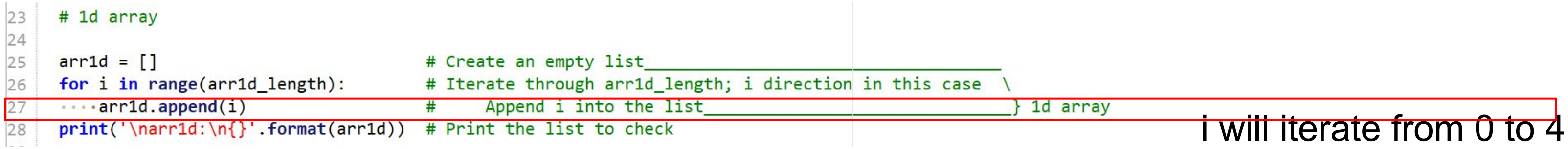


```
# 1d array

arr1d = []                          # Create an empty list_____________________________________
for i in range(arr1d_length):       # Iterate through arr1d_length; i direction in this case  \
    arr1d.append(i)                 #     Append i into the list______________________________} 1d array
print('\narr1d:\n{}'.format(arr1d)) # Print the list to check
```

i will iterate from 0 to 4
i is 0 in the 1st loop.
i is 1 in the 2nd loop.
i is 2 in the 3rd loop.
i is 3 in the 4th loop.

append

arr1d [0, 1, 2, 3   ]

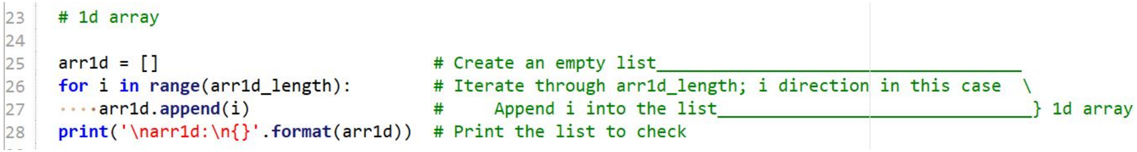


```
# 1d array

arr1d = []                            # Create an empty list____________________________________
for i in range(arr1d_length):         # Iterate through arr1d_length; i direction in this case  \
····arr1d.append(i)                   #     Append i into the list______________________________} 1d array
print('\narr1d:\n{}'.format(arr1d))  # Print the list to check
```

i will iterate from 0 to 4
i is 0 in the 1st loop.
i is 1 in the 2nd loop.
i is 2 in the 3rd loop.
i is 3 in the 4th loop.
i is 4 in the 5th loop.

append

arr1d [0, 1, 2, 3, 4]

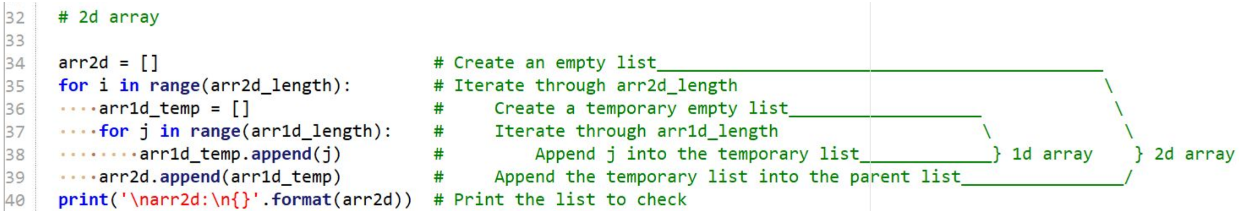


```
# 1d array

arr1d = []                           # Create an empty list_____________________________________
for i in range(arr1d_length):        # Iterate through arr1d_length; i direction in this case  \
    arr1d.append(i)                  #     Append i into the list_______________________________} 1d array
print('\narr1d:\n{}'.format(arr1d))  # Print the list to check
```

i will iterate from 0 to 4
i is 0 in the 1st loop.
i is 1 in the 2nd loop.
i is 2 in the 3rd loop.
i is 3 in the 4th loop.
i is 4 in the 5th loop.

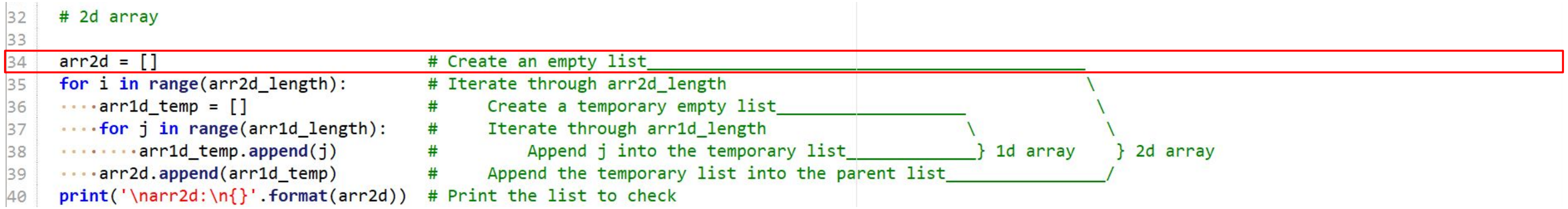
arr1d [0, 1, 2, 3, 4]

MAS T2P1 Tutorials Series
Nested list basics



```
# 2d array

arr2d = []                           # Create an empty list____________________________________________
for i in range(arr2d_length):        # Iterate through arr2d_length                                     \
    arr1d_temp = []                  #     Create a temporary empty list___________________              \
    for j in range(arr1d_length):    #     Iterate through arr1d_length                     \             \
        arr1d_temp.append(j)         #         Append j into the temporary list______________} 1d array    } 2d array
    arr2d.append(arr1d_temp)         #     Append the temporary list into the parent list_________________/
print('\narr2d:\n{}'.format(arr2d))  # Print the list to check
```

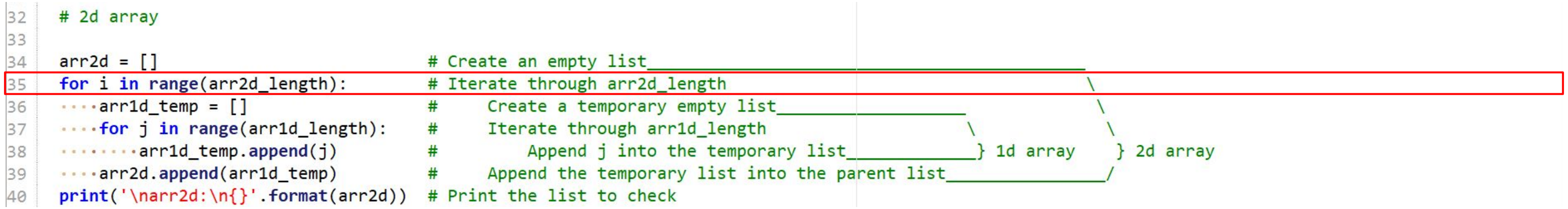
MAS T2P1 Tutorials Series
Nested list basics



```
# 2d array

arr2d = []                           # Create an empty list_____________________________________________
for i in range(arr2d_length):        # Iterate through arr2d_length                                     \
    arr1d_temp = []                  #     Create a temporary empty list___________________              \
    for j in range(arr1d_length):    #     Iterate through arr1d_length                     \             \
        arr1d_temp.append(j)         #         Append j into the temporary list______________} 1d array    } 2d array
    arr2d.append(arr1d_temp)         #     Append the temporary list into the parent list________________/
print('\narr2d:\n{}'.format(arr2d))  # Print the list to check
```

arr2d [ ]

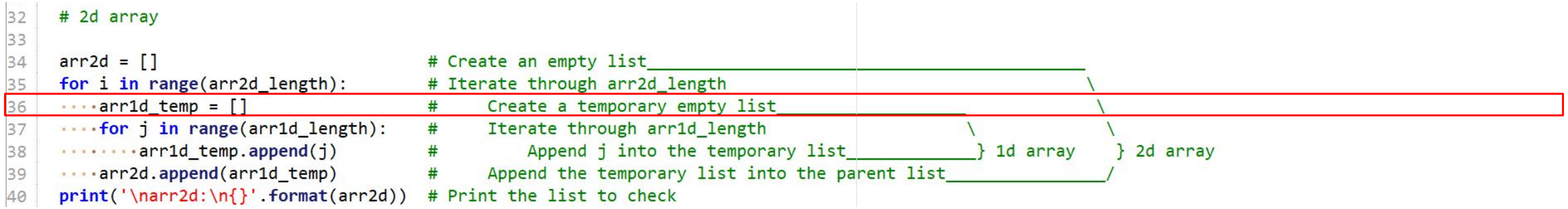
MAS T2P1 Tutorials Series
Nested list basics



```
# 2d array

arr2d = []                          # Create an empty list______________________________________________
for i in range(arr2d_length):       # Iterate through arr2d_length                                      \
    arr1d_temp = []                 #     Create a temporary empty list___________________               \
    for j in range(arr1d_length):   #     Iterate through arr1d_length                    \               \
        arr1d_temp.append(j)        #         Append j into the temporary list____________} 1d array    } 2d array
    arr2d.append(arr1d_temp)        #     Append the temporary list into the parent list________________/
print('\narr2d:\n{}'.format(arr2d)) # Print the list to check
```

i will iterate from 0 to 3 (i is 0)

arr2d [ ]

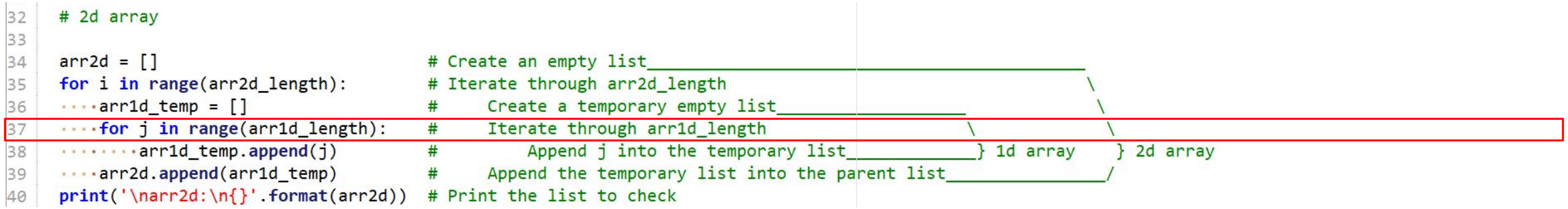


```
# 2d array

arr2d = []                             # Create an empty list____________________________________________
for i in range(arr2d_length):          # Iterate through arr2d_length                                    \
    arr1d_temp = []                    #     Create a temporary empty list___________________             \
    for j in range(arr1d_length):      #     Iterate through arr1d_length                    \             \
        arr1d_temp.append(j)           #         Append j into the temporary list_____________} 1d array    } 2d array
    arr2d.append(arr1d_temp)           #     Append the temporary list into the parent list________________/
print('\narr2d:\n{}'.format(arr2d))  # Print the list to check
```

arr1d_temp [ ]

i will iterate from 0 to 3 (i is 0)

arr2d [ ]



