

# Python tutorial

---

## Python tutorial videos

Click on the below links to refresh your Python skills.

1. [Array Math](#)
2. [NumPy Array](#)
3. [Functions](#)
4. [Data processing using NumPy and Pandas](#)
5. [Introduction to object-oriented programming](#)

Note that a detailed tutorial with 17 hours of youtube video is given here:

<https://github.com/rohitash-chandra/python-bootcamp>

---

# Numpy - Introduction

We assume that you already know Python, either from doing the course ZZEN9021 Principles of Programming or something equivalent, so we won't be teaching it to you. This lesson is a refresher on the aspects of Python that are particularly important for this course.

In slide focusses on Numpy. Click on the below links to practise Numpy:

[Practise link 1](#)

[Practise link 2](#)

Now the run the below code.

```
import numpy as np

#create and print arrays

a = np.arange(6)                # 1d array
print(a, ' a ')

b = np.arange(12).reshape(4,3)  # 2d array
print(b, ' b ')

c = np.arange(24).reshape(2,3,4) # 3d array
print(c, ' c ')
```

```
import numpy as np

a = np.zeros((2,2))    # Create an array of all zeros
print(a)               # Prints "[[ 0.  0.]
                        #           [ 0.  0.]]"

b = np.ones((1,2))     # Create an array of all ones
print(b)               # Prints "[[ 1.  1.]]"

c = np.full((2,2), 7)  # Create a constant array
print(c)               # Prints "[[ 7.  7.]
                        #           [ 7.  7.]]"
```

```
import numpy as np

# Create the following rank 2 array with shape (3, 4)
# [[ 1  2  3  4]
#  [ 5  6  7  8]
#  [ 9 10 11 12]]
a = np.array([[1,2,3,4], [5,6,7,8], [9,10,11,12]])

# Use slicing to pull out the subarray consisting of the first 2 rows
# and columns 1 and 2; b is the following array of shape (2, 2):
# [[2 3]
#  [6 7]]
b = a[:2, 1:3]

# A slice of an array is a view into the same data, so modifying it
# will modify the original array.
print(a[0, 1])    # Prints "2"
b[0, 0] = 77      # b[0, 0] is the same piece of data as a[0, 1]
print(a[0, 1])    # Prints "77"
```

# [10]] (3, 1)"

---

# Numpy Array Math

Run the below [Python code](#) on an array.

```
import numpy as np

x = np.array([[1,2],[3,4]], dtype=np.float64)
y = np.array([[5,6],[7,8]], dtype=np.float64)

# Elementwise sum; both produce the array
# [[ 6.0  8.0]
#  [10.0 12.0]]
print(x + y)
print(np.add(x, y))

# Elementwise difference; both produce the array
# [[-4.0 -4.0]
#  [-4.0 -4.0]]
print(x - y)
print(np.subtract(x, y))

# Elementwise product; both produce the array
# [[ 5.0 12.0]
#  [21.0 32.0]]
print(x * y)
print(np.multiply(x, y))

# Elementwise division; both produce the array
# [[ 0.2          0.33333333]
#  [ 0.42857143  0.5         ]]
print(x / y)
print(np.divide(x, y))

# Elementwise square root; produces the array
# [[ 1.          1.41421356]
#  [ 1.73205081  2.         ]]
print(np.sqrt(x))

#https://cs231n.github.io/python-numpy-tutorial/
```

## Inner products

Note that unlike MATLAB, `*` is elementwise multiplication, not matrix multiplication. We instead use the `dot` function to compute inner products of vectors, to multiply a vector by a matrix, and to multiply matrices. `dot` is available both as a function in the [Numpy](#) module and as an instance method of array objects.

Run the code.

```
import numpy as np

x = np.array([[1,2],[3,4]])
y = np.array([[5,6],[7,8]])

v = np.array([9,10])
w = np.array([11, 12])

# Inner product of vectors; both produce 219
print(v.dot(w))
print(np.dot(v, w))

# Matrix / vector product; both produce the rank 1 array [29 67]
print(x.dot(v))
print(np.dot(x, v))

# Matrix / matrix product; both produce the rank 2 array
# [[19 22]
#  [43 50]]
print(x.dot(y))
print(np.dot(x, y))
```

An error occurred.

