iscte INSTITUTO UNIVERSITÁRIO DE LISBOA

emprego digital

UPskill

<Welcome To>

python™

1.

Overall Program Content

Web development with Python	Hours
Work skills development	50
Python Programming Introduction	150
Web Programming Introduction (html/css)	100
Databases Concepts and Structures	50
Web Servers Programming	150
Web services development	150
Total	650

iscte
INSTITUTO UNIVERSITÁRIO DE LISBOA

emprego digital

Python programming Introduction Content

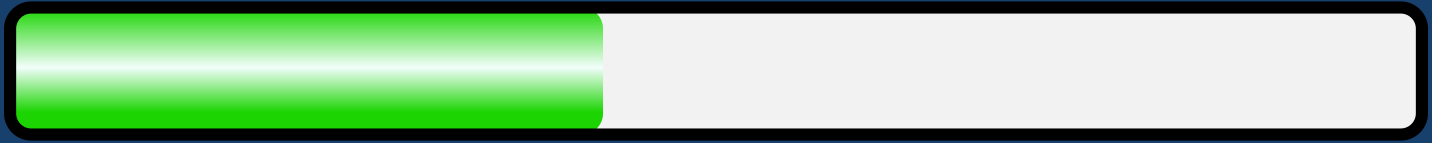
1. Course Introduction
- Why Python?
- Python Applications
- Installation Tools
- Building your code catalog
- Useful websites

Python programming Introduction Content

2. Data types/outputs/inputs
3. Operators
4. Functions and Modules

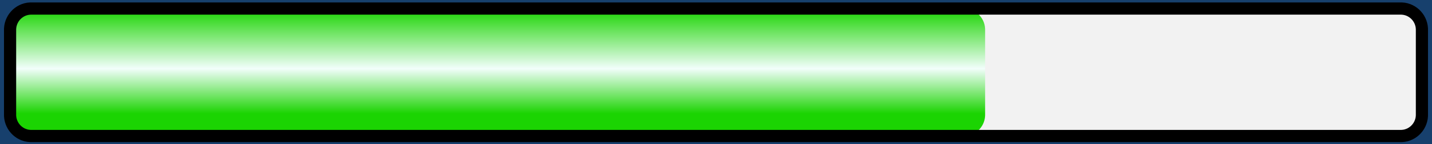
Python programming Introduction Content

5. Conditional statements and expression
6. Loops
7. Work with standard Library and Modules



Python programming Introduction Content

8. Data structure in python
9. List,
10. Tuple,
11. Dictionaries,
12. Set



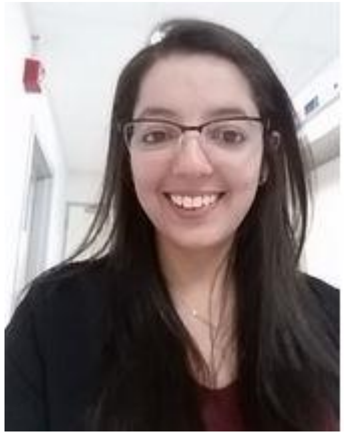
Python programming Introduction Content

Python programming Introduction Content

18. Introduction to matplotlib for data visualization
19. Data Preprocessing

100% Loaded

Our Teachers:



Joseanne Viana (Josi)