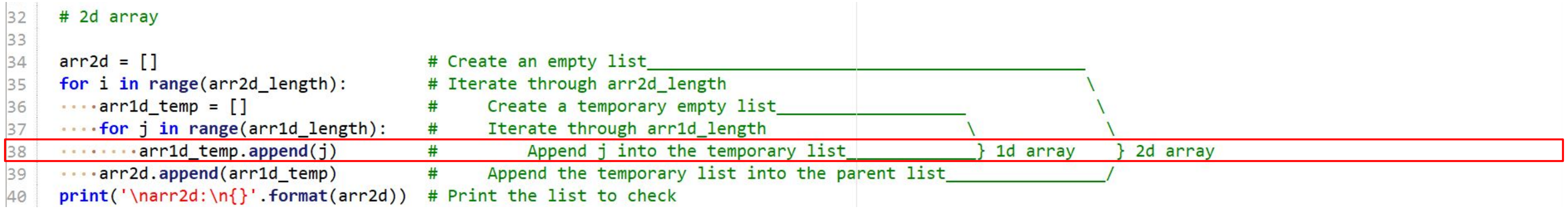
```
# 2d array

arr2d = []                           # Create an empty list____________________________________________
for i in range(arr2d_length):        # Iterate through arr2d_length                                     \
    arr1d_temp = []                  #     Create a temporary empty list___________________              \
    for j in range(arr1d_length):    #     Iterate through arr1d_length                     \             \
        arr1d_temp.append(j)         #         Append j into the temporary list_____________} 1d array     } 2d array
    arr2d.append(arr1d_temp)         #     Append the temporary list into the parent list________________/
print('\narr2d:\n{}'.format(arr2d))  # Print the list to check
```

j will iterate from 0 to 4

arr1d_temp [ ]

i will iterate from 0 to 3 (i is 0)

arr2d [ ]



```
# 2d array

arr2d = []                           # Create an empty list____________________________________________
for i in range(arr2d_length):        # Iterate through arr2d_length                                    \
    arr1d_temp = []                  #     Create a temporary empty list___________________             \
    for j in range(arr1d_length):    #     Iterate through arr1d_length                    \             \
        arr1d_temp.append(j)         #         Append j into the temporary list_____________} 1d array    } 2d array
    arr2d.append(arr1d_temp)         #     Append the temporary list into the parent list________________/
print('\narr2d:\n{}'.format(arr2d))  # Print the list to check
```

j will iterate from 0 to 4

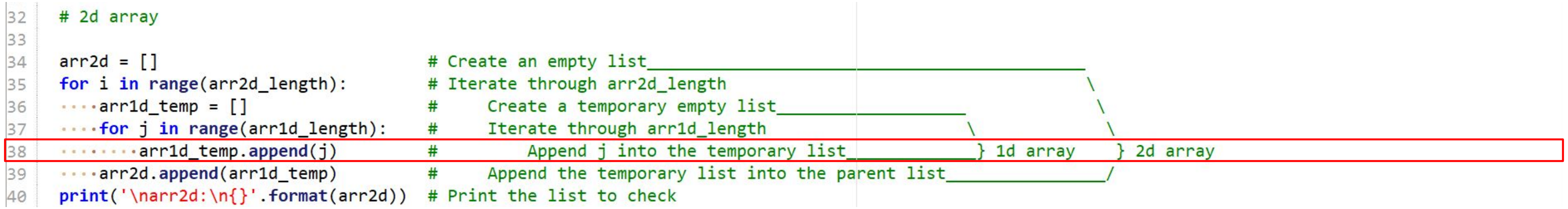
append

j is 0 in the 1st loop

arr1d_temp [0]

i will iterate from 0 to 3 (i is 0)

arr2d [ ]



```
# 2d array

arr2d = []                          # Create an empty list____________________________________________
for i in range(arr2d_length):       # Iterate through arr2d_length                                     \
    arr1d_temp = []                 #     Create a temporary empty list___________________              \
    for j in range(arr1d_length):   #     Iterate through arr1d_length                    \              \
        arr1d_temp.append(j)        #         Append j into the temporary list_____________} 1d array     } 2d array
    arr2d.append(arr1d_temp)        #     Append the temporary list into the parent list________________/
print('\narr2d:\n{}'.format(arr2d))  # Print the list to check
```

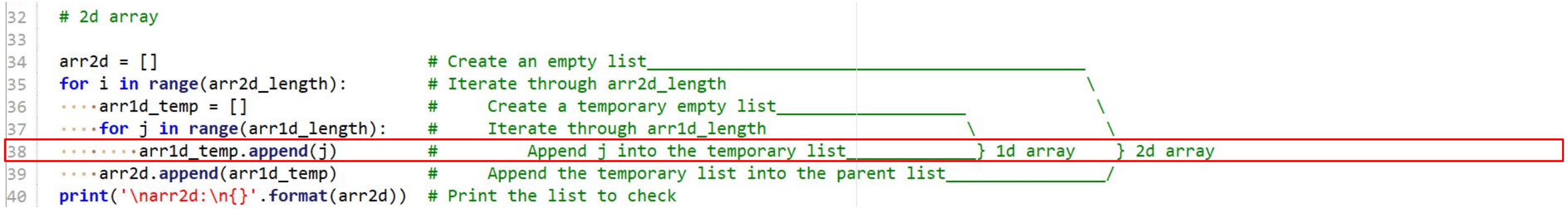
j will iterate from 0 to 4
j is 0 in the 1st loop
j is 1 in the 2nd loop

append

arr1d_temp [0, 1]

i will iterate from 0 to 3 (i is 0)

arr2d [ ]



```
# 2d array

arr2d = []                          # Create an empty list___________________________________________
for i in range(arr2d_length):       # Iterate through arr2d_length                                   \
    arr1d_temp = []                 #     Create a temporary empty list___________________            \
    for j in range(arr1d_length):   #     Iterate through arr1d_length                    \            \
        arr1d_temp.append(j)        #         Append j into the temporary list____________} 1d array    } 2d array
    arr2d.append(arr1d_temp)        #     Append the temporary list into the parent list_____________/
print('\narr2d:\n{}'.format(arr2d)) # Print the list to check
```

j will iterate from 0 to 4

j is 0 in the 1st loop

j is 1 in the 2nd loop

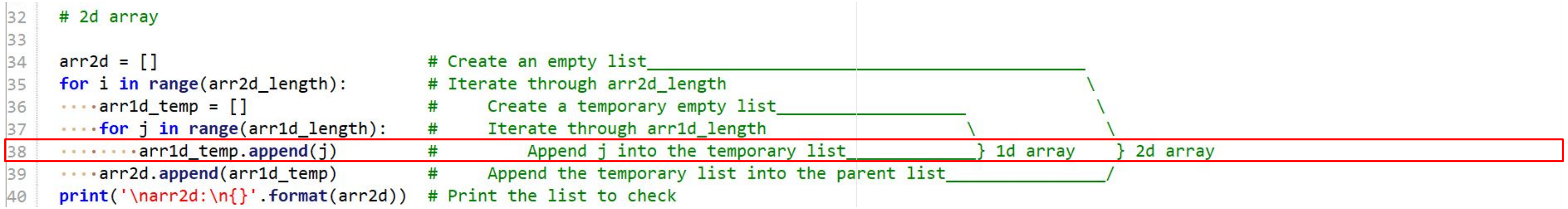
append

j is 2 in the 3rd loop

arr1d_temp [0, 1, 2]

i will iterate from 0 to 3 (i is 0)

arr2d [ ]



```
# 2d array

arr2d = []                           # Create an empty list____________________________________________
for i in range(arr2d_length):        # Iterate through arr2d_length                                    \
    arr1d_temp = []                  #     Create a temporary empty list___________________             \
    for j in range(arr1d_length):    #     Iterate through arr1d_length                    \             \
        arr1d_temp.append(j)         #         Append j into the temporary list_____________} 1d array    } 2d array
    arr2d.append(arr1d_temp)         #     Append the temporary list into the parent list_________________/
print('\narr2d:\n{}'.format(arr2d))  # Print the list to check
```

j will iterate from 0 to 4

j is 0 in the 1st loop

j is 1 in the 2nd loop

j is 2 in the 3rd loop

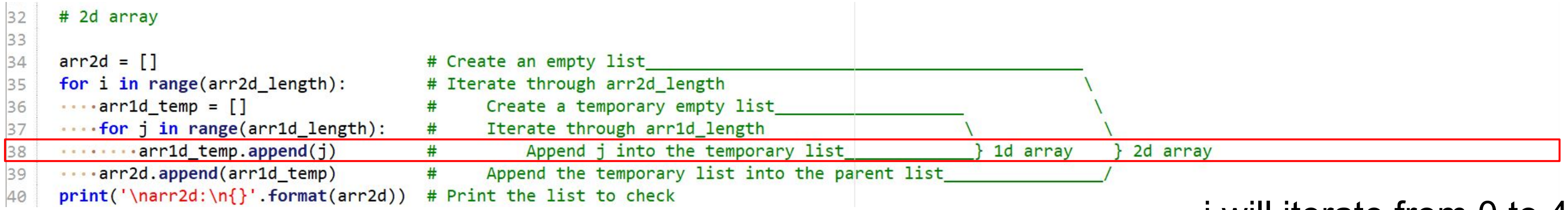
j is 3 in the 4th loop

append

arr1d_temp [0, 1, 2, 3]

i will iterate from 0 to 3 (i is 0)

arr2d [ ]



```
# 2d array

arr2d = []                           # Create an empty list____________________________________________
for i in range(arr2d_length):        # Iterate through arr2d_length                                    \
    arr1d_temp = []                  #     Create a temporary empty list___________________             \
    for j in range(arr1d_length):    #     Iterate through arr1d_length                    \             \
        arr1d_temp.append(j)         #         Append j into the temporary list_____________} 1d array    } 2d array
    arr2d.append(arr1d_temp)         #     Append the temporary list into the parent list________________/
print('\narr2d:\n{}'.format(arr2d))  # Print the list to check
```

j will iterate from 0 to 4

j is 0 in the 1st loop

j is 1 in the 2nd loop

j is 2 in the 3rd loop

j is 3 in the 4th loop

append

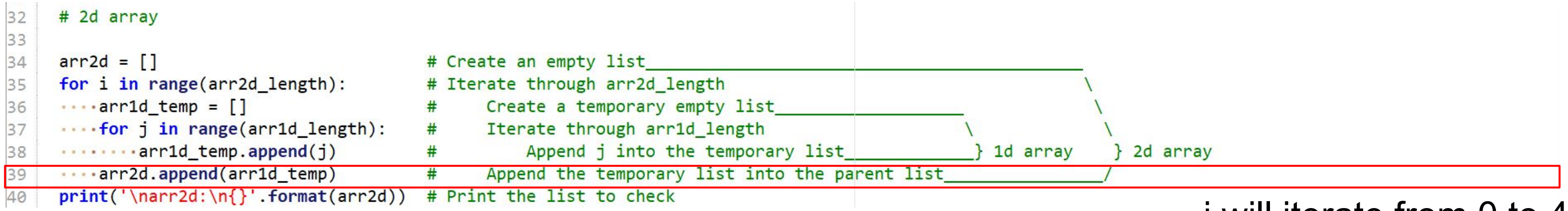
j is 4 in the 5th loop

arr1d_temp [0, 1, 2, 3, 4]

i will iterate from 0 to 3 (i is 0)

arr2d [ ]

MAS T2P1 Tutorials Series
Nested list basics



```
# 2d array

arr2d = []                           # Create an empty list____________________________________________
for i in range(arr2d_length):        # Iterate through arr2d_length                                    \
    arr1d_temp = []                  #     Create a temporary empty list___________________             \
    for j in range(arr1d_length):    #     Iterate through arr1d_length                   \             \
        arr1d_temp.append(j)         #         Append j into the temporary list____________} 1d array    } 2d array
    arr2d.append(arr1d_temp)         #     Append the temporary list into the parent list________________/
print('\narr2d:\n{}'.format(arr2d))  # Print the list to check
```

j will iterate from 0 to 4
j is 0 in the 1st loop
j is 1 in the 2nd loop
j is 2 in the 3rd loop
j is 3 in the 4th loop
j is 4 in the 5th loop

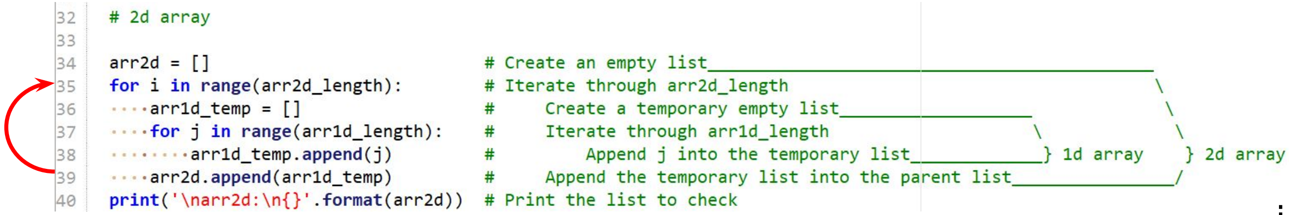
arr1d_temp [0, 1, 2, 3, 4]

append to the index 0
(= i)

i will iterate from 0 to 3 (i is 0)

arr2d [[0,1,2,3,4]]

