PROGRAMMING WITH PYTHON

DAY 1 – MAAIKE VAN PUTTEN

GOALS OF PYTHON BASICS

- Become familiar with Python
- Use Python to automate manual steps, such as:
  - Configuration
  - Search logs
  - Read files / write files
  - Calculations
  - Call REST APIs
  - Write a simple web app
  - And more!
- Know how to go from there

TODAY'S SCHEDULE

- Introduce yourself!
- Set up development environment
- Python basics (incl methods)
- Collections (list, set, dictionary = key value pair)
- Read and write files
- Stdin, stderr, stdout
- Environment variables read / set
- Pyinstaller / executables

INTRODUCTION ROUND

- Name
- Job / background / field / experience
- Home / family / kids / pets
- Hobbies and interests
- Etc ☺

WHAT IS PYTHON?

- High level programming language
- Easy, readable
- Born in the Netherlands
- Python 3

WHAT IS PYTHON USED FOR?

- Application development
- Data analysis / machine learning
- (Automation) Scripting
- Internet of things
- Glue language

SET UP DEVELOPMENT ENVIRONMENT

- Step 1: https://www.python.org/downloads/ to get latest version
- Step 2: run .exe you just downloaded
- Step 3: https://www.jetbrains.com/pycharm/download/ get latest version of Pycharm
- Run installation
- Open Pycharm

FIRST PROGRAM!

- Open a new file and write: print(“Hello world”)
- Run
- Check console: does it say “Hello world”?

VARIABLES

- Any element or factor that is liable to change (variation)

- For example: 2x
- x is different in 2x = 10 than in 2x = 12

- So if I say: greeting = x
- x can be "hello" the one time, and "bye" the other time, depending on the value of x

DATA TYPES

- String
- Integer
- Float
- Many more

DEFINE VARIABLE IN PYTHON

naam = 8

naam2 = “Hoi”

naam3 = 8.5

naam4 = True

OPERATIONS ON DATA TYPES

- +
- -
- / and //
- *
- **
- %

EXERCISE

- Write a program that takes the lengths of the two sides of a right triangle and calculates the length of the hypotenuse.

EXERCISE: OPERATION WITH DATA TYPES

- Play around with operation on data types
- Answer this question: What is the precedence from highest to lowest?
- Come up with a difficult formula
- When we're done we are going to look to each others code to see whether we can guess the result

EXERCISE: OPERATION WITH DATA TYPES

- Challenge!

- I want to do calculations with a time
- So if it's 10.00 and I add 61 minutes, the result should be 11.01
- Don't use if yet, we didn't discuss it and you don't need it

- If you're stuck, chat me for a hint

PYTHON BUILT-IN FUNCTIONS

- What are functions?

- Difference between variable and function

Examples:

- print()
- input()

EXERCISE:

- Write a program that asks that user to enter an article, a noun, and a verb.
- The program then creates a sentence of the form: *article* *noun* *verb*.

EXERCISE:

- Ask the user for his/her favorite color
- Save the result in a variable
- Ask the user for the first tool that comes to mind
- Save the result in a variable
- Print to the console: "combining your input I'm getting a *color* *tool*."

CASTING

- Converting from one data type, to another

- str()
- int()
- float()

- And more, but not for now ☺

EXERCISE:

- Ask the user for two numbers
- Add these numbers
- Print the calculation you did and the result

- Ask the user for two numbers
- Divide these numbers
- Print the calculation you did and the result

- Ask the user for name and highest age (s)he'll have this year
- Calculate the year of birth
- Print to the console: *username* is born in year *year*

IF ELIF ELSE

- If is used for making decisions
- If a certain condition is met, a block of code will be executed
- If not, it won't
- If no condition is met and there is an else block, the else block will be executed

EXAMPLE

```
x = int(input('Geef een nr: '))
if x < 10:
    print(x, "kleiner dan 10")
elif x < 20:
    print(x, "kleiner dan 20, groter of gelijk 10")
else:
    print(x, "gelijk of groter 20")
```

EXERCISE:

- Express each of the following statements using an if block:
  - a. If x divided by y is 5, then set x to 100.
  - b. If x times y is 5, then set x to 1.
  - c. If x is less than y, then double the value of x.
  - d. If x is greater than y, then increment the value of x by 1.

EXERCISE:

- Write a program that prompts the user to enter five test scores between 1 and 10. The program will then count the number of scores that are greater than 7.

- Express each of the following statements using an if/else block:
  - a. If x times y is 8, then set x to 1; otherwise, set x to 2.
  - b. If x is less than y, then double the value of x; otherwise, increment x by 1.
  - c. If x is greater than y, then increment both by 1; otherwise, decrement both by 1.