Email: jcova1@iscte-iul.pt

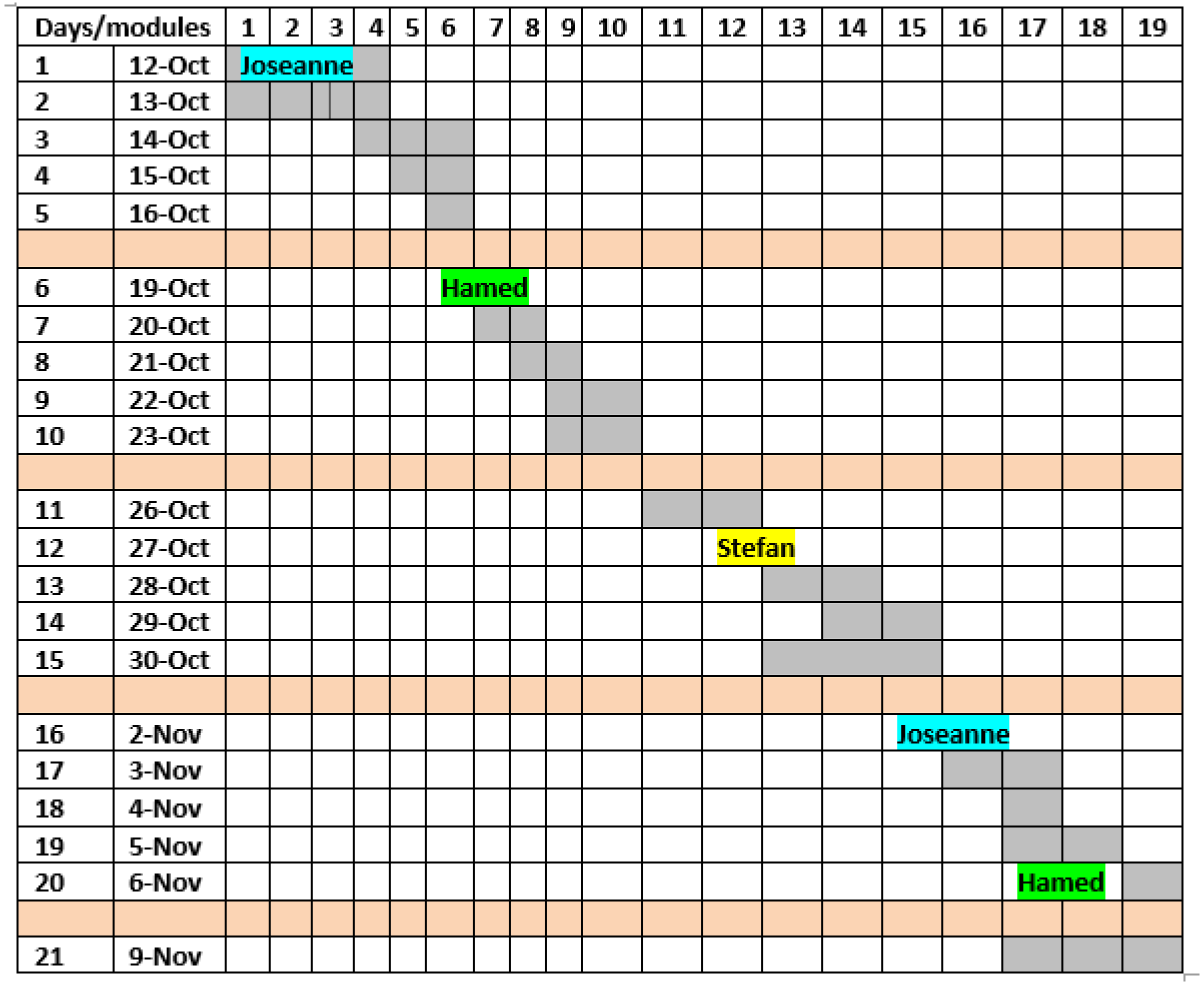
Stefan Postolache

Email:stefanpostolache@edu.ulisboa.pt

Hamed Farkhari

Email:Hamed_Farkhari@iscte-iul.pt

Schedule



Days/modules		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
1	12-Oct	Joseanne	▨	▨	▨															
2	13-Oct	▨	▨	▨	▨															
3	14-Oct				▨	▨	▨													
4	15-Oct					▨	▨													
5	16-Oct						▨													
6	19-Oct						Hamed													
7	20-Oct							▨	▨											
8	21-Oct								▨	▨										
9	22-Oct									▨	▨									
10	23-Oct									▨	▨									
11	26-Oct											▨	▨							
12	27-Oct												Stefan							
13	28-Oct													▨	▨					
14	29-Oct														▨	▨				
15	30-Oct													▨	▨	▨				
16	2-Nov															Joseanne				
17	3-Nov																▨	▨		
18	4-Nov																	▨		
19	5-Nov																	▨	▨	
20	6-Nov																	Hamed		▨
21	9-Nov																	▨	▨	▨

```
class BigFile:

    def __init__(self, datadir, ndims):
        idfile = os.path.join(datadir, "id.txt")
        self.names = [x.strip() for x in str.split(open(idfile).read()) if x.strip()]
        self.name2index = dict(zip(self.names, range(len(self.names))))
        self.ndims = ndims
        self.featurefile = os.path.join(datadir, "feature.bin")
        print "[BigFile] %d features, %d dimensions" % (len(self.names), self.ndims)
        print "                binary: %s" % self.featurefile
        print "                   txt: %s" % idfile

    def read(self, requested, isname=True):
        if isname:
            index_name_array = [(self.name2index[x], x) for x in requested if x in sel
        else:
            assert(min(requested)>=0)
            assert(max(requested)<len(self.names))
            index_name_array = [(x, self.names[x]) for x in requested]
        index_name_array.sort()

        vecs = seq_read(self.featurefile, self.ndims, [x[0] for x in index_name_arr
        return [x[1] for x in index_name_array], vecs

    def shape(self):
        return [len(self.names), self.ndims]
```

<Let’s get started >

Contents

1. List

List

Define a List

Define a List simply by using
[…, …, …]   ,   Or by List() function

```
a = [5 , 7, 12]     # define a list , [ ]
print(a[0])         # 5
print(a[1])         # 7
print(a[2])         # 12

print(type(a))      # <class 'list'>
print(len(a))       # 3
```

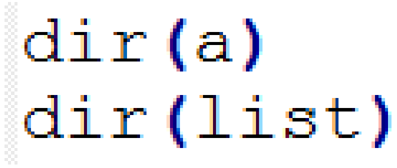
You can put different types in a list

```
 b = [1.618, 'Python Course', 0,

      {'joe': 21}, [], (3, 6, 9) ]
```

Methods Attributes of a List

a = [5 , 7, 12]

Simply use dir()

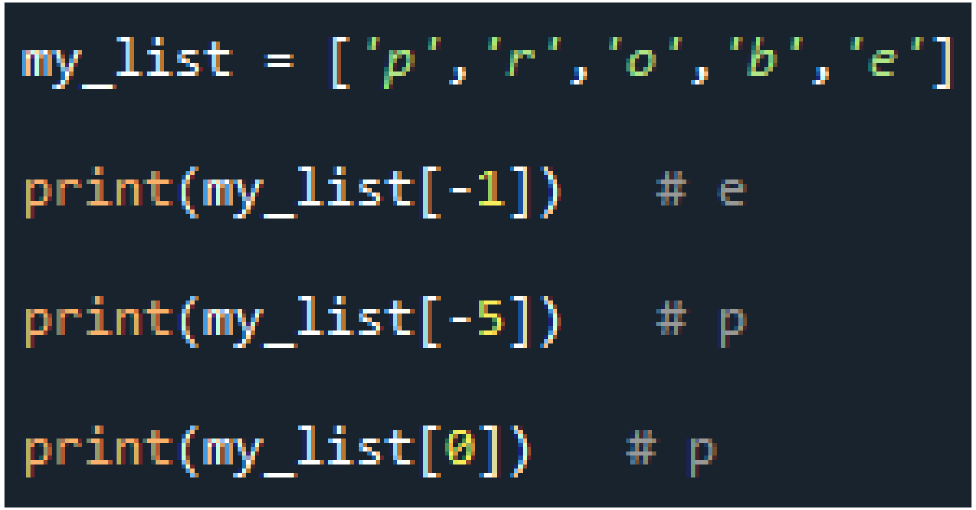


```
dir(a)
dir(list)
```

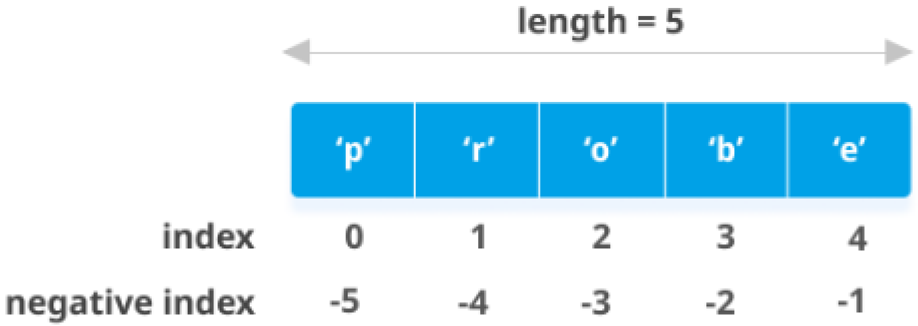
Methods in List:

index , count , insert , remove , pop , reverse , sort , extend , append , clear, copy , ...