MAS T2P1 Tutorials Series
Nested list basics



```
# 2d array

arr2d = []                           # Create an empty list_____________________________________________
for i in range(arr2d_length):        # Iterate through arr2d_length                                     \
    arr1d_temp = []                  #     Create a temporary empty list____________________             \
    for j in range(arr1d_length):    #     Iterate through arr1d_length                     \             \
        arr1d_temp.append(j)         #         Append j into the temporary list______________} 1d array    } 2d array
    arr2d.append(arr1d_temp)         #     Append the temporary list into the parent list_________________/
print('\narr2d:\n{}'.format(arr2d))  # Print the list to check
```

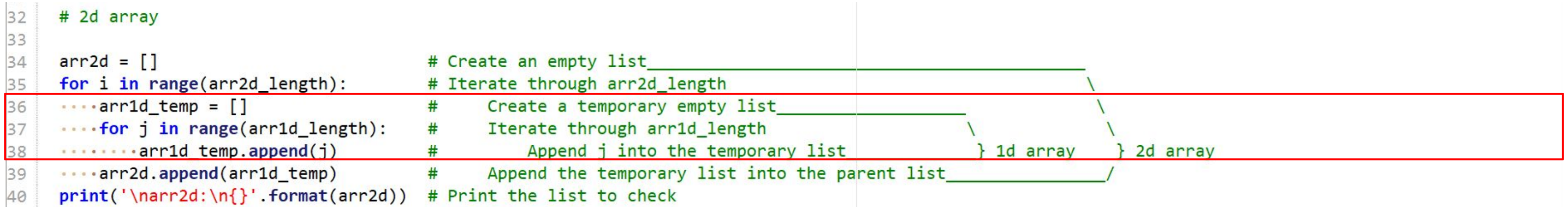
j will iterate from 0 to 4
j is 0 in the 1st loop
j is 1 in the 2nd loop
j is 2 in the 3rd loop
j is 3 in the 4th loop
j is 4 in the 5th loop

arr1d_temp [0, 1, 2, 3, 4]

i will iterate from 0 to 3 (i is 1)

arr2d [[0,1,2,3,4]   ]

MAS T2P1 Tutorials Series
Nested list basics



```
# 2d array

arr2d = []                           # Create an empty list____________________________________________
for i in range(arr2d_length):        # Iterate through arr2d_length                                     \
    arr1d_temp = []                  #     Create a temporary empty list___________________              \
    for j in range(arr1d_length):    #     Iterate through arr1d_length                     \             \
        arr1d_temp.append(j)         #         Append j into the temporary list_____________} 1d array    } 2d array
    arr2d.append(arr1d_temp)         #     Append the temporary list into the parent list_________________/
print('\narr2d:\n{}'.format(arr2d))  # Print the list to check
```

append

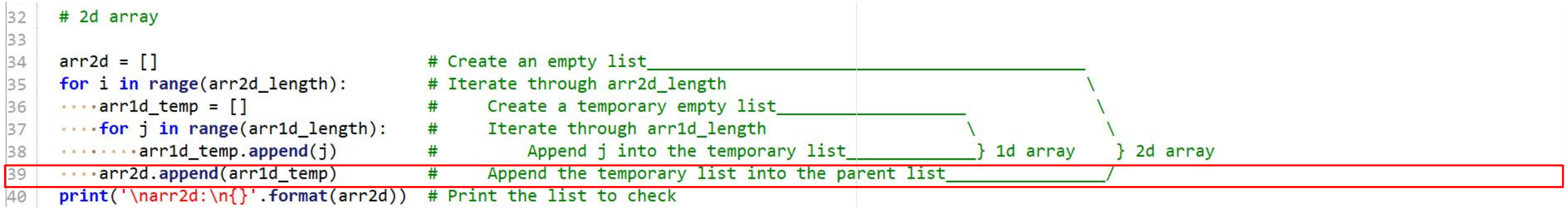
j will iterate from 0 to 4

arr1d_temp [0, 1, 2, 3, 4]

i will iterate from 0 to 3 (i is 1)

arr2d [[0,1,2,3,4]   ]

MAS T2P1 Tutorials Series
Nested list basics



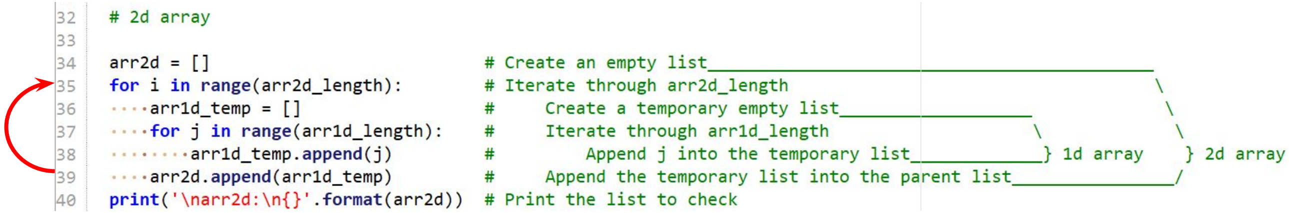
```
# 2d array

arr2d = []                          # Create an empty list__________________________________________
for i in range(arr2d_length):       # Iterate through arr2d_length                                   \
    arr1d_temp = []                 #     Create a temporary empty list_________________              \
    for j in range(arr1d_length):   #     Iterate through arr1d_length                  \             \
        arr1d_temp.append(j)        #         Append j into the temporary list____________} 1d array    } 2d array
    arr2d.append(arr1d_temp)        #     Append the temporary list into the parent list______________/
print('\narr2d:\n{}'.format(arr2d))  # Print the list to check
```

append to the index 1
(= i)

arr1d_temp [0, 1, 2, 3, 4]

i will iterate from 0 to 3 (i is 1)

arr2d [[0,1,2,3,4],[0,1,2,3,4]]



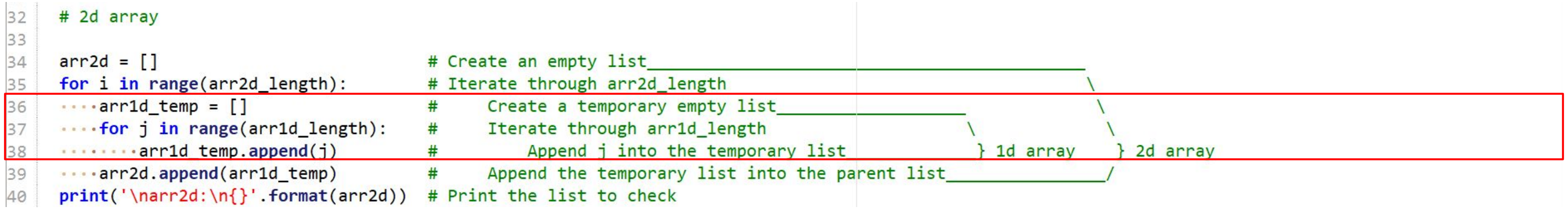
```
# 2d array

arr2d = []                           # Create an empty list____________________________________________
for i in range(arr2d_length):        # Iterate through arr2d_length                                    \
    arr1d_temp = []                  #     Create a temporary empty list___________________             \
    for j in range(arr1d_length):    #     Iterate through arr1d_length                    \             \
        arr1d_temp.append(j)         #         Append j into the temporary list_____________} 1d array    } 2d array
    arr2d.append(arr1d_temp)         #     Append the temporary list into the parent list________________/
print('\narr2d:\n{}'.format(arr2d))  # Print the list to check
```

arr1d_temp [0, 1, 2, 3, 4]

i will iterate from 0 to 3 (i is 2)

arr2d [[0,1,2,3,4],[0,1,2,3,4]   ]

MAS T2P1 Tutorials Series
Nested list basics



```
# 2d array

arr2d = []                           # Create an empty list____________________________________________
for i in range(arr2d_length):        # Iterate through arr2d_length                                    \
    arr1d_temp = []                  #     Create a temporary empty list___________________             \
    for j in range(arr1d_length):    #     Iterate through arr1d_length                    \             \
        arr1d_temp.append(j)         #         Append j into the temporary list_____________} 1d array    } 2d array
    arr2d.append(arr1d_temp)         #     Append the temporary list into the parent list________________/
print('\narr2d:\n{}'.format(arr2d))  # Print the list to check
```

append

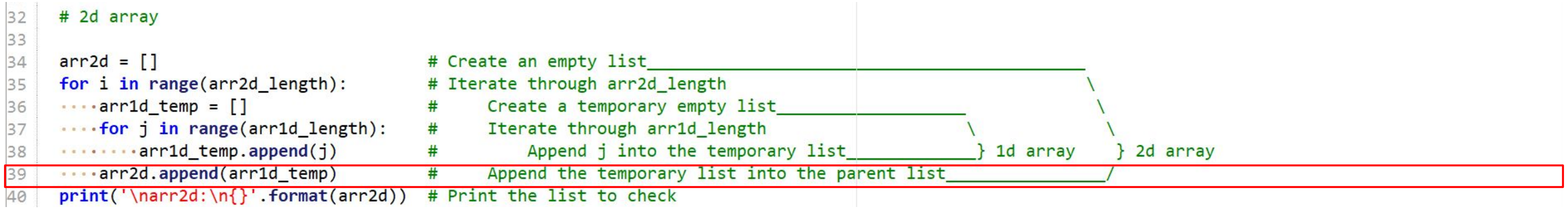
j will iterate from 0 to 4

arr1d_temp [0, 1, 2, 3, 4]

i will iterate from 0 to 3 (i is 2)

arr2d [[0,1,2,3,4],[0,1,2,3,4]]

MAS T2P1 Tutorials Series
Nested list basics



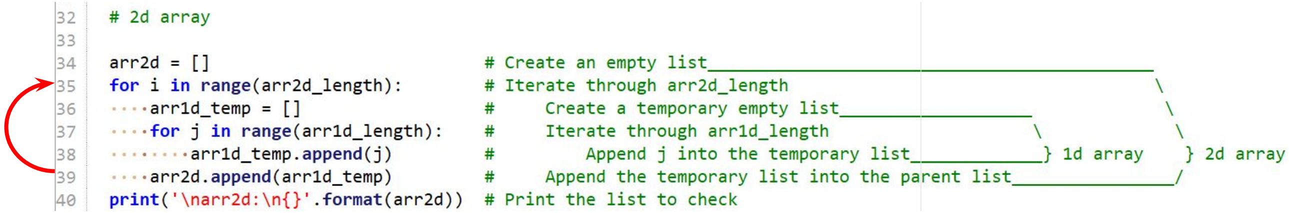
```
# 2d array

arr2d = []                            # Create an empty list____________________________________________
for i in range(arr2d_length):         # Iterate through arr2d_length                                    \
    arr1d_temp = []                   #     Create a temporary empty list___________________             \
    for j in range(arr1d_length):     #     Iterate through arr1d_length                     \            \
        arr1d_temp.append(j)          #         Append j into the temporary list_____________} 1d array   } 2d array
    arr2d.append(arr1d_temp)          #     Append the temporary list into the parent list_______________/
print('\narr2d:\n{}'.format(arr2d))   # Print the list to check
```

append to the index 2
(= i)

arr1d_temp [0, 1, 2, 3, 4]

i will iterate from 0 to 3 (i is 2)

arr2d [[0,1,2,3,4],[0,1,2,3,4],[0,1,2,3,4]]