LIST

[“list”, “with”, “different”, “elements”, “could”, “be”, 5, “too”]

Empty list: []

Listname[0] >> gets first element from list

Listname[-1] >> gets last element

Listname[-2] >> gets second last element, etc

BUILT-IN LIST FUNCTIONS

Length of list = len(*listname*)

Add to list: listname.append("new item")

Insert at an index: listname.insert(1,"second item")

Remove item at index: listname.pop(1)

Remove from end: listname.pop()

Remove item with value: listname.remove("second item")

Sort the list: listname.sort()

Reverse the items in a list: listname.reverse()

EXERCISE:

- Create a list with your favorite movies
- Create a list with your favorite friends
- Create a list with your favorite numbers

EXERCISE:

- Create a list with your favorite movies
- Create a list with your favorite friends
- Create a list with your favorite numbers

- Get the length of each list
- Sort the lists
- Remove your least favorite favorite one from each list
- Append an alternative
- Make sure that the lists are eventually in naturally reversed order (so Z to A and 9 to 0)

OTHER FUNCTIONS ON LISTS

- Make a new list object of a list that's sorted: sorted(listname)
- Get the max value: max(listname)
- Get the min value: min(listname)
- Get the sum: sum(listname)

EXERCISE:

- From your number list, get the min, max, length and sum
- Print all the lists sorted, without actually sorting them
- To prove, also print the original list afterwards

EXERCISE:

We have a shop with 4 products and their prices:

- articles = ["Cheese", "Bread", "Coffee", "Wine"]
- prices = [8, 2, 4, 3]

Create a program that:

- prints a list of products and asks the user to choose one
- when the user makes a choice, he can enter an amount
- calculate and print the total price for his order

BONUS EXERCISE:

- https://www.codingame.com/home choose temperatures

WHILE

- While repeats a code as long as a certain condition is met
- Break
- Continue

```
while i < 20:
        i+=1
        if i == 13:
                continue
        print(i)
```

EXERCISE

- Write a program that asks the user to enter three numbers. The program will then determine and print the largest of the three numbers.
- Create an input validation loop that only accepts numbers in the range of 1 through 10.
- Write a script that asks the user, "Are you sure you want to quit [Y, N]?" The script then checks the user's input and only accepts the letters Y and N as valid answers.

EXERCISE:

Create a while loop for the following:

- Ask the user for a word
- Store this word in a list (use append)
- Print the list
- When the user enters nothing, stop the program

Deduplication

- Change the program so that, every time the user gives a word, you test whether it is already in the list. If it is, you don't add it.

Lengths

- Ask the user to enter 4 words. Put those in a list with append.

After all words have been entered, print:

- The shortest and longest word
- The average length of all words

BOOLEAN

- True
- False

- and
- or
- not
- chaining

QUESTION

There is built-in Python function called bool()

You can use it on pretty much everything

When do you think it will return True? And when False?

FOR

Loop over elements of an iterable.

```
x = [1, 2, 3]

for y in x:
    print(y)
```

EXERCISE

- Loop over the letters of a string and print them to the console

- Create a list that contains the days of the week
- Create a list containing the working days of the week
- Loop over the elements of the first list and print days that are not in the working days

- We haven't seen this yet, but how can you look over the first 5 elements in a list?
- And the last five?

RANGE()

- Function that returns a sequence of numbers
- range(start, end, step)

- range(5) → return numbers 0, 1, 2, 3, 4
- range(2, 5) → returns numbers 2, 3, 4
- range(2, 5, 2) → returns numbers 2, 4

- You can use it directly in a for loop
  - for x in range(0, 100, 5):

EXERCISE

- Make a list with the numbers 1 to a milion
- Sum them and see how fast python can do this

- Use range() to make a list of 0 to 1000 and use this to create a new list with all these numbers squared

SET

- Unordered collection

example = {“some”, “set”, “elements”}

```
for x in example:
  print(x)
```

TUPLE

- ()
- Fixed values
- Overwriting entire tuple is possible

- When to use?

Faster than lists

Protect against change

As a key to a dictionary (dictionary is coming soon)

EXERCISE

- Imagine we are having a very particular company: we produce huge cubes
- We want to create a list of tuples for cubes with a length of 0 to 10
- Each tuple should contain: length of a side, cubic meters and the price per cubic meter, and total price
- The price per cubic meter starts with 10 for one cubic meter, and is being decreased with 0,01 for each extra cubic meter the cube has
- Bonus: where is the line that the cubes become free based on the above calculation?