What is the index?



```
my_list = ['p','r','o','b','e']

print(my_list[-1])    # e

print(my_list[-5])    # p

print(my_list[0])    # p
```



Index

Find the index of a member in a list

a = [5 , 7, 12]

```
print(a.index(7))    # 1
```

Use the index of a member to access it

```
print(a[1])          # 7
```

List is mutable, you can change it

```
a[1] = 8             # [5, 8, 12]
```

Index in strings

Compare a List with a string

```
s = 'sara'                # string

print(s[1])               # a

s[1]='d'                  # Error
```

string is immutable, you can Not change the string characters by index

Ordered

list is ordered

```
a = [1, 2]
b = [2, 1]
print(a == b)      # False
```

What is the meaning of Ordered???

What other data types you know which is ordered same as a List? Tuple, string

What data types you know which is Not ordered? Set

Example

show the items in a list

```python
friends = ['Hamed', 'Josi', 'Stefan']
for f in friends:
    print(f)
```

You can use range and index

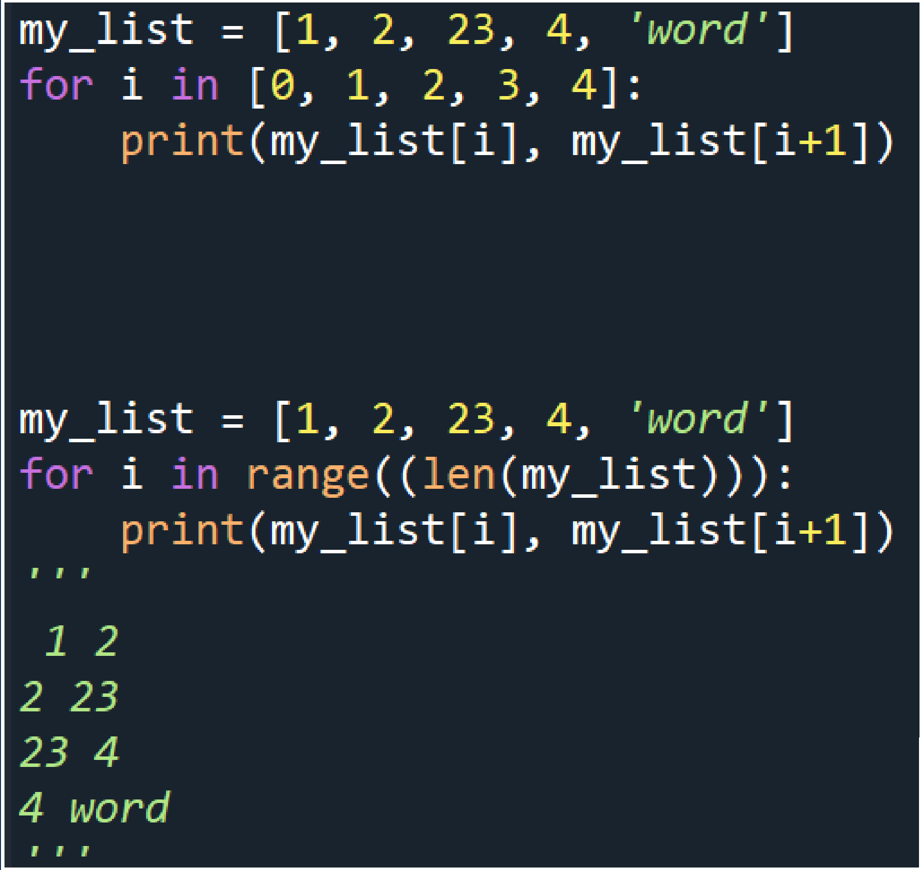
```python
for i in range(3):   # range(len(friends))
    print(friends[i])
```

```
class BigFile:

    def __init__(self, datadir, ndims):
        idfile = os.path.join(datadir, "id.txt")
        self.names = [x.strip() for x in str.split(open(idfile).read()) if x.strip()]
        self.name2index = dict(zip(self.names, range(len(self.names))))
        self.ndims = ndims
        self.featurefile = os.path.join(datadir, "feature.bin")
        print "[BigFile] %d features, %d dimensions" % (len(self.names), self.ndims)
        print "                binary: %s" % self.featurefile
        print "                txt: %s" % idfile

    def read(self, requested, isname=True):
        if isname:
            index_name_array = [(self.name2index[x], x) for x in requested if x in sel
        else:
            assert(min(requested)>=0)
            assert(max(requested)<len(self.names))
            index_name_array = [(x, self.names[x]) for x in requested]
        index_name_array.sort()

        vecs = seq_read(self.featurefile, self.ndims, [x[0] for x in index_name_arr
        return [x[1] for x in index_name_array], vecs

    def shape(self):
        return [len(self.names), self.ndims]
```