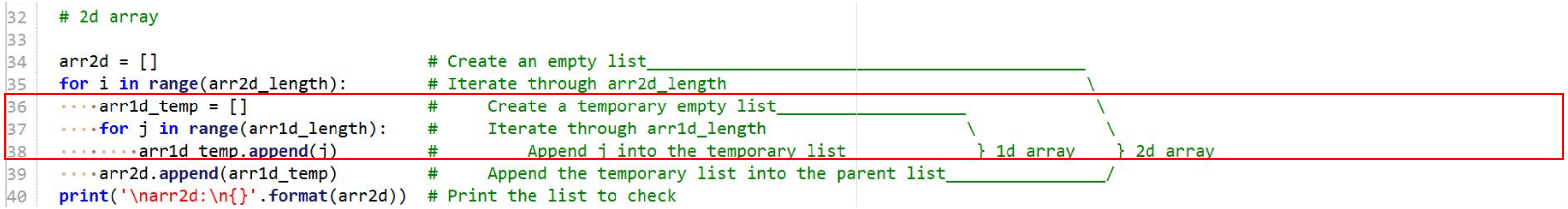
```
# 2d array

arr2d = []                           # Create an empty list____________________________________________
for i in range(arr2d_length):        # Iterate through arr2d_length                                    \
    arr1d_temp = []                  #     Create a temporary empty list___________________             \
    for j in range(arr1d_length):    #     Iterate through arr1d_length                    \             \
        arr1d_temp.append(j)         #         Append j into the temporary list_____________} 1d array    } 2d array
    arr2d.append(arr1d_temp)         #     Append the temporary list into the parent list________________/
print('\narr2d:\n{}'.format(arr2d))  # Print the list to check
```

arr1d_temp [0, 1, 2, 3, 4]

i will iterate from 0 to 3 (i is 3)

arr2d [[0,1,2,3,4],[0,1,2,3,4],[0,1,2,3,4],   ]

Nested list basics



```
# 2d array

arr2d = []                           # Create an empty list____________________________________________
for i in range(arr2d_length):        # Iterate through arr2d_length                                    \
    arr1d_temp = []                  #     Create a temporary empty list___________________             \
    for j in range(arr1d_length):    #     Iterate through arr1d_length                    \             \
        arr1d_temp.append(j)         #         Append j into the temporary list_____________} 1d array    } 2d array
    arr2d.append(arr1d_temp)         #     Append the temporary list into the parent list________________/
print('\narr2d:\n{}'.format(arr2d))  # Print the list to check
```

append

j will iterate from 0 to 4

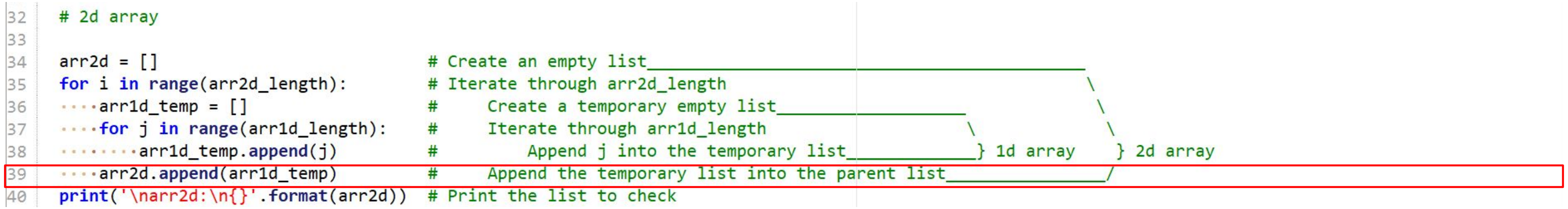
arr1d_temp [0, 1, 2, 3, 4]

i will iterate from 0 to 3 (i is 3)

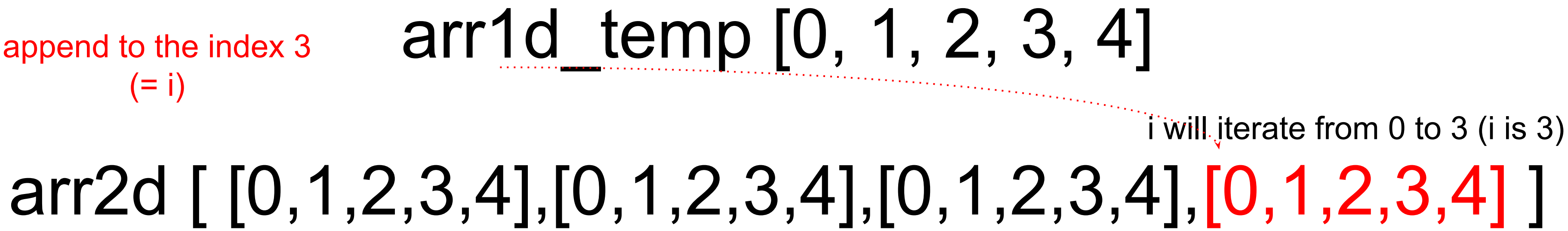
arr2d [[0,1,2,3,4],[0,1,2,3,4],[0,1,2,3,4],]

MAS T2P1 Tutorials Series

Nested list basics



```
# 2d array

arr2d = []                           # Create an empty list____________________________________________
for i in range(arr2d_length):        # Iterate through arr2d_length                                    \
    arr1d_temp = []                  #     Create a temporary empty list___________________             \
    for j in range(arr1d_length):    #     Iterate through arr1d_length                    \             \
        arr1d_temp.append(j)         #         Append j into the temporary list_____________} 1d array    } 2d array
    arr2d.append(arr1d_temp)         #     Append the temporary list into the parent list________________/
print('\narr2d:\n{}'.format(arr2d))  # Print the list to check
```



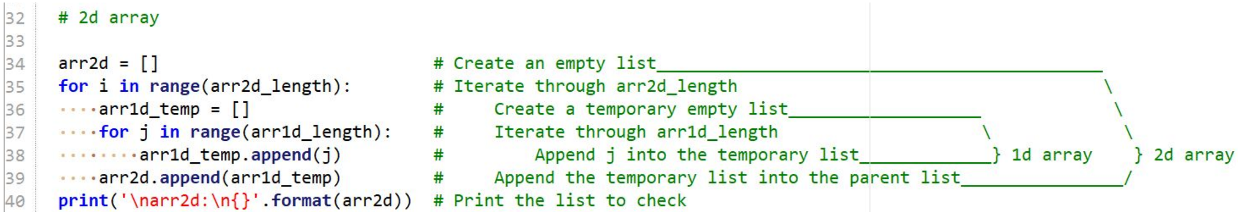
append to the index 3
(= i)

arr1d_temp [0, 1, 2, 3, 4]

i will iterate from 0 to 3 (i is 3)

arr2d [[0,1,2,3,4],[0,1,2,3,4],[0,1,2,3,4],[0,1,2,3,4]]

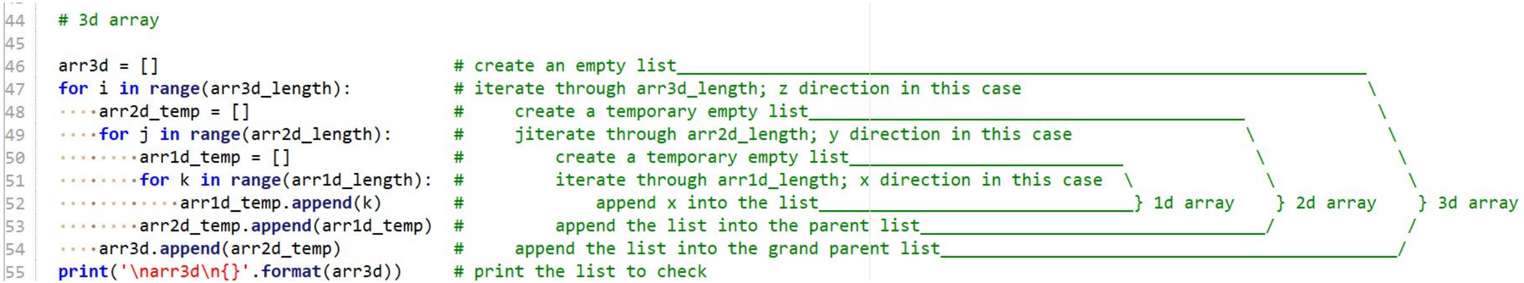
MAS T2P1 Tutorials Series
Nested list basics



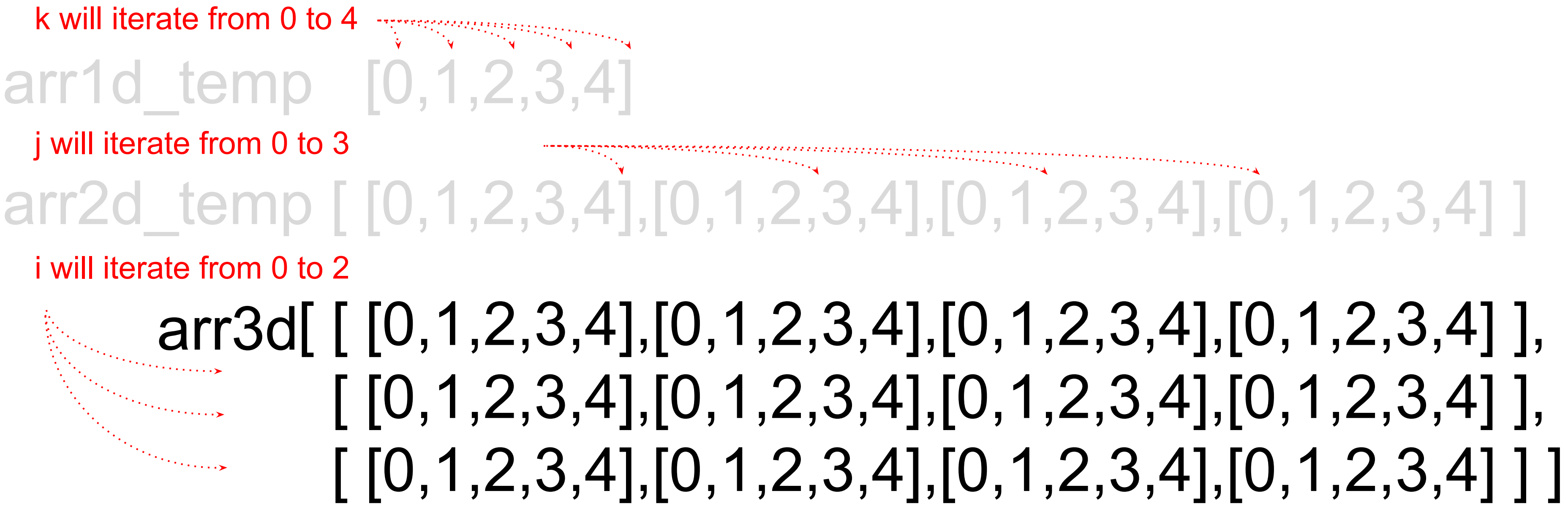
```
# 2d array

arr2d = []                           # Create an empty list_____________________________________________
for i in range(arr2d_length):        # Iterate through arr2d_length                                     \
    arr1d_temp = []                  #     Create a temporary empty list____________________             \
    for j in range(arr1d_length):    #     Iterate through arr1d_length                     \             \
        arr1d_temp.append(j)         #         Append j into the temporary list______________} 1d array    } 2d array
    arr2d.append(arr1d_temp)         #     Append the temporary list into the parent list________________/
print('\narr2d:\n{}'.format(arr2d))  # Print the list to check
```

arr2d [[0,1,2,3,4],[0,1,2,3,4],[0,1,2,3,4],[0,1,2,3,4]]