DICTIONARY

- {}
- Key value pairs

```
thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
```

```
thisdict["year"] = 2018
```

LOOP THROUGH DICTIONARY

```
for x, y in thisdict.items():
  print(x, y)

for x in thisdict.values():
  print(x)

for x in thisdict.keys():
  print(x)
```

DICTIONARY FUNCTIONS

- .items
- .keys
- .values
- .popitem
- .pop(key)
- .get(key)

EXERCISE

- Create three dictionaries of your favorite sport players, store some important properties of a sport player, like what sport, name, year (s)he was born, description
- Loop over the items in each dictionary, write the content to a file to create mini biographies about each
- Create a list with these dictionaries
- Loop over the list, and in each iteration, loop over the items in the dictionary and print these to the console

GAME TIME! – EXERCISE 1

- Write a game that takes a random integer
- And makes the user guess that integer until he guess correctly
- Show higher or lower to help the user get it right

- Bonus:
  - Keep the nr of guesses
  - Collect all the guessed numbers and print them later
  - If someone uses less than 5 guesses, print that (s)he's awesome
  - Make the program safe for non-numeric input
  - Save this data to a file: guesses, nr of guesses, correct number

- Bonus bonus:
  - Write a script that does the guessing, and log the guesses it did
  - Run it 10000 times to get the average nr of guesses
  - See if you can optimize the script even further

WRITE YOUR OWN FUNCTIONS

- Arguments
- Return

- def

```
def goodmorning(name):
        print("Goodmorning," + name + "!")

goodmorning("Maaike")
```

EXERCISE

- Split the game time exercise up in separate functions
- Call these functions

TYPES OF ARGUMENTS IN FUNCTIONS

```
def goodmorning(name, day_name):
        print("Goodmorning," + name + " on " + dayname + "!")
```

Positional:

```
goodmorning("Maaike","Wednesday")
```

Keyword arguments:

```
goodmorning(day_name="Wednesday", name="Maaike")
```

OPTIONAL ARGUMENTS IN FUNCTIONS

```
def goodmorning(name, day_name="this day"):
        print("Goodmorning," + name + " on " + dayname + "!")
```

goodmorning("Maaike") → will print: goodmorning, Maaike on this day!

RETURN

```
def goodmorning(name, day_name="this day"):
        return "Goodmorning," + name + " on " + dayname + "!"
```

goodmorning("Maaike") → won't show anything, but returns: goodmorning, Maaike on this day!

EXERCISE

- Write a function that returns the sum of letters in a name, in which each letter corresponds with its position in the alphabet

LET'S EXPLORE

- What happens if you send a list to a function, and change the list in the function, but return nothing? Is the list changed everywhere?
- What if you want to have a variable number of arguments?
- What if you want to have a variable number of key value pairs as arguments?

EXERCISE

- Change your lyric function to accept many pieces of lyrics

MODULES

- A module is just file with Python functions and properties.
- You can import a module using the "import" keyword on top of the class
- There are many useful standard modules, e.g. math
- A module name is the file name without the .py extension
- The module name is the value of the global variable __name__

EXERCISE MODULES:

- Find a module to generate a random integer
- Find a module to get the ceiling and floor of x
- Play around with turtle module!

EXPORT YOUR FUNCTIONS IN YOUR OWN MODULES

- We've been writing modules al along! You save them with the extension .py
- After that you can import them using the import statement

- Exercise:

Create a separate .py file with a function to print information about a user to a file

Import this function to your current file

Call the function

Bonus: how to import not an entire module but only one function?

OPEN/READ/WRITE FILES

- open(name)
- read() or readlines()
- close()

- open(name,“w+”)
- write(“blabla”)
- close()

EXERCISE

- Create a file
- Write a little story about the new things you've learnt today

- Read from the file and print it to the console

STDIN, STDOUT, STDERR

- Standard input – user input information is read from this file-handle. The input is given to the standard input.
- Standard output – normal information is written to this file-handle. The output is returned via the Standard output
- Standard error - error information is written to this file-handle. Errors are returned via the Standard error

PYTHON SYS MODULE

File objects for stdin, stdout, and stderr:

- sys.stdin
- sys.stdout
- sys.stderr

- Similar to a file, it can be opened and closed

- Examples:
- https://www.askpython.com/python/python-stdin-stdout-stderr

ENVIRONMENT VARIABLES READ / SET

```
import os

print(os.environ['HOME'])
print(os.environ['PATH'])
os.environ['HOME'] = "set to whatever"
```

GAME TIME! – EXERCISE 2

- Monty hall problem:
  - Three doors
  - Behind one is a big prize that you win when you choose that door
  - Choose a door
  - Monty opens one of the others that does not contain the prize
  - And asks you if you'd like to switch to the other unopened door, or stick with your unopened door
- Use Python to simulate this situation and answer the following question:
  - If you switch doors when Monty gives you the opportunity, do you have a bigger chance of winning?
- Bonus: right results to a file

PYINSTALLER

pip install pyinstaller

pyinstaller /path/to/main.py

Very many options, for example:

pyinstaller –onefile /path/to/main.py

SUMMARY

- Set up development environment
- Python basics (incl methods)
- Collections (list, set, dictionary = key value pair)
- Read and write files
- Stdin, stderr, stdout
- Environment variables read / set
- Pyinstaller / executables

BONUS EXERCISE

- Go to Codingame and choose "The Descent"

NEXT SESSION

- Libraries
- API calls
- Pandas
- Convert van bestanden
- GEF
- GUI voor desktop (tkinter en pyqt5)