PHOT 110: Introduction to programming

Midterm exam questions

Michaël Barbier, Spring semester (2023-2024)

Before you start

The midterm exam counts for 10% of your total grade of the course. The exam is performed on the computer, and you do not need to provide any answers on paper. There is one question (the third question) which asks to correct a script which contains various errors, other questions require you to write Python scripts.

Take the following point into account before you start the exam.

- Your exam is placed as an assignment on MS-Teams. Download all files attached before you start:
  - The questions: `midterm_questions_x.pdf`
  - The script with errors for question 3: `script_with_errors.py`
  - Three cheat-sheets: for Python, Numpy, and Matplotlib.
- Make sure that the `Numpy` and `Matplotlib` libraries are installed already (we will test this together before the exam starts).
- Please make sure that you save all the scripts and output files of the exam, so you can upload them at the end of the exam.
- Let us know if during the exam there is any issue with the computer, PyCharm, or libraries. We will try to verify this up front, but please inform us in case of any issues.

You will be asked to save plots to your folder, this can be done using the `savefig` method, see the following example (the output of this example is a simple line plot):

```
import matplotlib.pyplot as plt

fig, ax = plt.subplots()
x = [1, 3]
y = [2, 5]
ax.plot(x, y)
fig.savefig("output_plot_example.png")
```

Questions

Please solve all the following questions.

Question 1: Print the third powers

Write a script that prints the first N positive integer numbers to the third power, that is, for a given N (you can take N as a parameter of your script) prints the numbers:

$$1^3,\ 2^3,\ 3^3,\ 4^3,\ \ldots\ , N^3$$

where each number is printed on a separate line. If you set `N = 5`, then the output should look similar to:

```
1
8
27
64
125
```

Save your solution as a script with file name: `solution_1.py`.

Question 2: Plot the error function

Implement a script that plots the error function:

$$\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x \exp(-t^2)\,\mathrm{d}t$$

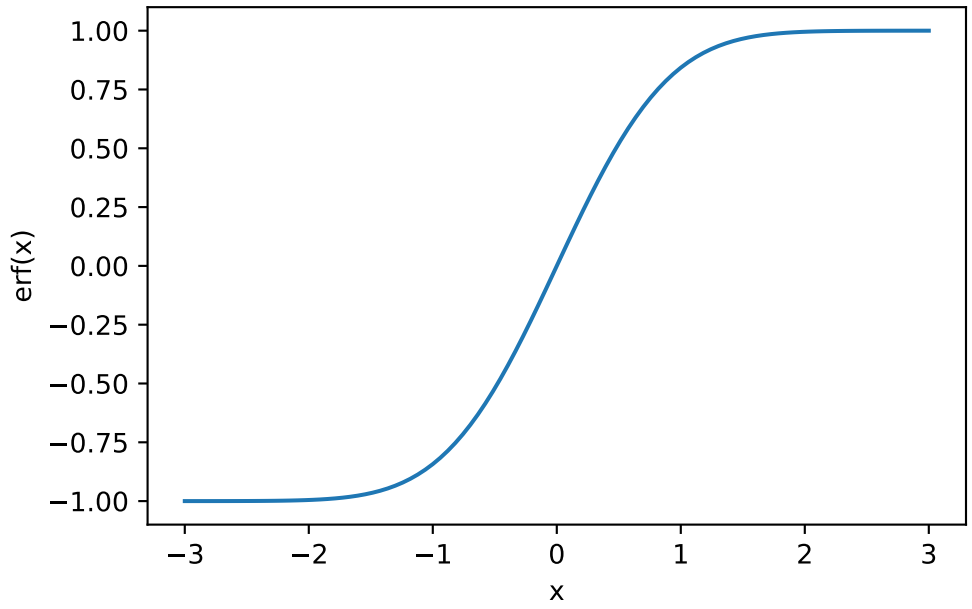
You can make use of the error function in the `math` library: `math.erf()` which takes one argument (the x-value in the above formula) and can be used as in the following example:

```
import math

# This example prints the error function for x = 0.8
y = math.erf(0.8)
print(y)
```

```
0.7421009647076605
```

Calculate the error function for multiple x-values in interval $[-3, 3]$, and make a line plot (take a sufficiently high number of x values so the curve looks smooth). **Save the plot** under the file name: `output_plot_math_erf.png`. Save your solution as a script with file name: `solution_2.py`.