<Exercise 1>

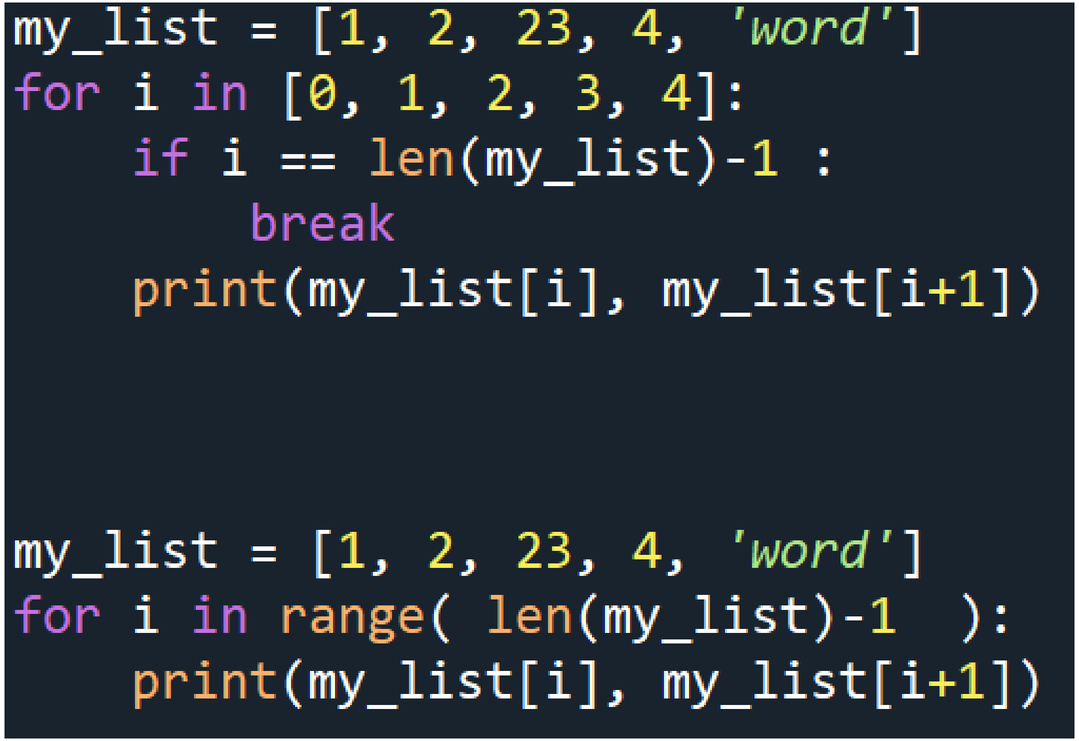
Exercise

Change the below programs to avoid getting Error
IndexError: list index out of range



```
my_list = [1, 2, 23, 4, 'word']
for i in [0, 1, 2, 3, 4]:
    print(my_list[i], my_list[i+1])




my_list = [1, 2, 23, 4, 'word']
for i in range((len(my_list))):
    print(my_list[i], my_list[i+1])
'''
 1 2
2 23
23 4
4 word
'''
```

Exercise Solutions



```
my_list = [1, 2, 23, 4, 'word']
for i in [0, 1, 2, 3, 4]:
    if i == len(my_list)-1 :
        break
    print(my_list[i], my_list[i+1])



my_list = [1, 2, 23, 4, 'word']
for i in range( len(my_list)-1  ):
    print(my_list[i], my_list[i+1])
```

Data types in a list

You can use different data types in a list

[int, int, bool, str, float, list, ...]

```
L = [3, 6, True, 'ali', 2.7, [5,8]]
```

Slicing

Slicing in a list

list_name[Start : Stop : Step]

Start is included,

Stop is **Not** included

Default Step is +1

Slicing Example

a = [7, 5, 30, 2, 6, 25]

list_name[Start : Stop] , step is +1

```
print(a[1:4])        # [5 , 30 , 2]
```

list_name[: Stop]

':' means start from the first, step is +1

```
print(a[:3])         # [7 , 5 , 30]
```

list_name[Start :]

':' means until the end, step is +1

```
print(a[3:])         # [2 , 6 , 25]
```

Slicing Example

a = [7, 5, 30, 2, 6, 25]

Start > Stop, No Step means 'Step = +1'

```
print(a[3:0])           # []
```

Start > Stop , Step = -1,
remember that the Stop is Not included

```
print(a[3:0:-1])    # [2, 30, 5]
```

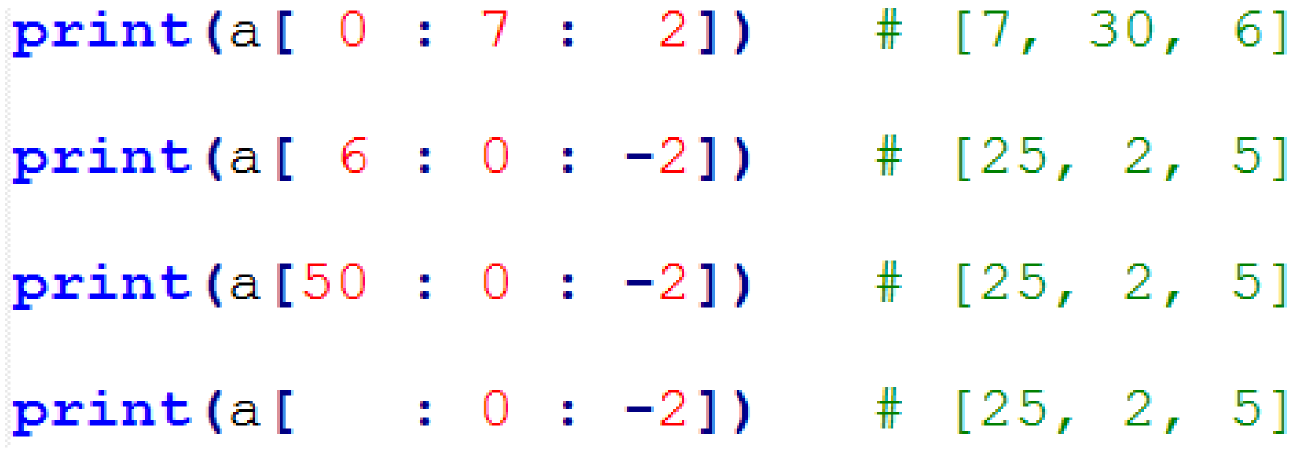
list_name[: : -1]
reverse a list

```
print(a[::-1])          # [25, 6, 2, 30, 5, 7]
```

Slicing Example

a = [7, 5, 30, 2, 6, 25]

Slicing with step



```
print(a[ 0 : 7 :  2])    # [7, 30, 6]

print(a[ 6 : 0 : -2])    # [25, 2, 5]

print(a[50 : 0 : -2])    # [25, 2, 5]

print(a[    : 0 : -2])    # [25, 2, 5]
```

Slicing
&
Change values

```
a = [7, 5, 30, 2, 6, 25]

a[3:5]=[14, 15]

print(a)   # [7, 5, 30, 14, 15, 25]
```