MAS T2P1 Tutorials Series
Nested list basics



```
# 3d array

arr3d = []                             # create an empty list____________________________________________________________________
for i in range(arr3d_length):          # iterate through arr3d_length; z direction in this case                                  \
    arr2d_temp = []                    #     create a temporary empty list___________________________________________             \
    for j in range(arr2d_length):      #     jiterate through arr2d_length; y direction in this case                 \             \
        arr1d_temp = []                #         create a temporary empty list___________________________             \             \
        for k in range(arr1d_length):  #         iterate through arr1d_length; x direction in this case  \             \             \
            arr1d_temp.append(k)       #             append x into the list_______________________________} 1d array    } 2d array    } 3d array
        arr2d_temp.append(arr1d_temp)  #         append the list into the parent list__________________________________/             /
    arr3d.append(arr2d_temp)           #     append the list into the grand parent list_____________________________________________/
print('\narr3d\n{}'.format(arr3d))     # print the list to check
```



k will iterate from 0 to 4

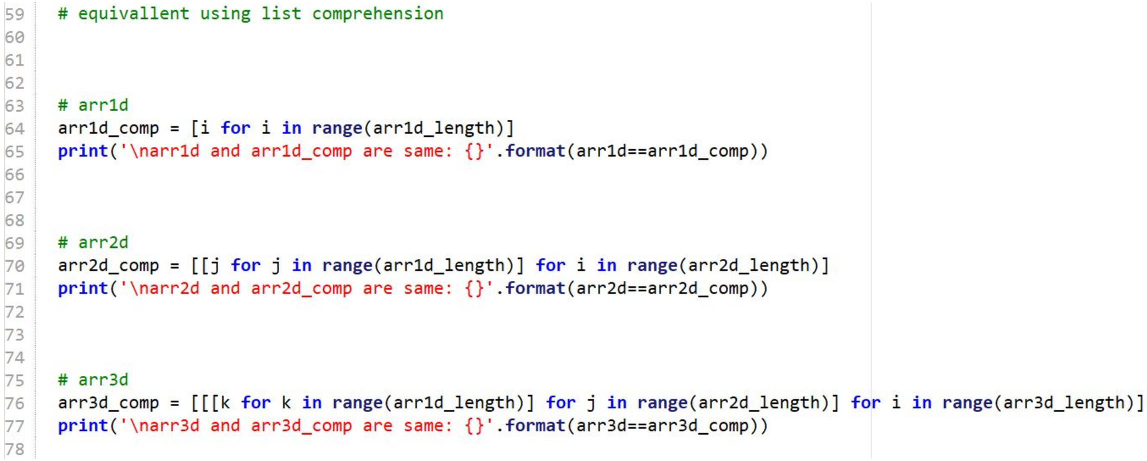
arr1d_temp [0,1,2,3,4]

j will iterate from 0 to 3

arr2d_temp [[0,1,2,3,4],[0,1,2,3,4],[0,1,2,3,4],[0,1,2,3,4]]

i will iterate from 0 to 2

arr3d[[[0,1,2,3,4],[0,1,2,3,4],[0,1,2,3,4],[0,1,2,3,4]],
[[0,1,2,3,4],[0,1,2,3,4],[0,1,2,3,4],[0,1,2,3,4]],
[[0,1,2,3,4],[0,1,2,3,4],[0,1,2,3,4],[0,1,2,3,4]]]

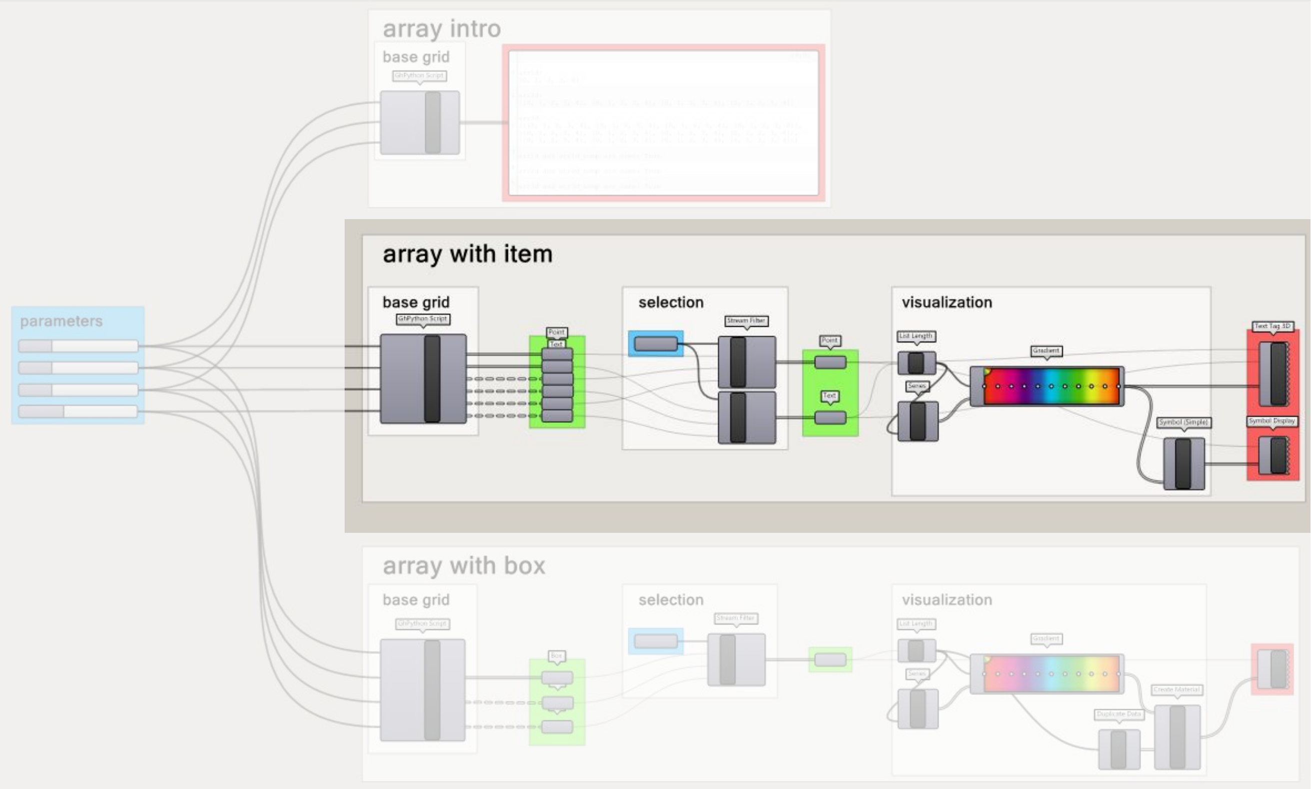


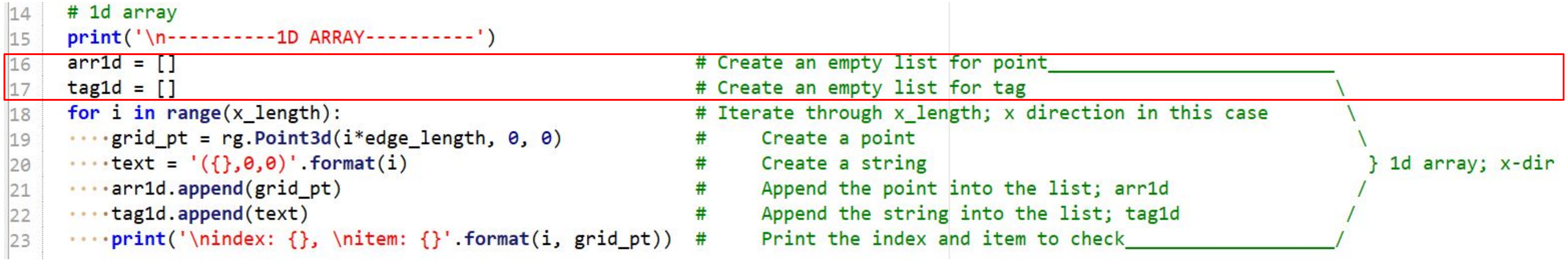
```
# equivallent using list comprehension



# arr1d
arr1d_comp = [i for i in range(arr1d_length)]
print('\narr1d and arr1d_comp are same: {}'.format(arr1d==arr1d_comp))



# arr2d
arr2d_comp = [[j for j in range(arr1d_length)] for i in range(arr2d_length)]
print('\narr2d and arr2d_comp are same: {}'.format(arr2d==arr2d_comp))



# arr3d
arr3d_comp = [[[k for k in range(arr1d_length)] for j in range(arr2d_length)] for i in range(arr3d_length)]
print('\narr3d and arr3d_comp are same: {}'.format(arr3d==arr3d_comp))

```

How can we use this for 3d space…?

```
arr3d[ [ [0,1,2,3,4],[0,1,2,3,4],[0,1,2,3,4],[0,1,2,3,4] ],
       [ [0,1,2,3,4],[0,1,2,3,4],[0,1,2,3,4],[0,1,2,3,4] ],
       [ [0,1,2,3,4],[0,1,2,3,4],[0,1,2,3,4],[0,1,2,3,4] ] ]
```

MAS T2P1 Tutorials Series
Nested list basics



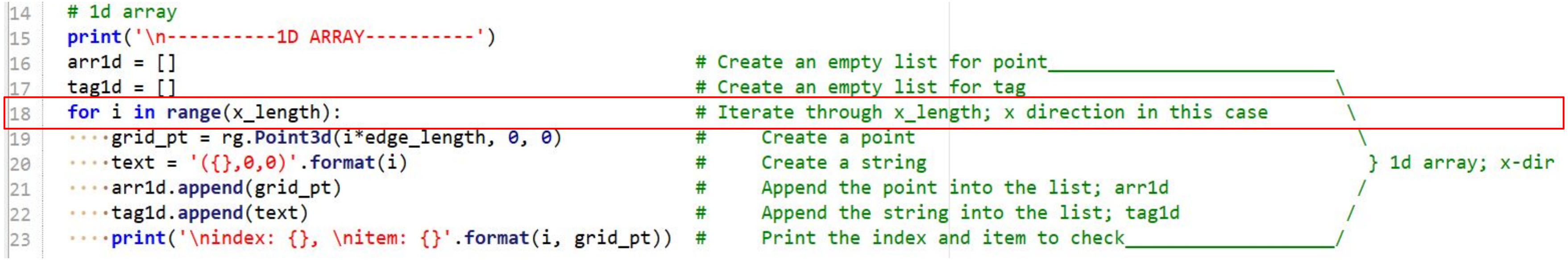


```
# 1d array
print('\n----------1D ARRAY----------')
arr1d = []                                               # Create an empty list for point__________________________
tag1d = []                                               # Create an empty list for tag                             \
for i in range(x_length):                                # Iterate through x_length; x direction in this case        \
    grid_pt = rg.Point3d(i*edge_length, 0, 0)            #      Create a point                                        \
    text = '({},0,0)'.format(i)                          #      Create a string                                        } 1d array; x-dir
    arr1d.append(grid_pt)                                #      Append the point into the list; arr1d                 /
    tag1d.append(text)                                   #      Append the string into the list; tag1d               /
    print('\nindex: {}, \nitem: {}'.format(i, grid_pt))  #      Print the index and item to check___________________/
```

arr1d [   ]

tag1d [   ]

MAS T2P1 Tutorials Series
Nested list basics



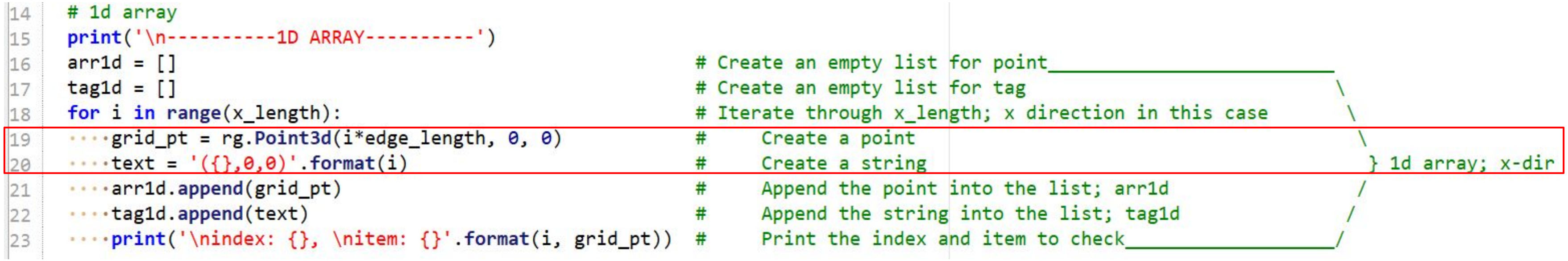
```
# 1d array
print('\n----------1D ARRAY----------')
arr1d = []                                              # Create an empty list for point__________________________
tag1d = []                                              # Create an empty list for tag                            \
for i in range(x_length):                               # Iterate through x_length; x direction in this case       \
    grid_pt = rg.Point3d(i*edge_length, 0, 0)           #     Create a point                                        \
    text = '({},0,0)'.format(i)                         #     Create a string                                        } 1d array; x-dir
    arr1d.append(grid_pt)                               #     Append the point into the list; arr1d                 /
    tag1d.append(text)                                  #     Append the string into the list; tag1d               /
    print('\nindex: {}, \nitem: {}'.format(i, grid_pt)) #     Print the index and item to check___________________/
```

i will iterate from 0 to 4 (i = 0)

arr1d [ ]

tag1d [ ]