Question 3: Correct a Python script

Open the script with name: `script_with_errors.py` and correct the errors.

The corrected script prompts the user to type a sentence. It then reverses the order of the words, prints this reversed sentence on the console and saves it to a text-file.

For example when you would enter: `This is my sentence.` as input, then the script will print the following output to the console:

```
sentence. my is This
```

It also saves the same sentence to a text file with file name: `output_text_script_with_errors.txt`

Question 4: User input and error handling

Prompt the user to input a floating point number between 0 and 20. print the integer part and the rest part. Print also whether the number is larger than or equal to 10. Extend the

program by catching any errors (using the `try` and `except` keywords) when the user types a number which is too small, too large, or types invalid input. Allow the user to try again in that case.

Example:

```
Give a decimal number in interval [0, 20]: 25
The number 25 is invalid, please try again.

Give a decimal number in interval [0, 20]: twenty
The number twenty is invalid, please try again.

Give a decimal number in interval [0, 20]: 9.4
The number you provided has:
  integer part: 9
  decimal part: 0.4
The number is smaller than 10
```

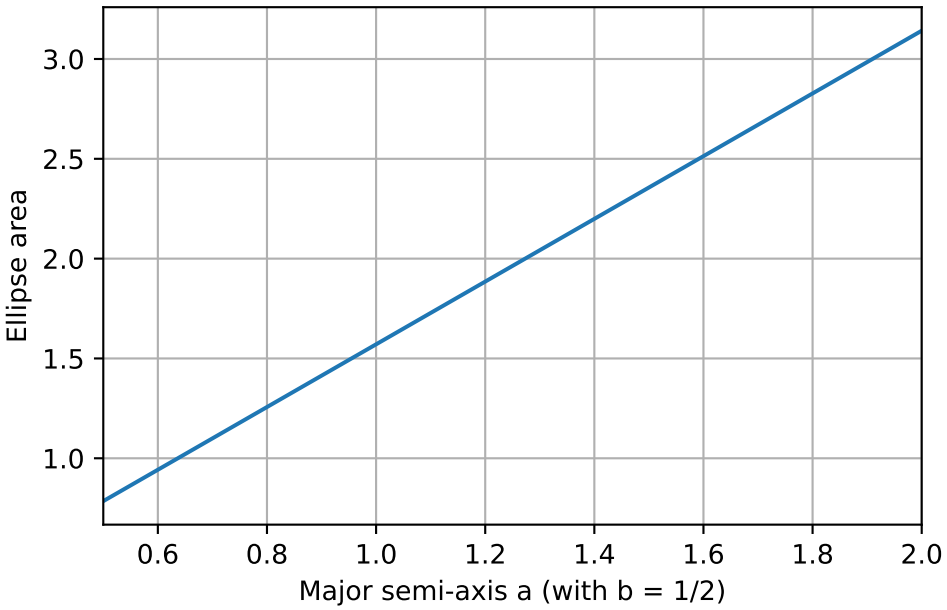
Save your solution as a script with file name: `solution_4.py`.

Question 5: Creating and using modules

Create a **module** (a separate script file) with file name `module_ellipse.py` which contains two functions: `perimeter_ellipse(a, b)` and `area_ellipse(a, b)` which calculate the perimeter and the area of an ellipse. The parameters `a` and `b` are the major and minor semi-axis. Use the following formulas to calculate the (approximate) perimeter P and the area A of the ellipse:

$$A = \pi a b$$
$$P = \pi \left[3\,(a+b) - \sqrt{(3a+b)(a+3b)} \right]$$

Afterwards, import and use this module in a script (with file name: `solution_5.py`) to plot the **area** as function of a where you fix the minor semi-axis $b = 1/2$. Take the interval for parameter $a \in [0.5, 2]$. Use a line plot with enough points to get a smooth curve. Let the script **save the plot** to a file named `output_plot_ellipse.png`. The saved file should contain a graph similar to the one below:


3.0
2.5
2.0
1.5
1.0
Ellipse area
0.6
0.8
1.0
1.2
1.4
1.6
1.8
2.0
Major semi-axis a (with b = 1/2)