+ *

Repeat and Concatenate lists with * and +

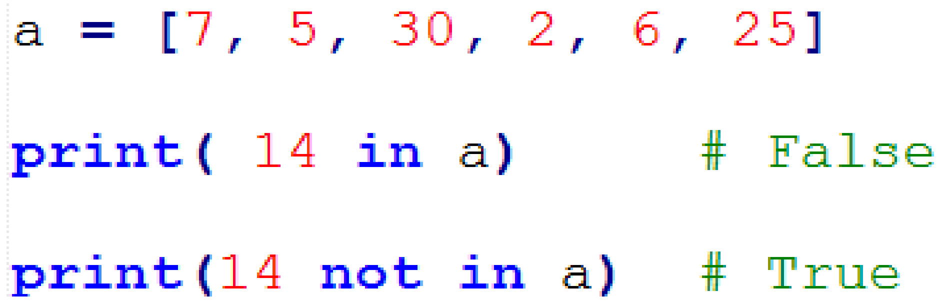
```
a = [4, 7]
b = a*2                    # '*' repeat list 2 times
print(b)                   # [4, 7, 4, 7]


a = [1, 2]
b = ['a', 'b', 'c']
c = a + b                  # '+' concatinate 2 lists
print(c)                   # [1, 2, 'a', 'b', 'c']
```

iscte INSTITUTO UNIVERSITÁRIO DE LISBOA

emprego digital

in
not in

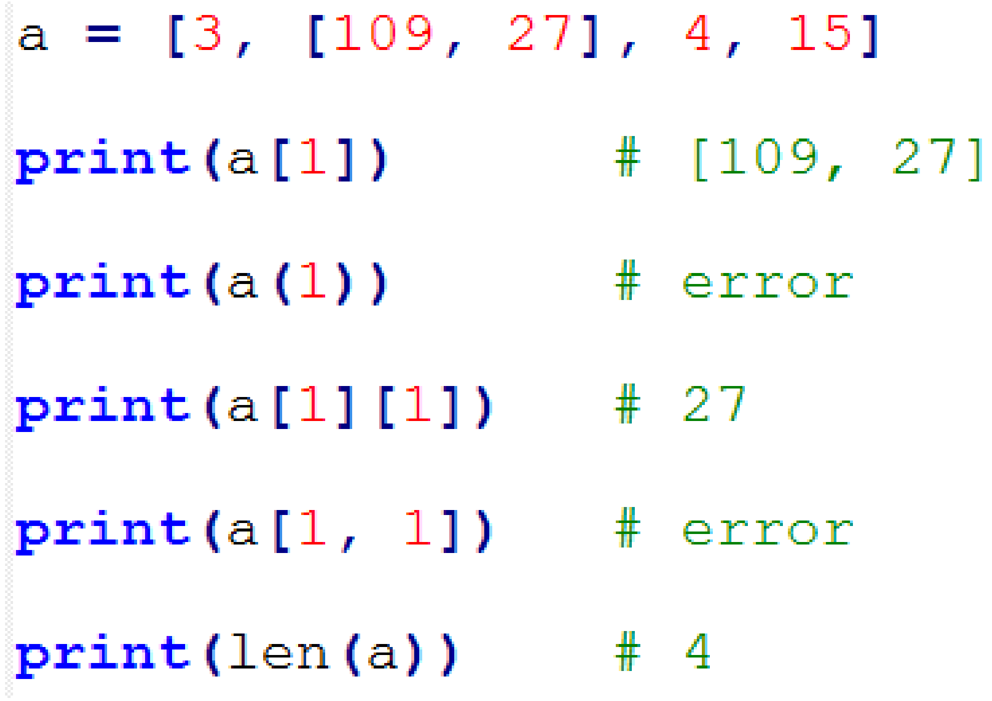
Check membership with 'in' and 'not in'



```
a = [7, 5, 30, 2, 6, 25]

print( 14 in a)        # False

print(14 not in a)   # True
```

Lists in a List

How to access the values of nested lists



```
a = [3, [109, 27], 4, 15]

print(a[1])          # [109, 27]

print(a(1))          # error

print(a[1][1])       # 27

print(a[1, 1])       # error

print(len(a))        # 4
```

<Exercise 2>

Exercise

Find the maximum value in this list?

```
a = [7 , 5 , 30 , 2 , 6 , 25]
```

Find the index of maximum value also?

Calculate sum of all members with 'for' loop?

Exercise

Find the maximum value in this list?

```
a = [7 , 5 , 30 , 2 , 6 , 25]

m = a[0]
for i in a:
    if i > m:
        m = i
print(m)            # 30
```

Max
Min
Sum

```python
a = [7 , 5 , 30 , 2 , 6 , 25]

print(max(a))        # 30

print(min(a))        # 2

print(sum(a))        # 75
```

calculate sum with ‘for’ loop

```python
s = 0
for i in a:
    s += i           # S = S + i
print(s)             # 75
```

Count
Insert

count(), number of occurrences of a value

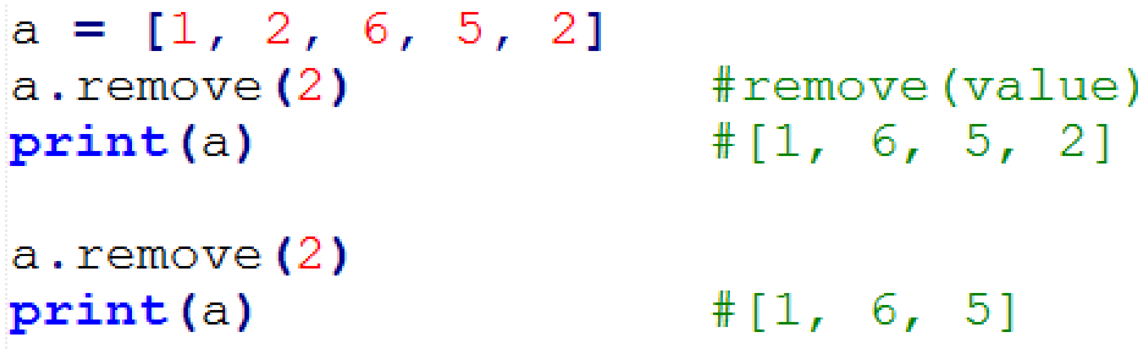
```
a = [1, 3, 6, 5, 3]
print(a.count(3))        # 2
```

insert(), insert object before index

```
a = [1, 2, 6, 5, 2]
a.insert(2,13)              # insert(index, obj)
print(a)                    # [1, 2, 13, 6, 5, 2]
```