---

Try [watching this video on www.youtube.com](https://www.youtube.com), or enable JavaScript if it is disabled in your browser.

An error occurred.

---

Try watching this video on [www.youtube.com](https://www.youtube.com), or enable JavaScript if it is disabled in your browser.

<https://cs231n.github.io/python-numpy-tutorial/>

```
#source: https://cs231n.github.io/python-numpy-tutorial/

import numpy as np

x = np.array([[1,2],[3,4]])

print(np.sum(x)) # Compute sum of all elements; prints "10"
print(np.sum(x, axis=0)) # Compute sum of each column; prints "[4 6]"
print(np.sum(x, axis=1)) # Compute sum of each row; prints "[3 7]"

print(' -----')

x = np.array([[1,2], [3,4]])
print(x)      # Prints "[[1 2]
               #         [3 4]]"
print(x.T)    # Prints "[[1 3]
               #         [2 4]]"

# Note that taking the transpose of a rank 1 array does nothing:
v = np.array([1,2,3])
print(v)      # Prints "[1 2 3]"
print(v.T)    # Prints "[1 2 3]"

print(' -----')

# We will add the vector v to each row of the matrix x,
# storing the result in the matrix y
x = np.array([[1,2,3], [4,5,6], [7,8,9], [10, 11, 12]])
v = np.array([1, 0, 1])
y = np.empty_like(x) # Create an empty matrix with the same shape as x

# Add the vector v to each row of the matrix x with an explicit loop
for i in range(4):
    y[i, :] = x[i, :] + v
```

```
# Now y is the following
# [[ 2  2  4]
#   [ 5  5  7]
#   [ 8  8 10]
#   [11 11 13]]
print(y)
```

---

# Basic functions

Basic functions to check conditions (if and else)

Run the python program by activating the terminal and type and enter: `python function.py`

timepass exercise:

update function, `speed_function_super()` so that considers different fine for speed of 100 - 119, 120 - 139 and 140 - 159, based on `speed_function_advanced()`. You can create fine based on your own rules, similar to 80 -100 limit based on `speed_function_advanced()`.



---

# Functions with Jupyter Notebook

Simple function example for speed calculator

# Numpy array examples

```
import numpy as np

import random

def numpy_lists():

    magic = np.random.rand(3,4)
    magic_three = np.random.rand(3,4,2)
    # homework, write a function for summing 3D magic
    magic_one = np.random.rand(10)

    #print(magic)
    #print(magic_one)
    print(magic_three)
    print(magic.shape)

    magic_sum = sum_numpy(magic)
    print( magic_sum, ' is magic sum')
    return magic_sum

def sum_numpy(a):
    sum = 0
    for x in range(a.shape[0]):
        for y in range(a.shape[1]):
            print(x, y, ' *' , a[x][y] )
            sum = sum + a[x][y]
    return sum

def sum_3D(a):
    print('homework - sum elements of 3d numpy array')

    # use nested for loops
    #https://www.ict.social/python/basics/multidimensional-lists-in-python

def main():

    print('running: numpy_lists()')
    sum = numpy_lists()
    print(sum, ' is sum')

if __name__ == '__main__':
    main()
```

```

import numpy as np
import random
import sys # for end or endlne in nested food loops, python 3

def nested_loops():

    print(' nested loops function')
    #https://www.ict.social/python/basics/multidimensional-lists-in-python

    length = 3
    width = 4
    height = 3

    two_dimen = np.random.rand(length, width)
    print(two_dimen)

    first_row = two_dimen[0,:]
    second_col = two_dimen[:,1]

    print(first_row, ' first_row')
    print(second_col, ' second_col')

    three_dimen = np.zeros((length, width, height))

    print(three_dimen)

    for i in range(length):
        for j in range(width):
            two_dimen[i,j] = i*j

    print (two_dimen)

    #print (k, end = ' ') # end refers new line
    print (' 3 nested ')
    for i in range(1,3):
        for j in range(1,3):
            for k in range(1,3):
                z = i*j * k
                print (z) # end refers new line

    print (' 3 nested save to np array ')

    for i in range(length):
        for j in range(width):
            for k in range(height):
                three_dimen[i,j,k] = i*j * k
    print(three_dimen)

