Exercise Sheet 01 – `print('Hello Python!')`

Sebastian Höffner Aline Vilks

Deadline: Mon, 10 Apr 2017 08:00 +0200

Submission

By the end of this sheet you will have a number of different files to submit. In Stud.IP you will have a directory for your own group, please upload them there. It is easier for you if you just archive all files and upload your archive, but it is okay if you upload them one by one.

Preamble

Throughout the course we will sometimes (especially in the beginning) refer to the command line or the terminal, and then again use them interchanged. Do not be disheartened! It is the same concept, but some operating systems call it one way and others the other. On Ubuntu you can open it up with the keyboard shortcut `Ctrl + Alt + T`. On MacOS press `Cmd + Space` and type `Terminal` to open it. On Windows, press the `Win + R` and type `cmd`, then hit return.

This is the basic "prompt" (... another term) where you can unleash the powers of your operating system (and break things!). Try `ls` on Mac/Linux or `dir` on Windows[1]. You can navigate around with `cd` (go back up with `cd ..`).

Errors & Web Search

Although most exercises should be solvable with what we provide in the slides and cover in class, you will have problems solving some of the exercises – that is by design. For example to solve the bonus exercise 1d) on this sheet, you might need to search around a little bit.

[1] No worries, this won't break anything, it just shows the files of your current directory.

Exercise 1: Your development environment

Installing Python

Miniconda is a package management system which allows us to keep the administrative overhead of installing Python to a minimum and similar for all operating systems.

- Download Miniconda (Python 3.6) from https://conda.io/miniconda.html.
- Install it. Make sure it is in your path[2].
- Open your terminal / command line and run the following to install an IDE[3] we will and two packages we might use:

  ```
  conda -y install pip spyder numpy matplotlib scipy
  ```

- For stuff used in e.g. Neuroinformatics, Machine Learning, Computer Vision, or other classes, run additionally:

  ```
  conda -y install pandas jupyter scikit-learn scikit-image
  ```

Exercise 2: Hello you!

Open your terminal or command line. Type `python` and hit return. This will start the python interpreter and you will see something like this:

```
Python 3.6.0 |Continuum Analytics, Inc.| (default, Dec 23 2016, 13:19:00)
[GCC 4.2.1 Compatible Apple LLVM 6.0 (clang-600.0.57)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Note that the second line (`[GCC ...] on darwin`) will most likely be different, but even the first might differ from what you see above. As long as you can see `Python 3.6.x` there and some `>>>`, you are all set.

Now type the following and hit return:

```
print("Hello Python!")
```

Output:

```
Hello Python!
```

[2] This is different for all operating systems. On Linux and MacOS it usually involves opening a file `~/.bashrc` (Linux) or `~/.bash_profile` (MacOS) and adding this line at the end: `export PATH="$PATH:~/miniconda3/bin"`. On Windows you have to go to your System settings and add `C:\Users\yourname\anaconda3` to the environment variable `Path` – if you really have to do it manually, try searching the web or ask us for help (Best before Sunday night).

[3] Integrated Development Environment

Modify the print statement you just typed so that it says each of the following:

1. `Hello [your name here]!`
2. `What's up?`
3. *Bonus*: `Are you "Monty"?`
4. *Bonus 2*: `Can't you just pretend you were "Monty"?`

Copy all of the successful statements (you may add your own) into an empty text file and call it `hello.py`. This is one of the submission files. If you don't get this to work, just upload a funny image and write the commands into the Stud.IP file description. We will then discuss this in the feedback sessions.

You can exit the python interpreter by using `exit()` or hitting the control key (`Ctrl`) and `d` (On Windows `Ctrl + Z` – even though we write a capital Z, you do not need the shift key!).[4]

Exercise 3: Turtles all the way down

In the late 20th century it became very popular to teach programming with a turtle[5]. Those turtles were little robots that could move around – check one out here: https://youtu.be/8wU6NdjTVTA. Did you notice that it drew on the sheet of paper? No? Watch again then.

Since we can not afford to buy a turtle for everyone, Python allows us to draw with a virtual turtle. Open your python interpreter again, as you did in the previous exercise.

To use the turtle, you first have to tell python that we need it:

```
import turtle
```

To show our turtle, use:

