COMPUTING SUBJECT:	Machine Learning
TYPE:	WORK ASSIGNMENT
IDENTIFICATION:	Python Basic No. 1
COPYRIGHT:	*Michael Claudius*
DEGREE OF DIFFICULTY:	Easy
TIME CONSUMPTION:	1 hour
EXTENT:	< 60 lines
OBJECTIVE:	Basic understanding Simple operations on 1- dim integer lists

COMMANDS:

IDENTIFICATION: Python Basics/MICL

The Mission

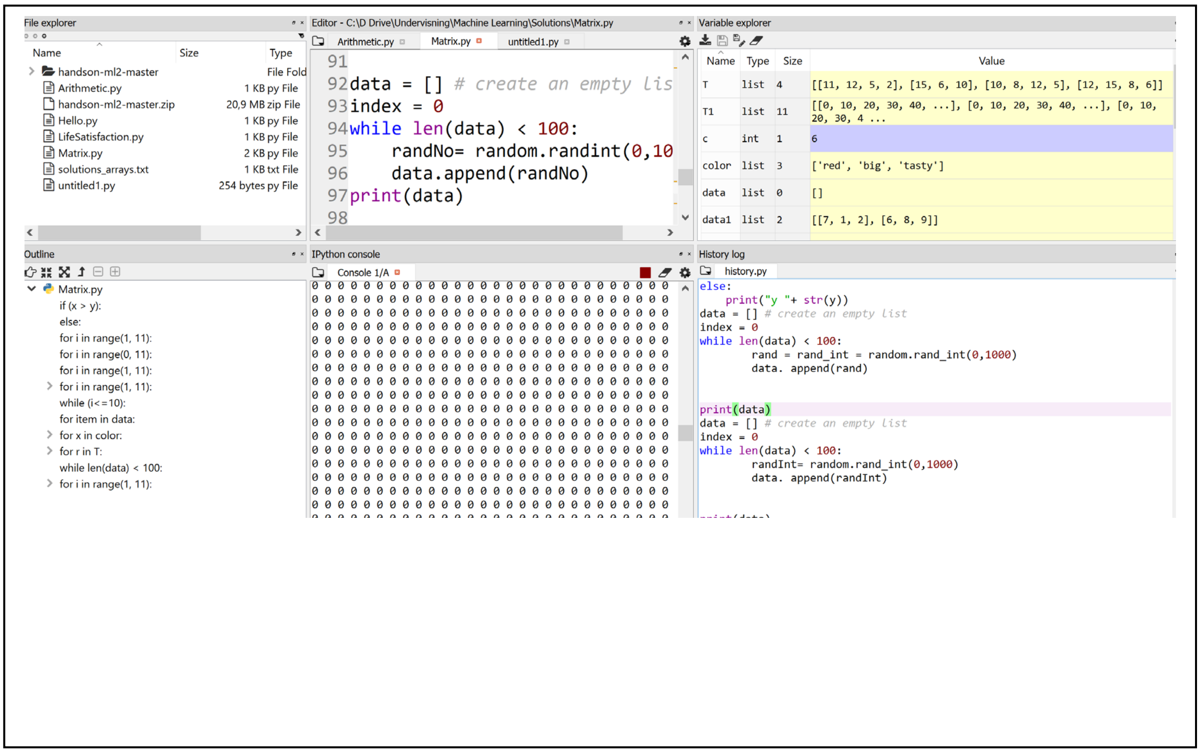
This is a crash course in basic Python. The goal is you will be able to read and understand the code used in the course. Together we are going to do a Python program, with some simple code. In order to keep a good speed no GUI no structure of program. Just coding.

The problem

To do a Python program with simple declarations, initialising & changing values, printing, find maximum, minimum, sum and average of the elements in an integer array.

Useful links
https://www.w3schools.com/Python/default.asp
https://docs.python.org/3/library/random.html
https://www.tutorialspoint.com/python_data_structure/python_2darray.htm
https://stackoverflow.com/questions/6142689/initialising-an-array-of-fixed-size-in-python

Spyder in Anaconda is a development environment with very good interaction and has many possibilities. It looks like this when all windows are open (non standard, which is an overkill at the moment!).



Name	Size	Type
handson-ml2-master		File Fold
Arithmetic.py	1 KB	py File
handson-ml2-master.zip	20,9 MB	zip File
Hello.py	1 KB	py File
LifeSatisfaction.py	1 KB	py File
Matrix.py	2 KB	py File
solutions_arrays.txt	1 KB	txt File
untitled1.py	254 bytes	py File

```
data = [] # create an empty lis
index = 0
while len(data) < 100:
    randNo= random.randint(0,10
    data.append(randNo)
print(data)
```

Name	Type	Size	Value
T	list	4	[[11, 12, 5, 2], [15, 6, 10], [10, 8, 12, 5], [12, 15, 8, 6]]
T1	list	11	[[0, 10, 20, 30, 40, ...], [0, 10, 20, 30, 40, ...], [0, 10, 20, 30, 4 ...
c	int	1	6
color	list	3	['red', 'big', 'tasty']
data	list	0	[]
data1	list	2	[[7, 1, 2], [6, 8, 9]]

```
Matrix.py
    if (x > y):
    else:
    for i in range(1, 11):
    for i in range(0, 11):
    for i in range(1, 11):
    for i in range(1, 11):
    while (i<=10):
    for item in data:
    for x in color:
    for r in T:
    while len(data) < 100:
    for i in range(1, 11):
```

```
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
else:
    print("y "+ str(y))
data = [] # create an empty list
index = 0
while len(data) < 100:
        rand = rand_int = random.rand_int(0,1000)
        data. append(rand)


print(data)
data = [] # create an empty list
index = 0
while len(data) < 100:
        randInt= random.rand_int(0,1000)
        data. append(randInt)
```

Assignment 1: Application program, declaration

Start Anaconda and launch Spyder.