```

```
#operations to select parts of 2d array
```

```
def main():  
  
    print('running: nested_loops()')  
    nested_loops()  
  
if __name__ == '__main__':  
    main()
```

---

## Sum Matrices

Sum elements in a 2D Matrix example given using nested loops ( to demonstrate nested loops)

Challenge: implement a function for 3D case using nested loops

---

# Nested loops: 2D and 3D

Example of operations with nested loops

---

## Advanced Numpy methods

*This code slide does not have a description.*

---

## Load and save files

*This code slide does not have a description.*



---

# Mandlebrot example

[https://en.wikipedia.org/wiki/Mandelbrot\\_set](https://en.wikipedia.org/wiki/Mandelbrot_set)

source: #<https://www.codingame.com/playgrounds/2358/how-to-plot-the-mandelbrot-set/mandelbrot-set>

An error occurred.

---

Try watching this video on [www.youtube.com](https://www.youtube.com), or enable JavaScript if it is disabled in your browser.

---

# Pandas data processing

```
import numpy as np
import pandas as pd

s = pd.Series([1, 3, 5, np.nan, 6, 8])
print(s, ' s series')

dates = pd.date_range('20130101', periods=6)
print(dates, ' dates')
```

source: [https://pandas.pydata.org/pandas-docs/stable/getting\\_started/10min.html](https://pandas.pydata.org/pandas-docs/stable/getting_started/10min.html)

```
import numpy as np
import pandas as pd

data = np.array(['', 'Col1', 'Col2'], ['Row1', 1, 2], ['Row2', 3, 4])

print(data)

print(pd.DataFrame(data=data[1:,1:], index=data[1:,0], columns=data[0,1:]))
```

```
import numpy as np
import pandas as pd

df = pd.DataFrame(np.array([[1, 2, 3], [4, 5, 6]]))

# Using `iloc[]`
print(df.iloc[0][0]) #https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.
#https://www.shanelynn.ie/select-pandas-dataframe-rows-and-columns-using-iloc-loc-and-ix/

# Using `iat[]`
print(df.iat[0,0])
```

source: <https://www.datacamp.com/community/tutorials/pandas-tutorial-dataframe-python>

tutorial: <https://realpython.com/pandas-python-explore-dataset/>

<https://www.geeksforgeeks.org/python-pandas-dataframe/>

<https://www.geeksforgeeks.org/python-pandas-working-with-text-data/>



An error occurred.

---

Try watching this video on [www.youtube.com](http://www.youtube.com), or enable JavaScript if it is disabled in your browser.

---

# Data cleaning with Pandas

External tutorials here:

1. <https://www.kaggle.com/regivm/data-cleaning-and-eda-tutorial>
2. <https://realpython.com/python-data-cleaning-numpy-pandas/>

An error occurred.

---

Unable to execute JavaScript.

An error occurred.

---

Try watching this video on [www.youtube.com](https://www.youtube.com), or enable JavaScript if it is disabled in your browser.

notebook for above video: <https://github.com/KeithGalli/pandas>

---

# Object oriented programming

## Classes

[https://www.w3schools.com/python/python\\_classes.asp](https://www.w3schools.com/python/python_classes.asp)

```
# source: https://www.csdojo.io/class

class Robot:
    def __init__(self, name, color, weight):
        self.name = name
        self.color = color
        self.weight = weight

    def introduce_self(self):
        print("My name is " + self.name)

# r1 = Robot()
# r1.name = "Tom"
# r1.color = "red"
# r1.weight = 30
#
# r2 = Robot()
# r2.name = "Jerry"
# r2.color = "blue"
# r2.weight = 40

r1 = Robot("Tom", "red", 30)
r2 = Robot("Jerry", "blue", 40)

r1.introduce_self()
r2.introduce_self()
```

An error occurred.