```
turtle.shape('turtle')
```

You should see a window like this:

Now you can let your turtle move around and draw with the following commands:

```
turtle.forward(100)
turtle.right(90)
turtle.backward(100)
turtle.left(90)
```

[4] You may find the key combination "control key and d" denoted as `Ctrl + D`, `^D`, `EOF`, or `EOT`. It means: "Here is the end of my input, you may exit", or in technical terms "end of file" or "end of transmission". Technically there is a difference between EOF (`^D`) and EOT (`^V ^D`) (and the Windows `^Z`), but for us this does not matter much.

[5] Check https://en.wikipedia.org/wiki/Turtle_(robot)[6] for more info. The concept is still popular today, although turtles are not enough anymore (https://studio.code.org/s/frozen/stage/1/puzzle/1[7]).

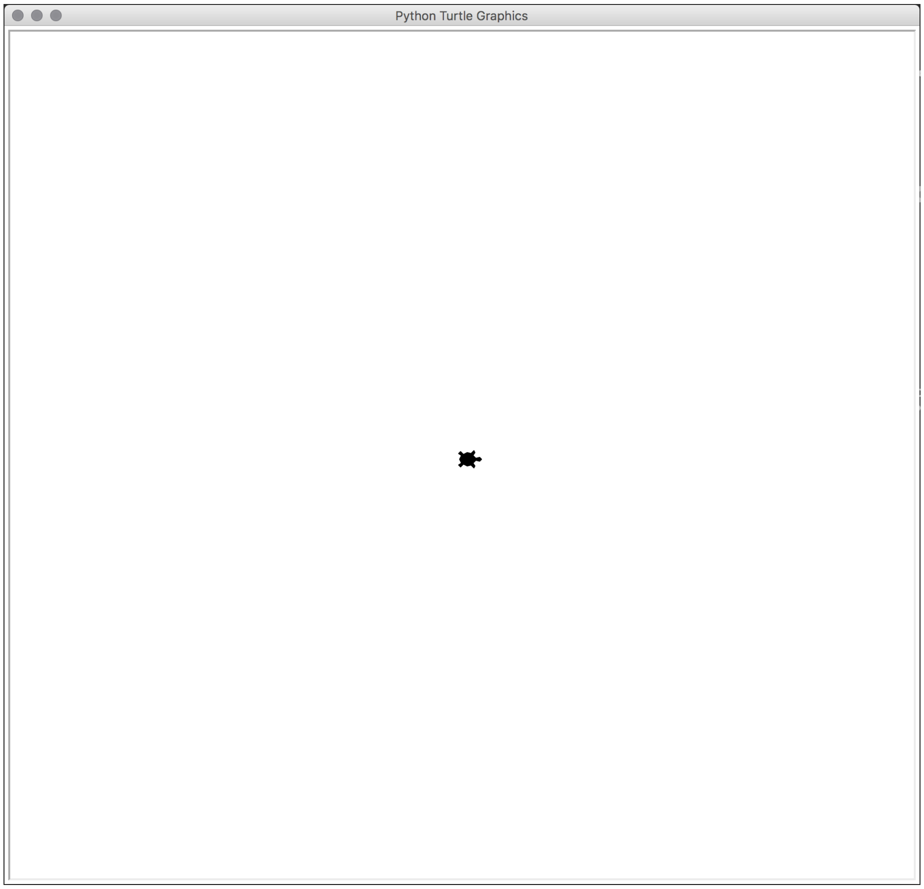

Figure 1: Turtle window

If you want to reset what your turtle drew, you can use `turtle.reset()`.

In Germany little kids learn to draw the *Haus vom Nikolaus* (St. Nicholas' house) in one stroke. It looks like this:

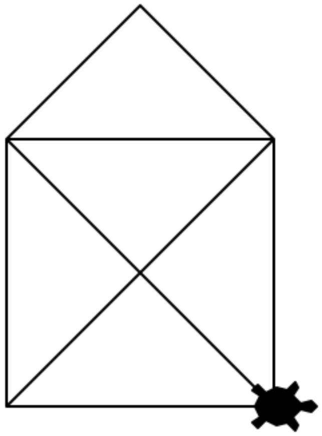

Figure 2: St. Nicholas' house

Can you give your turtle commands to draw something like this?

Store your successful commands in a file `nicholas.py` for your submission.