The purpose of this training assignment is to take your very first steps in Python. We will see how to use Python as a simple calculator, how to write an easy script using variables and lists and how to make a plot (with the module matplotlib). You should start working on these training assignment during the scheduled computer lab hours and complete them on your own at home.

This assignment has 12 tasks.

**Premisses**

Don’t forget to import the required modules for these tasks by executing

```python
from scipy import *
from pylab import *
import sys
```

If you get an error message of the form

```
NameError: name 'arctan' is not defined
```

you forgot at least one of these steps.

**Task 1**

Run in an IPython shell (console window at the down right corner of your Spyder window) the following two commands to check, that you installed the correct version of Python:

```python
sys.version
```

If you got a version number larger than 3, then the version is the correct one for this course.

**Task 2**

Check whether the expression

\[ x^2 + 0.25x - 5 \]

is zero for \( x = 2.3 \).
Task 3

Try out this code segment from the lecture

```python
L = [1, 2]
L3 = 3*L
```

and test the following commands:

- `L3[0]`
- `L3[-1]`
- `L3[10]`

Task 4

What does the following command do?

```python
L4 = [k**2 for k in L3]
```

Task 5

Concatenate `L3` and `L4` to a new list `L5`.

Task 6

Use the command `range` and a list comprehension to generate a list with 100 equidistantly spaced values between (and including) 0 and 1.

Task 7

What does this code do?

```python
s = 0
for i in range(0, 500):
    s = s + i
print(s)
```

and this one

```python
ss = [0]
for i in range(1, 500):
    ss.append(ss[i-1] + i)
```

Compare the last element in the list `ss` with `s`. What is the value of `i` after the execution of the `for` loop?
Task 8

Again, like in Task 6, set up a list with the name `xplot` which contains 100 equidistant values between 0 and 1. But this time, use a `for`-loop with a counter instead of a list comprehension. Pay attention to the `append`-method for lists. You might need to generate an empty list first.

Task 9

Set up a list `yplot` which contains the values `arctan(x)` for all the `x` in `xplot`.

Task 10

Make a plot of `yplot` versus `xplot`. Use for this the command `plot`. Depending on your environment, you might also need to run `show()` to show the plot.

Task 11

Compute

\[ \sum_{i=1}^{200} \frac{1}{\sqrt{i}}. \]

Extra Task 12 (in case you want an extra challenge)

Now we use lists for recursions. Consider the recursion formula:

\[ u_{n+3} = u_{n+2} + ha \left( \frac{23}{12} u_{n+2} - \frac{4}{3} u_{n+1} + \frac{5}{12} u_n \right) \]

with \( n = 0, \ldots, 1000 \), \( h = 1/1000 \) and \( a = -0.5 \).

1. Create a list `u`. Store in its first three elements \( e^0 \), \( e^{ha} \) and \( e^{2ha} \). These represent \( u_0 \), \( u_1 \), and \( u_2 \) in the above formula. Build up the complete list from the recursion formula.

2. Construct a second list `td` in which you store the values \( nh \), with \( n = 0, \ldots, 1000 \). Plot `td` versus `u`. Make a second plot in which you plot the difference, i.e. \( |e^{tn} - u_n| \), where \( t_n \) represents the values inside the vector `td`.

The `abs` function computes the absolute value of a number. Set axis labels and a title (investigate the commands `xlabel`, `ylabel` and `title`).

Plot both the approximation and the exact solution in the same figure and compare the result. You might want to plot the difference between the approximation and the exact solution instead. The mathematical background to this problem will be taught in later courses when you became familiar with differential equations.

Good luck!