MAS T2P1 Tutorials Series
Nested list basics



```
# 1d array
print('\n----------1D ARRAY----------')
arr1d = []                                               # Create an empty list for point__________________________
tag1d = []                                               # Create an empty list for tag                            \
for i in range(x_length):                                # Iterate through x_length; x direction in this case         \
    grid_pt = rg.Point3d(i*edge_length, 0, 0)            #     Create a point                                          \
    text = '({},0,0)'.format(i)                          #     Create a string                                           } 1d array; x-dir
    arr1d.append(grid_pt)                                #     Append the point into the list; arr1d                  /
    tag1d.append(text)                                   #     Append the string into the list; tag1d               /
    print('\nindex: {}, \nitem: {}'.format(i, grid_pt))  #     Print the index and item to check___________________/
```

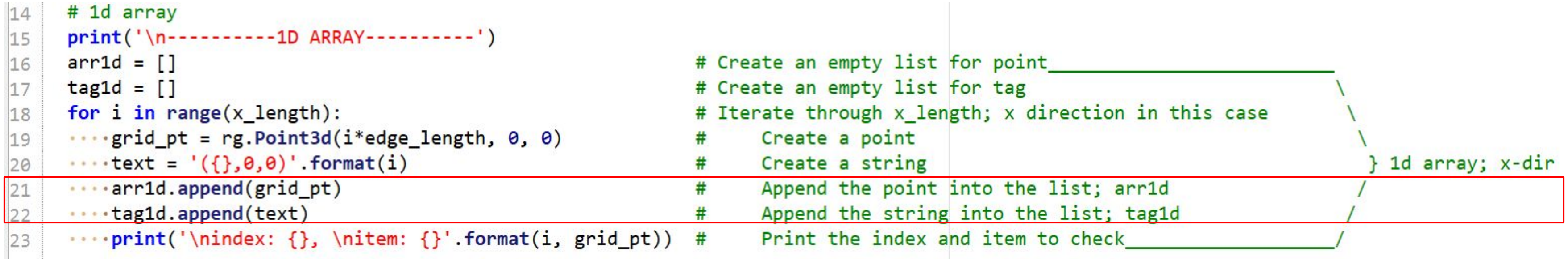
i will iterate from 0 to 4 (i = 0)

grid_pt = (0, 0, 0)

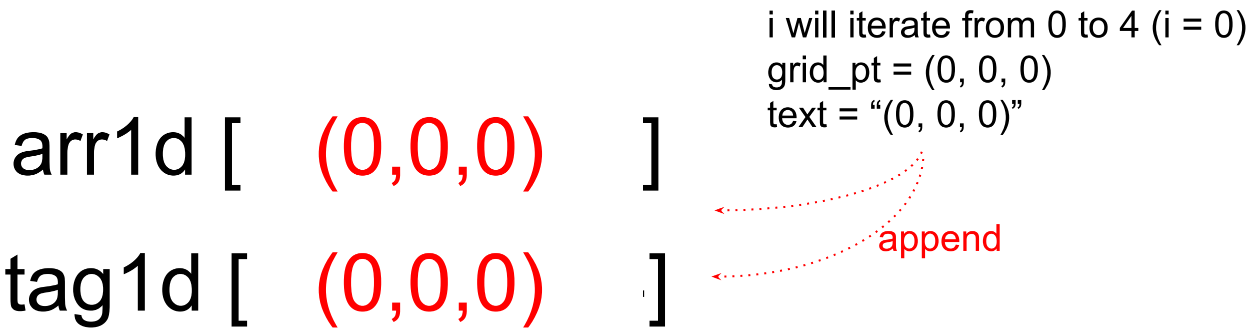
text = “(0, 0, 0)”

arr1d [ ]

tag1d [ ]

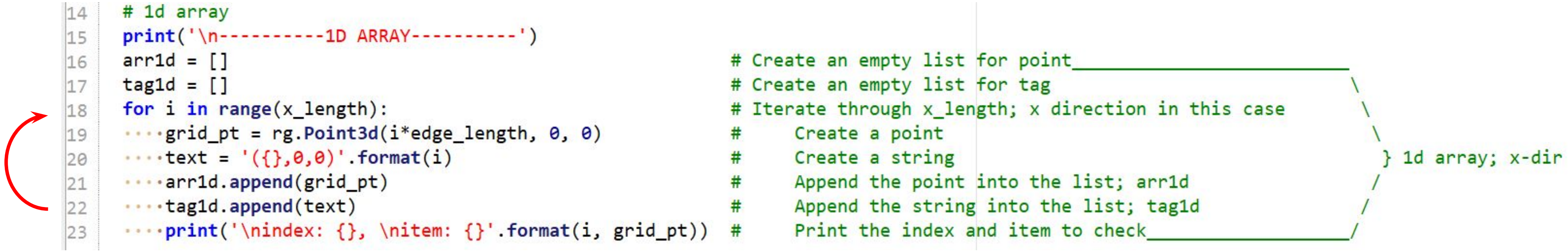


```
# 1d array
print('\n----------1D ARRAY----------')
arr1d = []                                      # Create an empty list for point__________________________
tag1d = []                                      # Create an empty list for tag                            \
for i in range(x_length):                       # Iterate through x_length; x direction in this case       \
····grid_pt = rg.Point3d(i*edge_length, 0, 0)   #     Create a point                                        \
····text = '({},0,0)'.format(i)                 #     Create a string                                        } 1d array; x-dir
····arr1d.append(grid_pt)                       #     Append the point into the list; arr1d                 /
····tag1d.append(text)                          #     Append the string into the list; tag1d               /
····print('\nindex: {}, \nitem: {}'.format(i, grid_pt))  #     Print the index and item to check___________________/
```



i will iterate from 0 to 4 (i = 0)
grid_pt = (0, 0, 0)
text = “(0, 0, 0)”

MAS T2P1 Tutorials Series
Nested list basics



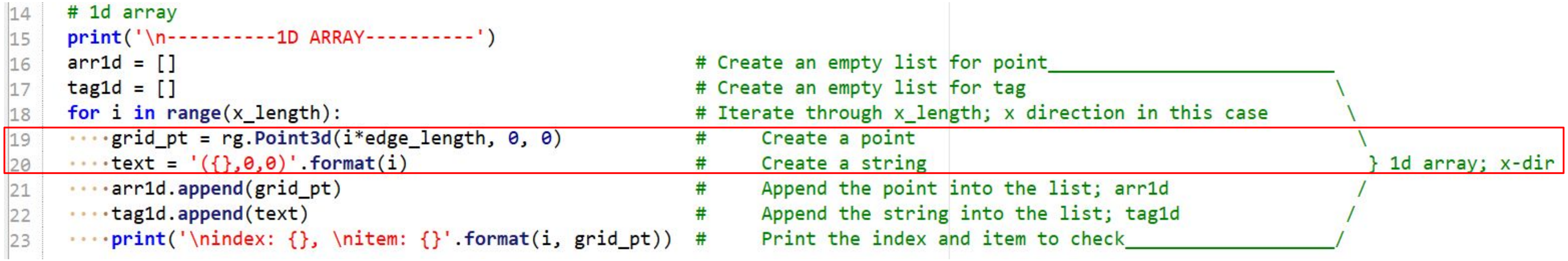
```
# 1d array
print('\n----------1D ARRAY----------')
arr1d = []                                        # Create an empty list for point__________________
tag1d = []                                        # Create an empty list for tag                      \
for i in range(x_length):                         # Iterate through x_length; x direction in this case  \
    grid_pt = rg.Point3d(i*edge_length, 0, 0)     #     Create a point                                    \
    text = '({},0,0)'.format(i)                   #     Create a string                                    } 1d array; x-dir
    arr1d.append(grid_pt)                         #     Append the point into the list; arr1d             /
    tag1d.append(text)                            #     Append the string into the list; tag1d           /
    print('\nindex: {}, \nitem: {}'.format(i, grid_pt)) #     Print the index and item to check_____________/
```

i will iterate from 0 to 4 (i = 1)
grid_pt = (0, 0, 0)
text = “(0, 0, 0)”

arr1d [(0,0,0)]

tag1d [(0,0,0)]

MAS T2P1 Tutorials Series
Nested list basics



```
# 1d array
print('\n----------1D ARRAY----------')
arr1d = []                                           # Create an empty list for point__________________________
tag1d = []                                           # Create an empty list for tag                            \
for i in range(x_length):                            # Iterate through x_length; x direction in this case       \
····grid_pt = rg.Point3d(i*edge_length, 0, 0)        #     Create a point                                        \
····text = '({},0,0)'.format(i)                      #     Create a string                                        } 1d array; x-dir
····arr1d.append(grid_pt)                            #     Append the point into the list; arr1d                 /
····tag1d.append(text)                               #     Append the string into the list; tag1d               /
····print('\nindex: {}, \nitem: {}'.format(i, grid_pt)) #     Print the index and item to check_________________/
```

i will iterate from 0 to 4 (i = 1)

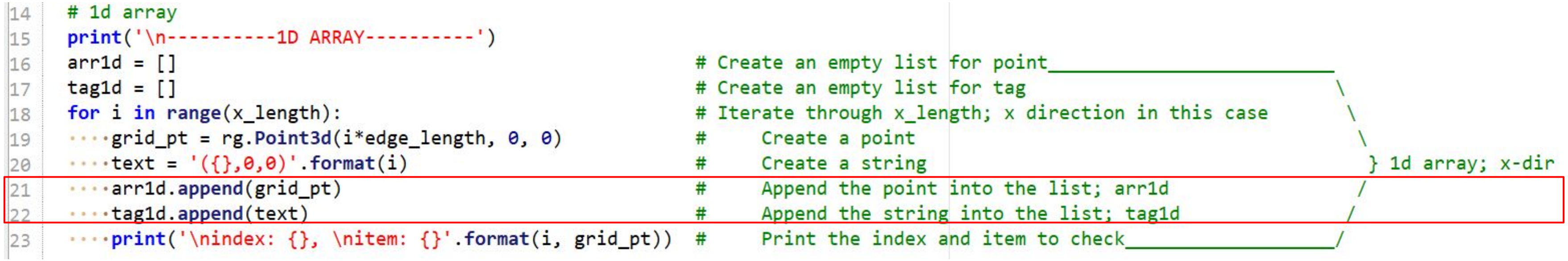
grid_pt = (1, 0, 0)

text = “(1, 0, 0)”

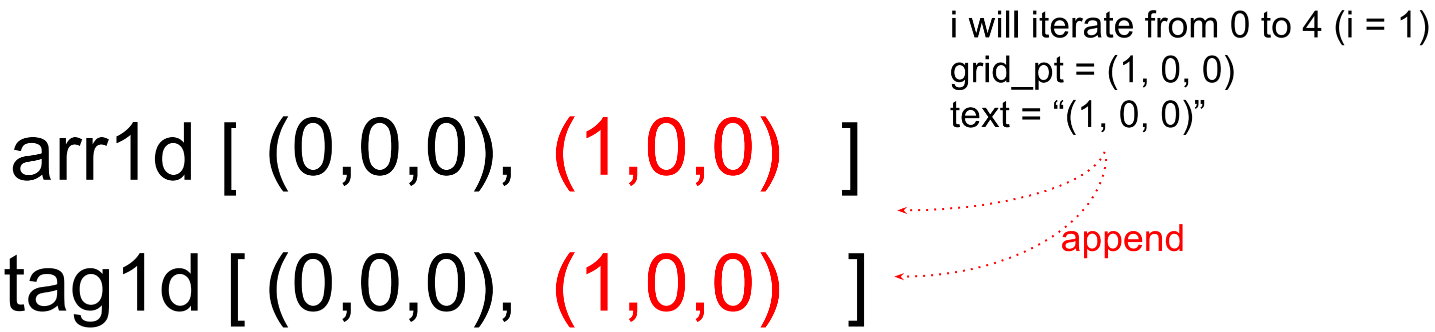
arr1d [(0,0,0)   ]

tag1d [(0,0,0)   ]