---

Try watching this video on [www.youtube.com](https://www.youtube.com), or enable JavaScript if it is disabled in your browser.

## Inheritance

[https://www.w3schools.com/python/python\\_inheritance.asp](https://www.w3schools.com/python/python_inheritance.asp)

```
# source: https://www.csdojo.io/class2

class Robot:
    def __init__(self, n, c, w):
        self.name = n
        self.color = c
        self.weight = w

    def introduce_self(self):
        print("My name is " + self.name)

class Person:
    def __init__(self, n, p, i):
        self.name = n
        self.personality = p
        self.isSitting = i

    def sit_down(self):
        print("My name is " + self.name)

r1 = Robot("Tom", "red", 30)
r2 = Robot("Jerry", "blue", 40)

p1 = Person("Alice", "aggressive", False)
p2 = Person("Becky", "aggressive", True)

p1_robot_owned = r2
p2_robot_owned = r1

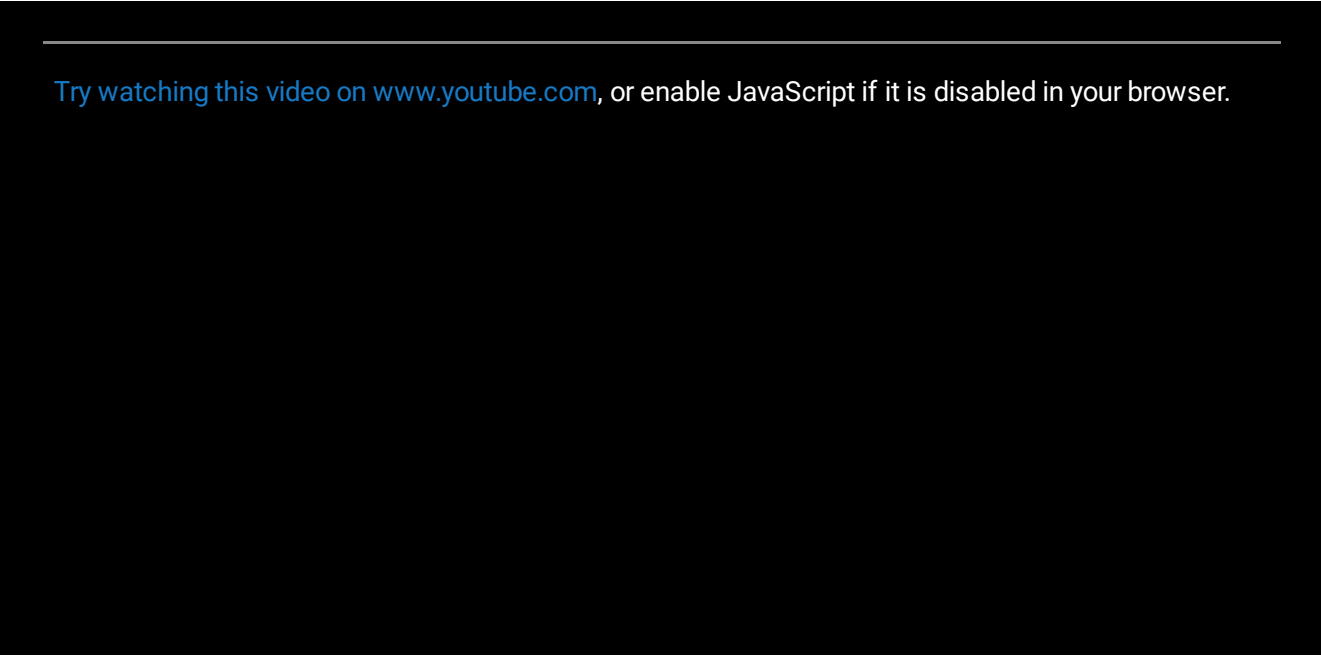
p1_robot_owned = r2

p1_robot_owned.introduce_self()
```

An error occurred.

---

Try watching this video on [www.youtube.com](https://www.youtube.com), or enable JavaScript if it is disabled in your browser.



some other examples:

<https://www.programiz.com/python-programming/object-oriented-programming>

<https://python.swaroopch.com/oop.html>