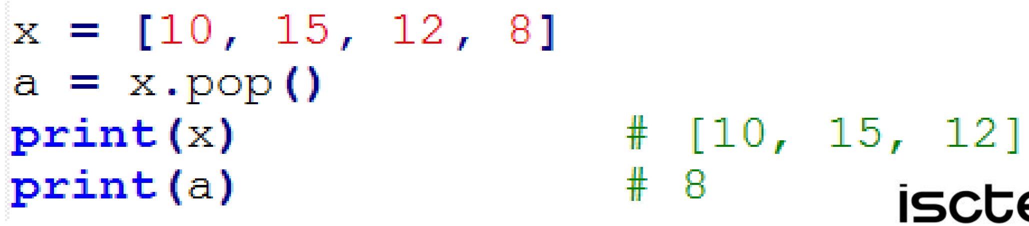
Remove
pop

remove() first occurrence of value



```
a = [1, 2, 6, 5, 2]
a.remove(2)                 #remove(value)
print(a)                    #[1, 6, 5, 2]

a.remove(2)
print(a)                    #[1, 6, 5]
```

pop() , Remove and return item at index (default last)



```
x = [10, 15, 12, 8]
a = x.pop()
print(x)                  # [10, 15, 12]
print(a)                  # 8
```

pop
Del

remove the obj by index from the list

```
y = ['a', 'b', 'c']
p = y.pop(1)    # pop(index)
print(p)        # b
print(y)        # ['a','c']
```

del does Not return the deleted value

```
a = [5 , 9 , 3]

del a[1]

b = del a[1]   # Error

print(a)       # [5, 3]
```

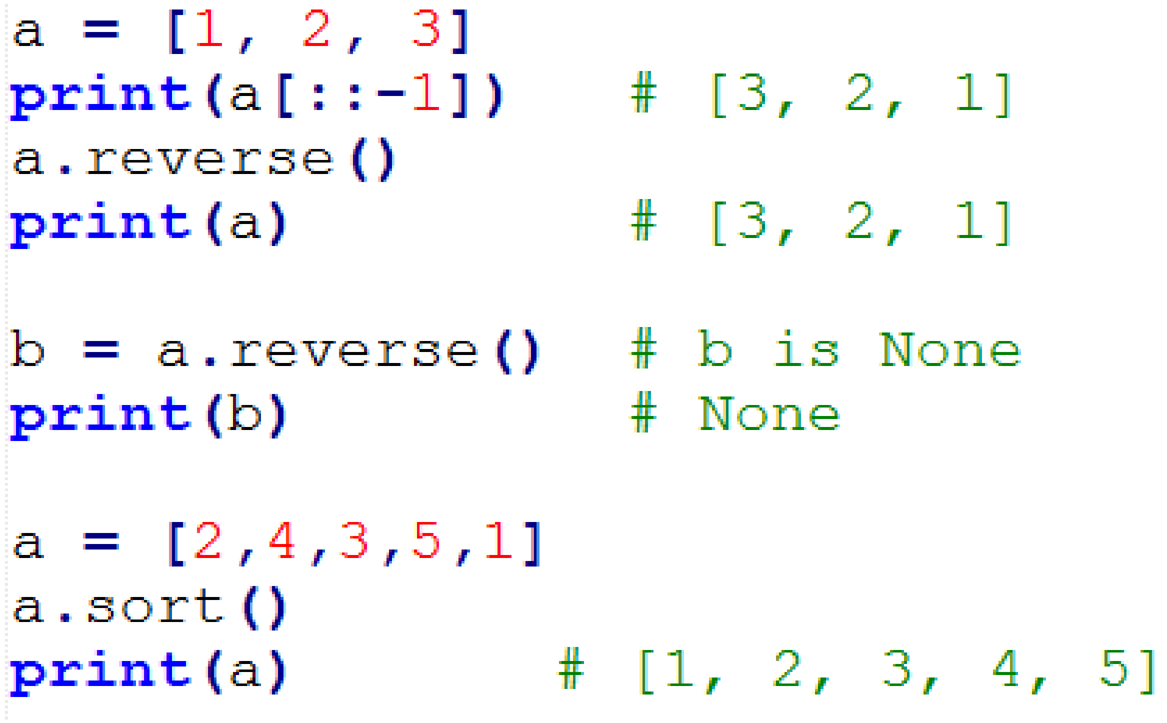
Del
Slicing

del multi objs by slicing

```
a = [0, 1, 2, 3, 4, 5, 6]

del a[2:4]

print(a)        #[0, 1, 4, 5, 6]
```

Reverse
Sort

Reverse and Sorting



```
a = [1, 2, 3]
print(a[::-1])     # [3, 2, 1]
a.reverse()
print(a)           # [3, 2, 1]

b = a.reverse()    # b is None
print(b)           # None

a = [2,4,3,5,1]
a.sort()
print(a)         # [1, 2, 3, 4, 5]
```

Extend

extend()

```
x = [1, 2, 3]
x.extend(5)          # Error
x.extend([5])
print(x)             # [1, 2, 3, 5]
```

join the list X to the end of list Y

```
x = [1, 2, 3]
y = [4, 5]
x.extend(y)
print(x)            # [1, 2, 3, 4, 5]
print(len(x))       # 5
print(len(y))       # 2
```

Append

append()