MAS T2P1 Tutorials Series
Nested list basics



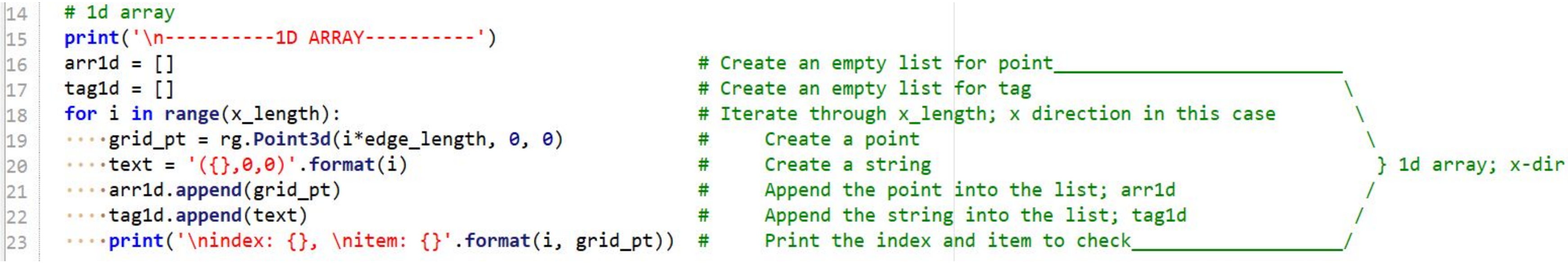
```
# 1d array
print('\n----------1D ARRAY----------')
arr1d = []                                        # Create an empty list for point__________________________
tag1d = []                                        # Create an empty list for tag                            \
for i in range(x_length):                         # Iterate through x_length; x direction in this case       \
····grid_pt = rg.Point3d(i*edge_length, 0, 0)     #     Create a point                                         \
····text = '({},0,0)'.format(i)                   #     Create a string                                         } 1d array; x-dir
····arr1d.append(grid_pt)                         #     Append the point into the list; arr1d                  /
····tag1d.append(text)                            #     Append the string into the list; tag1d                /
····print('\nindex: {}, \nitem: {}'.format(i, grid_pt))  #     Print the index and item to check____________________/
```



i will iterate from 0 to 4 (i = 1)
grid_pt = (1, 0, 0)
text = “(1, 0, 0)”

arr1d [(0,0,0), (1,0,0)]

tag1d [(0,0,0), (1,0,0)]

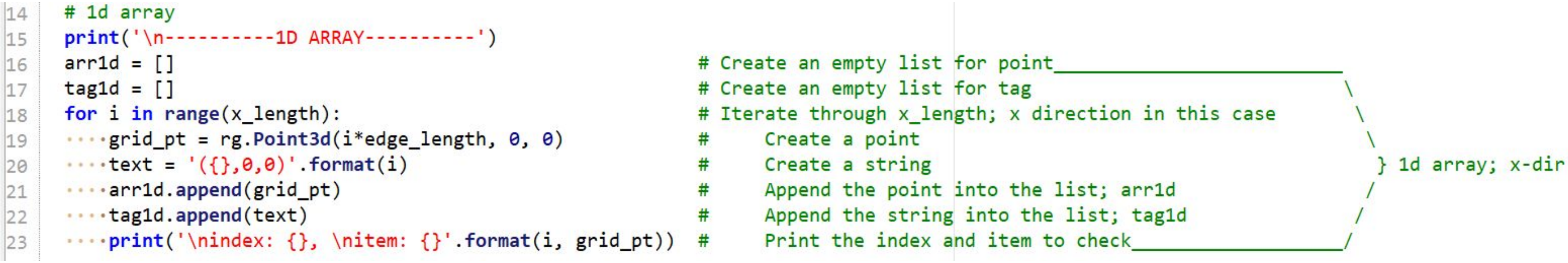


```
# 1d array
print('\n----------1D ARRAY----------')
arr1d = []                                              # Create an empty list for point____________________________
tag1d = []                                              # Create an empty list for tag                              \
for i in range(x_length):                               # Iterate through x_length; x direction in this case         \
    grid_pt = rg.Point3d(i*edge_length, 0, 0)           #     Create a point                                          \
    text = '({},0,0)'.format(i)                         #     Create a string                                          } 1d array; x-dir
    arr1d.append(grid_pt)                               #     Append the point into the list; arr1d                   /
    tag1d.append(text)                                  #     Append the string into the list; tag1d                 /
    print('\nindex: {}, \nitem: {}'.format(i, grid_pt)) #     Print the index and item to check___________________/
```

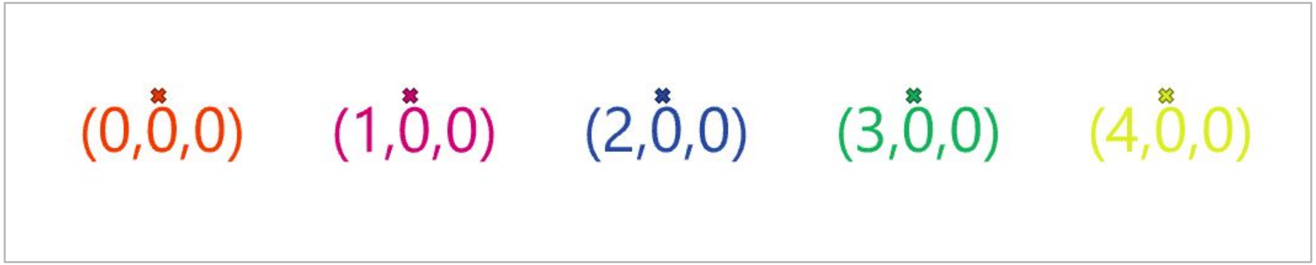
arr1d [(0,0,0), (1,0,0), (2,0,0), (3,0,0), (4,0,0)]

tag1d [(0,0,0), (1,0,0), (2,0,0), (3,0,0), (4,0,0)]

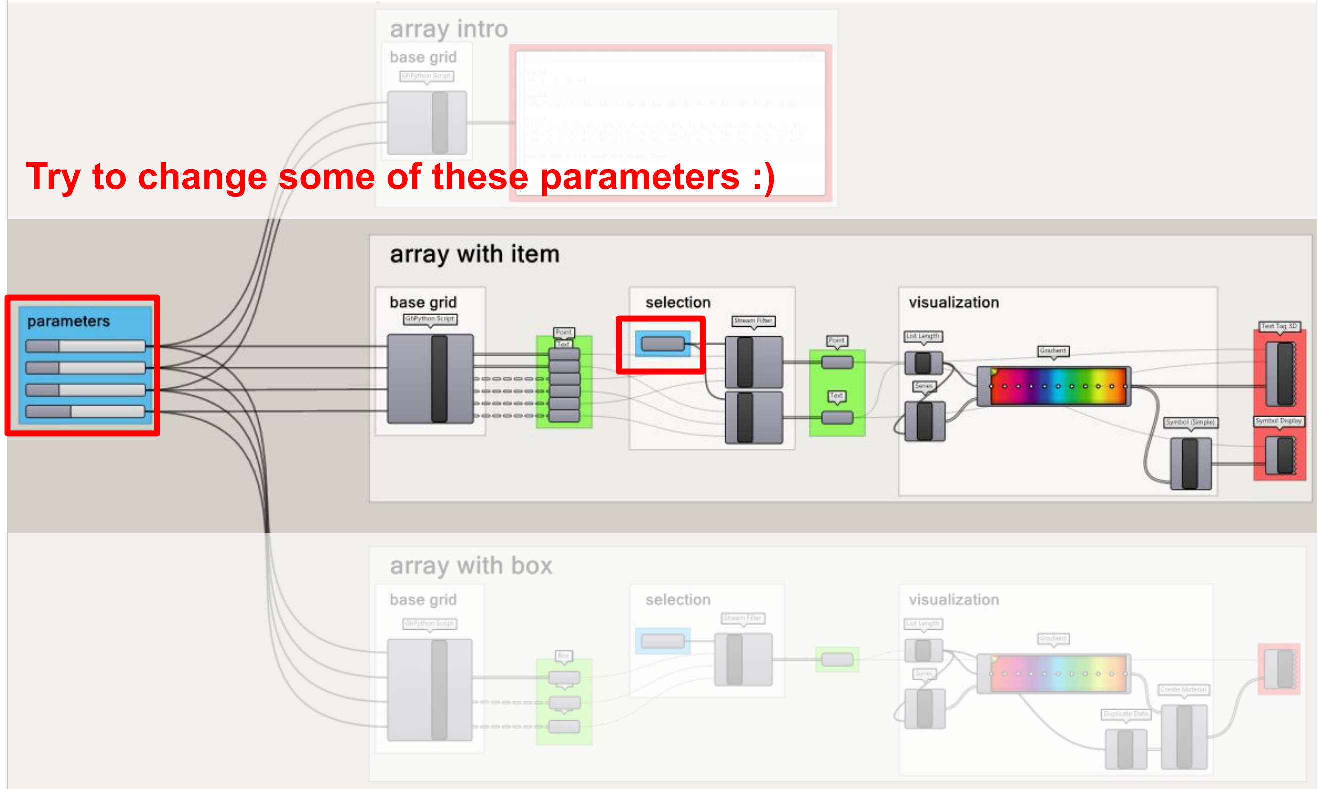
MAS T2P1 Tutorials Series
Nested list basics



```
# 1d array
print('\n----------1D ARRAY----------')
arr1d = []                                                 # Create an empty list for point__________________________
tag1d = []                                                 # Create an empty list for tag                            \
for i in range(x_length):                                  # Iterate through x_length; x direction in this case       \
    grid_pt = rg.Point3d(i*edge_length, 0, 0)              #     Create a point                                         \
    text = '({},0,0)'.format(i)                            #     Create a string                                         } 1d array; x-dir
    arr1d.append(grid_pt)                                  #     Append the point into the list; arr1d                  /
    tag1d.append(text)                                     #     Append the string into the list; tag1d                /
    print('\nindex: {}, \nitem: {}'.format(i, grid_pt))    #     Print the index and item to check____________________/
```