In Spyder type the following lines:

```
int x = 1
y = 3
z = 5
sum = x+y+z
Print(sum)
```

Choose: Debug -> Debug

Unfortunately there are errors.
Notice the red spot (Code analysis: Invalid syntax). Correct the errors.
Choose: Run or Click on the green triangle (extremely small)
Note: What did you learn ? Case sensitive. Type declaration
Write it down.

Now print out the result with a text by calling *print*:

```
print("Sum is: " + sum)
print("Sum is: " + str(sum) + " DKK")
#Finally use the f-strings, a new feature in Python 3
print(f"Sum is: + {sum} +  DKK")
```

Highlight the four lines.
Choose: Run -> Run cell

Unfortunately there are errors. Correct the errors.
Note: What did you learn? Type casting. Comments.
Write it down.
Note: One could also have use the format commands.

Assignment 2: Application program, loops

For-loops in Python are actually for-each loops.
Try to print out a list of squares:

```
for i in range(1, 10):
    print(i,i*i)
  print()
```

Ups *Indentation error*. Correct and run again.
Change the print sentence to:

```
print(i,i*i, end = “ “)
```

Note: What did you learn? Indentation yes; no brackets.

Assignment 3: Application program, nested for loops

The little table of the first 10 numbers looks like this

	1	2	3	4	5	6	**7**	8	9	10
1	1	2	3	4	5	6	7	8	9	10
2	2	4	6	8	10	12	**14**	16	18	20
3	3	6								30
4	4	8								40
5	5	10								.
6	6	12								.
7	7	.								.
8	8	.							72	.
9	9	18	27	36	.	.	.	.	81	90
10	10	20	30	40	50	60	70	80	90	100

Ring a bell from primary school!? Notice: 2x7 = 14!!
Now you have to print out the numbers in little table using nested loops; i.e. a for-loop inside another for-loop. The formatting (a real table format) does not matter; it is the loops and content we focus on.
*Tip: print(i*j, end = “ “)*

Assignment 4: Application program, while-loops
Try to print out a list of squares using a while-loop:

```
i = 1
while (i<=10):
     ........
```

Write the rest of the code lines yourself.
Debug and Run.

Assignment 5: Application program, if-else

Try to print out the maximum of two numbers: x and y

```
x = 7
y = 23
if (x > y):
    print("x: "+ str(x))
else:
    print("y: "+ str(y))
```

Save and compile and run! *No problems here, I hope!*

Assignment 6: Method maximum of two numbers

Extend with a new method *findMax(x,y),* which returns the maximum of the parameters

```
def findMax(x,y):
    if (x > y):
. . . . . .
```

Call the method and print out the maximum value.

Tip: Be inspired by assignment 5 but remember return !

Now we will switch domain to list (array).

Assignment 7: Application program 1-dim array (list) of numbers

Lists are actually the Python version of 1-dim arrays.
Declare a list with the primes:

```
data = [1,2,3,5,7,11,13,17,19,23]
print(data)
for item in data:
    print(item, end = " ")
```

Run as a cell. Explain the output.
Add some more lines:

```
print(data[3])
data[0] = 17
data.append(37)
print(data)
```

Run. What do you expect to see?
Add more lines:

```
data.insert(4,99)
print(data)
```

Run. What do you expect to see now?
What did you learn? Difference between append and insert.

Now we will switch domain to methods on lists.

Assignment 8: Method maximum

Extend with a new method *find_max(data),* which returns the maximum of the numbers in the list *data*.

```
def find_max(data):
    my_max_ = max(data)
    return my_max
```

Call *find_max* with data as parameter:

```
max_num = find_max(data)
```

and print out the maximum number.

Assignment 9: Method minimum

Extend with a new method *find_min(data),* which returns the minimum value of the numbers in the parameter *data*. Remember to introduce extra printing-sentences for this.

Assignment 10: Method Sum

Extend with a new method *find_sum(data),* which returns the sum value of the numbers in the array parameter *data*. Remember to introduce extra printing-sentences for this.
Follow the style from assignment 4.

Assignment 11: Average

Extend with a new method *average(data),* which returns the average of all numbers in the list.
Utilize *find_sum* method from previous assignment.

For the fast ones only

Assignment X: The 5 highest numbers

How to find the 5 highest numbers in a list. How to return this from a method.
Or maybe you can make it flexible so it can be any highest numbers like 6 or 11 etc.

Congratulations you have passed !