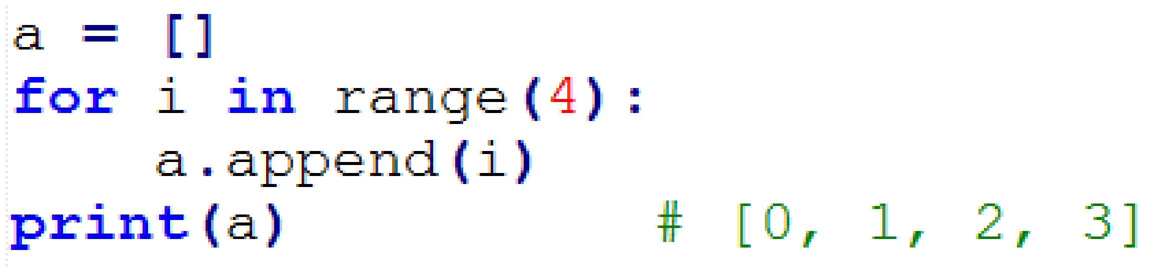
```
a = [1,2,3]
a.append(4)
print(a)          # [1, 2, 3, 4]
```

add list Y as One member to the list X

```
x = [1, 2, 3]
y = [4, 5]
x.append(y)
print(x)          # [1, 2, 3, [4, 5]]
print(len(x))     # 4
print(len(y))     # 2
```

Append

You can use loops and append() to initialize a list



```
a = []
for i in range(4):
    a.append(i)
print(a)                # [0, 1, 2, 3]
```

Clear
Copy

clear()

```
a = [1,2,3]
a.clear()
print(a)              #  []
print(len(a))         #  0
```

copy()

```
a = [1,2,3]
b = a.copy()
print(b)          # [1, 2, 3]
```

Copy

why we should use copy()

```
a = [1,2,3]
b = a.copy()  # a, b are independent
c = a         # a, c are dependent to each other
d = a[:]      # a, d are independent
```

when we change c or a, both of them be changed, but b is independent

```
a[1] = 22
c[0] = 11
d[2] = 33
```

```
print(a)    # [11, 22,  3]
print(b)    # [1 , 2 ,  3]
print(c)    # [11, 22,  3]
print(d)    # [1 , 2 , 33]
```

Example

X and Y are independent, because they are int, Not lists

```
x = 2
y = x
y += 1
print(x)    # 2
print(y)    # 3
```

X and Y both point to the same location in the memory, here X and Y are lists

```
x = []
y = x
y.append(5)
print(x)     # [5]
print(y)     # [5]
```

[<M operation> for M in a]

[<M operation> for M in a]

do the <M operation> for all members in a

M is the representor of members in a

```
a = [i for i in range(4)]
print(a)                          # [0, 1, 2, 3]
```

```
a = [i*2 for i in range(4)]
print(a)                          # [0, 2, 4, 6]
```

```
a = [i*i for i in range(3,6)]
print(a)                            # [9, 16, 25]
```

range create 3, 4, 5

[<M operation> for M in a]

```
a = [1 , -2 , 5 , -56 , 8]
b = [abs(i) for i in a]
print(b)                        # [1, 2, 5, 56, 8]
```

```
import math
a = [round(math.pi,i) for i in range(1,5)]
print(a)           # [3.1, 3.14, 3.142, 3.1416]
```

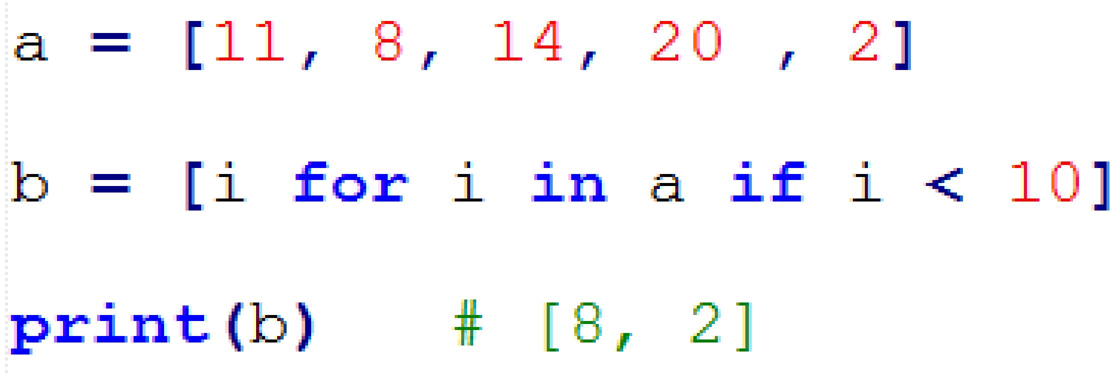
remove $ from all members in list

```
a = ['$ali', 'sara$']
b = [i.strip('$') for i in a]
print(b)           # ['ali', 'sara']
```

Operation on Filtered Members

[<M operation> for M in a if <M filter>]

do the <M operation>
for only filtered members in a



```
a = [11, 8, 14, 20 , 2]

b = [i for i in a if i < 10]

print(b)     # [8, 2]
```

iscte INSTITUTO UNIVERSITÁRIO DE LISBOA emprego digital

<Exercise 3>

Exercise

Could you guess what is the output?

```
a = [1, 2]
b = [1 ,4, 5]
c = []
for i in a:
    for j in b:
        if i != j:
            c.append((i,j))
print(c)
```

Exercise Answer

$i = 1$, $j = 1$, $c = [\ ]$
$i = 1$, $j = 4$, $c = [(1, 4)]$
$i = 1$, $j = 5$, $c = [(1, 4), (1, 5)]$
$i = 2$, $j = 1$, $c = [(1, 4), (1, 5)]$

...