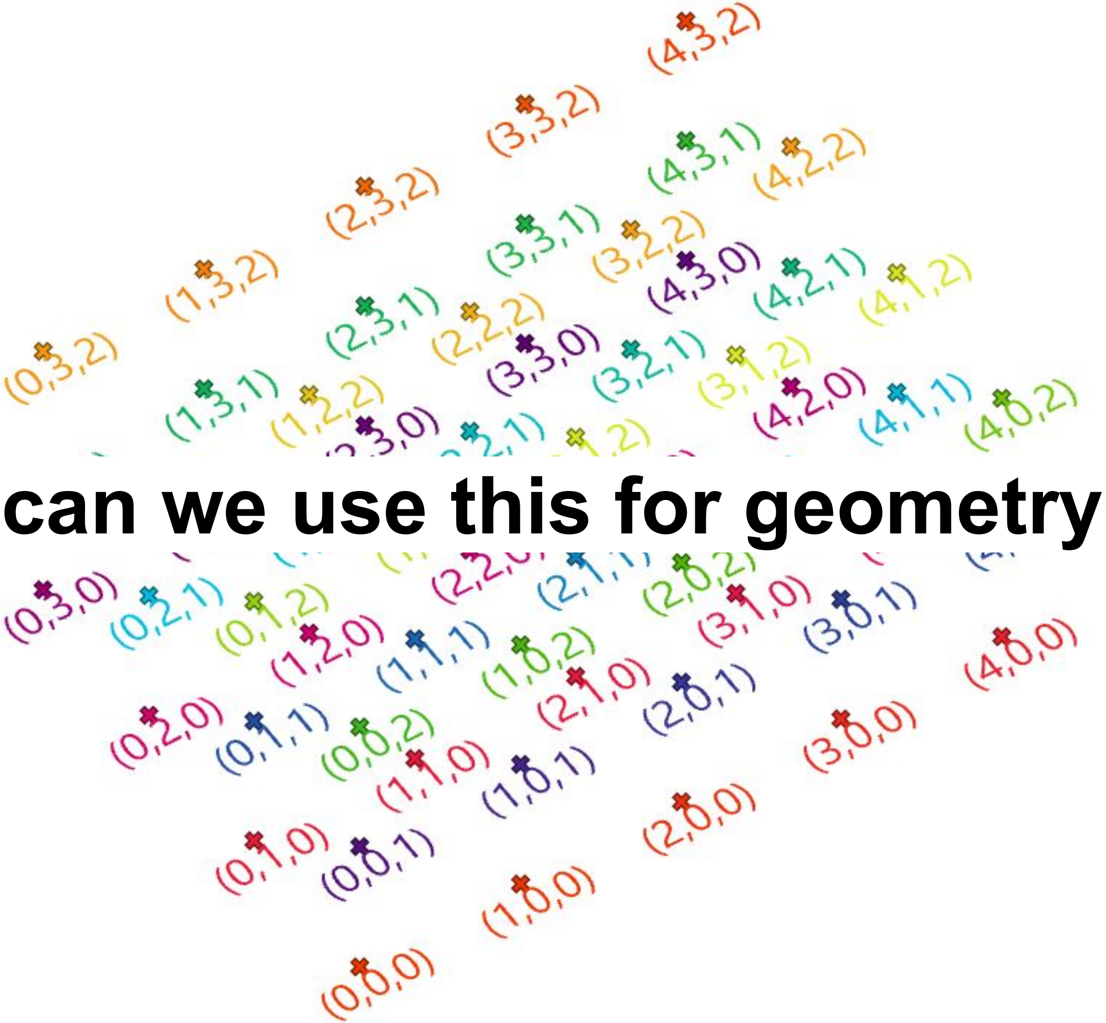


MAS T2P1 Tutorials Series
Nested list basics

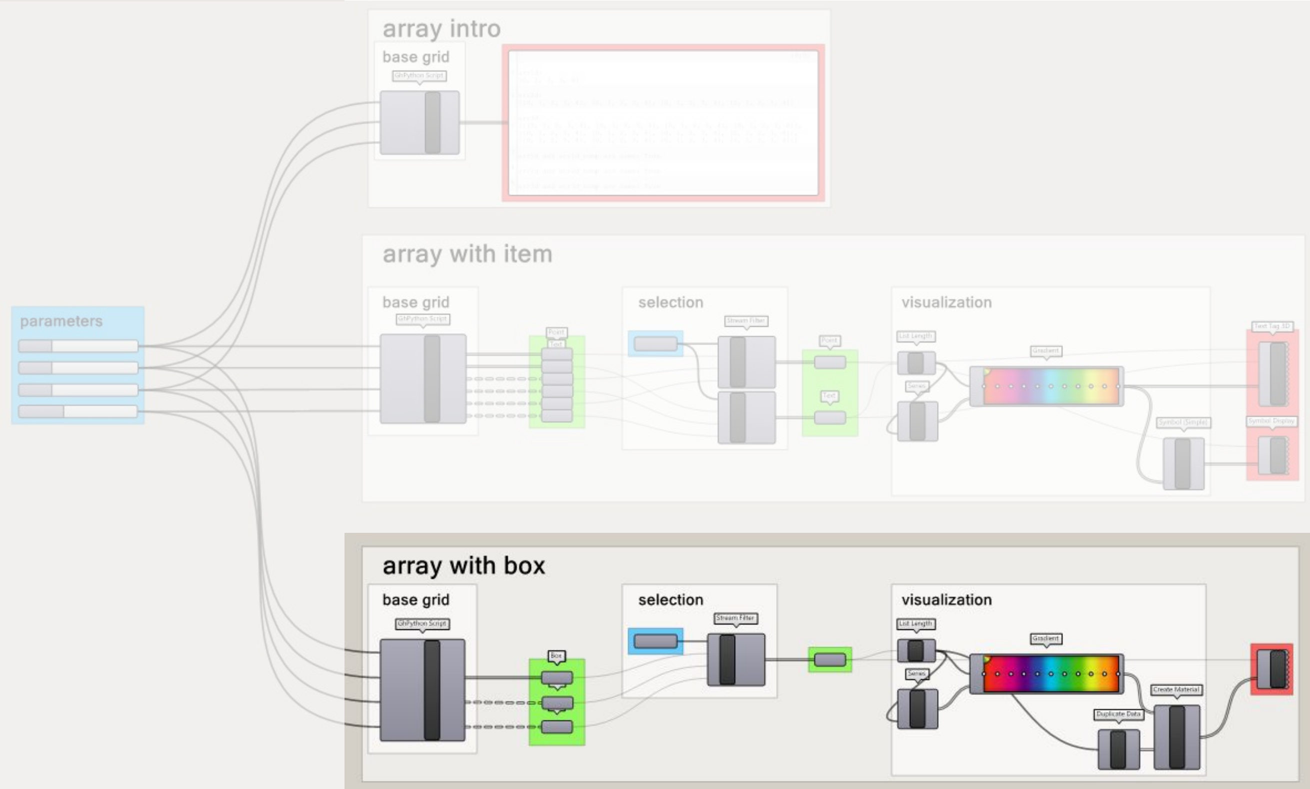


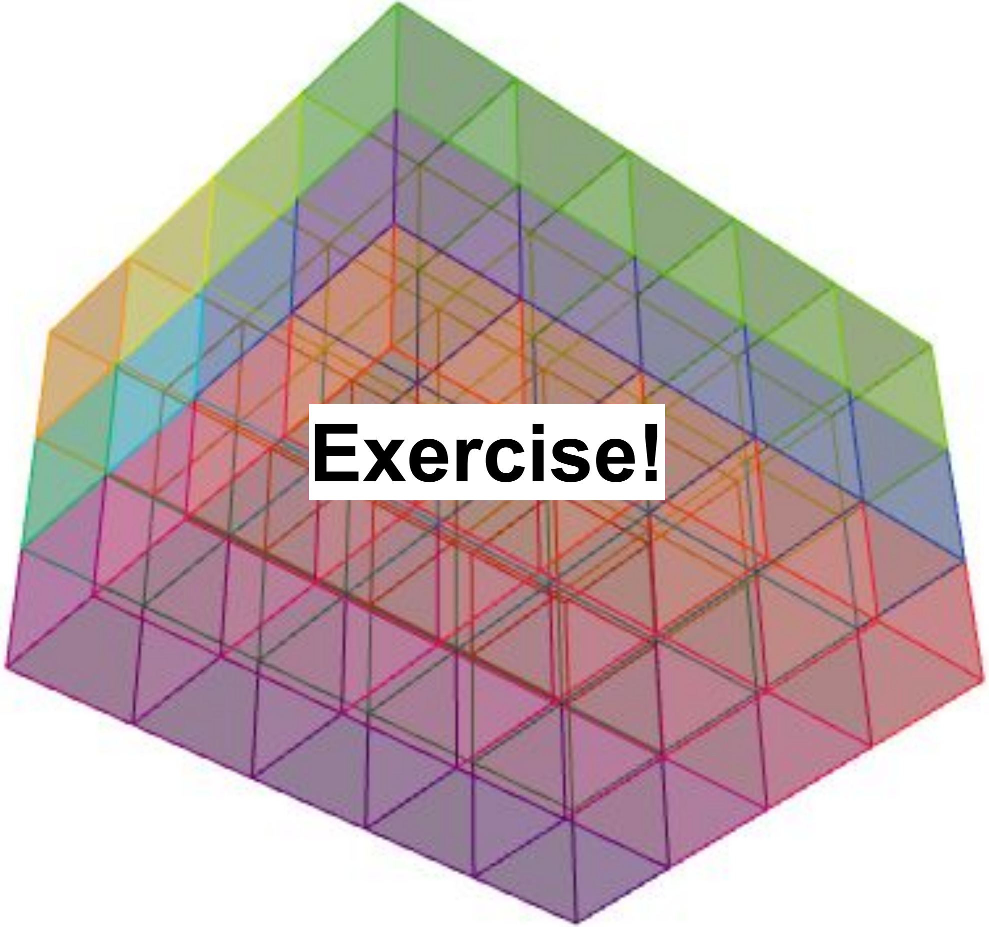
Try to change some of these parameters :)

How can we use this for geometry…?

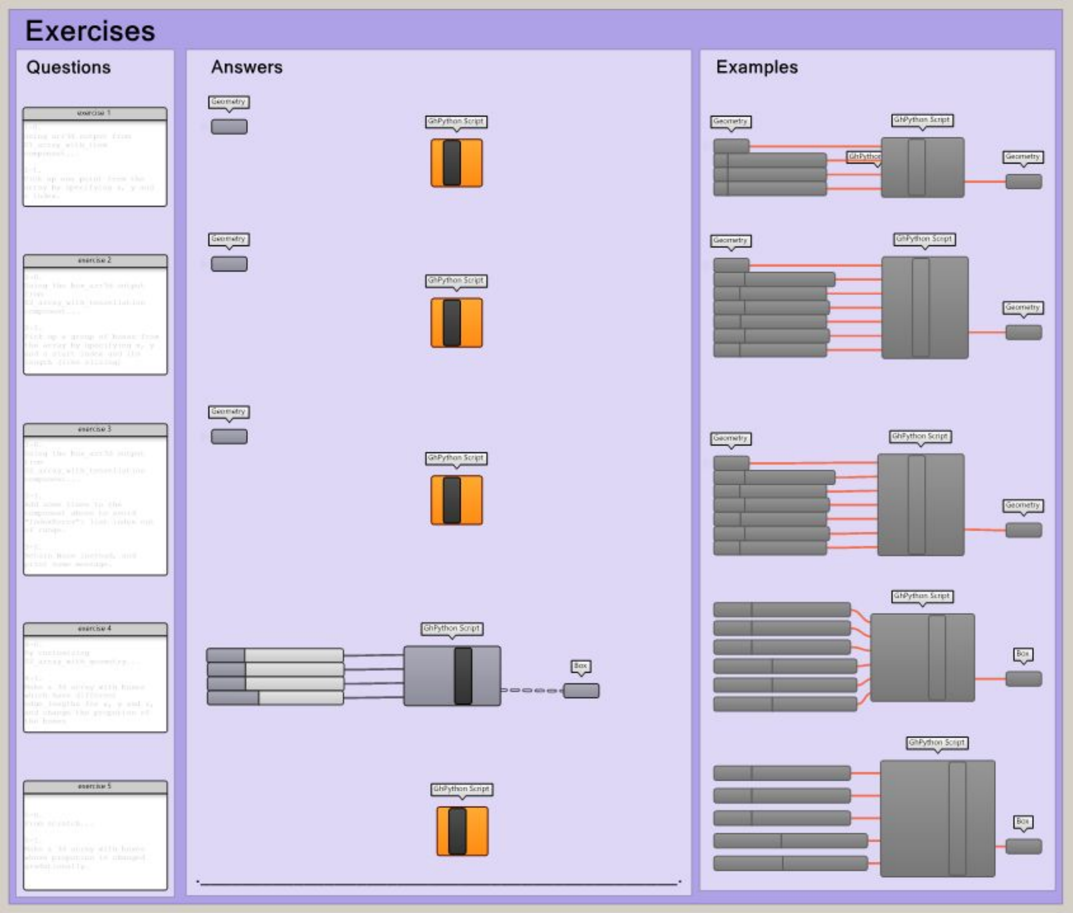


MAS T2P1 Tutorials Series
Nested list basics





Exercise!



exercise 1

1-0.
Using arr3d output from
3D array with line
component...

1-1.
Pick up one point from the
array by specifying x, y and
z index.

exercise 2

2-0.
Using the box_arr3d output
from
3D array with tessellation
component...

2-1.
Pick up a group of boxes from
the array by specifying x, y
and a start index and its
length (like slicing)

exercise 3

3-0.
Using the box_arr3d output
from
3D array with tessellation
component...

3-1.
Add some lines to the
component above to avoid
"IndexError": list index out
of range.

3-2.
Return None instead, and
print some message.

exercise 4

4-0.
By customizing
3D array with geometry...

4-1.
Make a 3d array with boxes
which have different
edge lengths for x, y and z,
and change the proportion of
the boxes

exercise 5

5-0.
From scratch...

5-1.
Make a 3d array with boxes
whose proportion is changed
probabilistically.

1. Legend
2. Tutorial
3. Exercise

TRY to PUSH!

Appendix

Topic:	**Module design**
Exercise:	**Make custom module**
Extra skill:	**Custom bake function in gh-python**

Topic: Custom module design
Exercise: Neighbor search
Extra skill: Custom export function in compas

Topic:	**Environment for development**
Exercise:	**Create own function**
Extra skill:	**Custom library / Environment control**