$i = 2$, $j = 5$,

$c = [(1, 4), (1, 5), (2, 1), (2, 4), (2, 5)]$

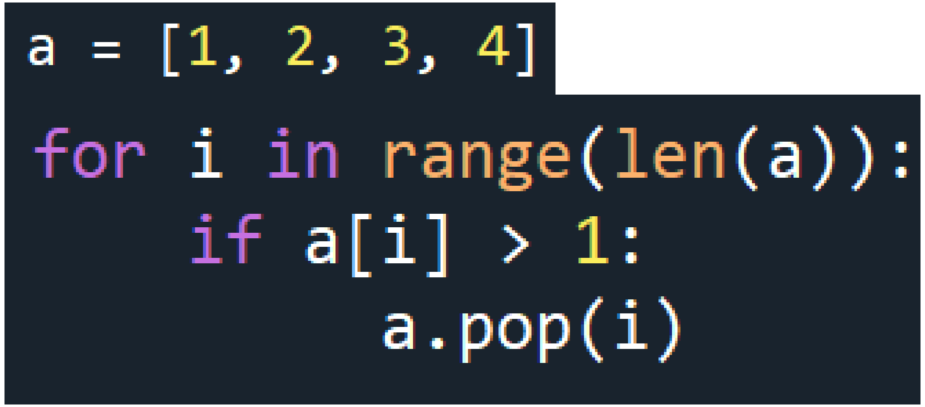
NaN values

Remove NaN from a list with for loop

```python
a = [2.6, float('NaN') , 4.8 , 6.9, float('NaN')]
b = []

import math

for i in a:
    if not math.isnan(i):
        b.append(i)

print(b)                 # [2.6, 4.8, 6.9]
```

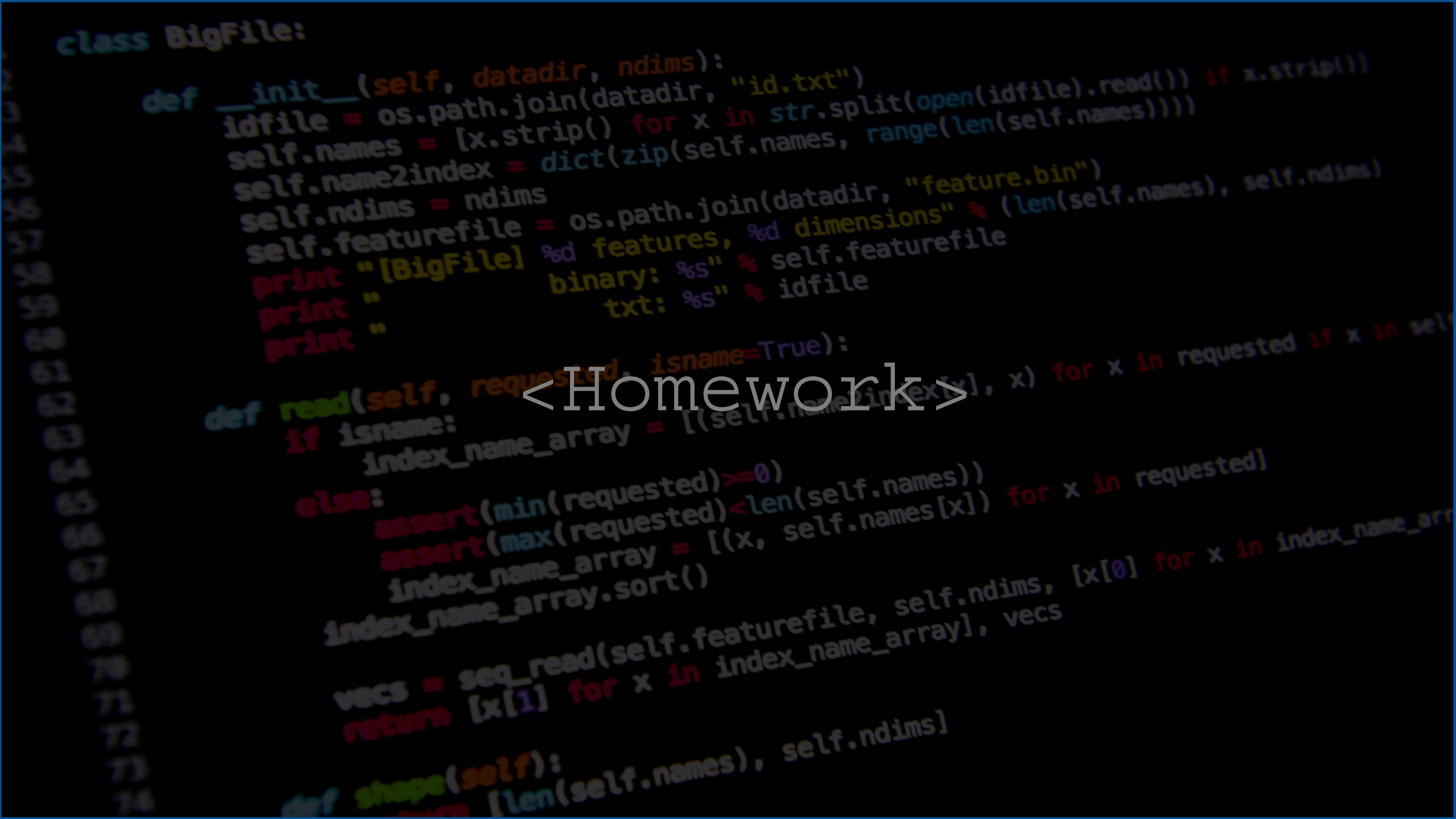
Hint

If you need to change length of a list, dict or set variable in a loop,
you need to be care about index of your variable!
Maybe you need to make a copy before your loop and change copied variable, not the main one.



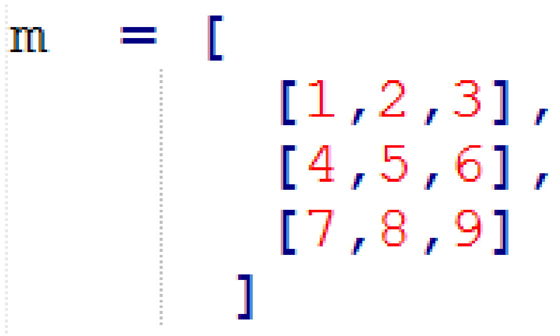
```python
a = [1, 2, 3, 4]
for i in range(len(a)):
    if a[i] > 1:
        a.pop(i)
```

Error: list index out of range



<Homework>

Matrix Exercise



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

1- print first row
2- print first column in single line
3- print main diameter 1, 5, 9
4- print another diameter 3 5 7
5- Calculate Sum of rows
6- Calculate Sum of columns

“

- *Make it work*
- *Make it Right*
- *Make it Fast*

O futuro profissional começa aqui

iscte
INSTITUTO UNIVERSITÁRIO DE LISBOA

emprego digital

UPskill