PHOT 110: Introduction to programming

Midterm exam (retake) questions, version A

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.

- For those who use their own computer/laptop: 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.
- For those who work on the desktop computers in the lab: all the required files should be found on the Desktop.
- Make sure that the `Numpy` and `Matplotlib` libraries are installed already (we will test this together before the exam starts).
- At the end of the exam we will go around with USB drives to collect the exams: put your solutions (and output files) in a folder which has both your name and your student number in the folder name, for example "Michael_Barbier_30029034".
- 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 out parts of a word

Write a script that prints parts of a word by growing it, starting from the first letter and growing a letter each time. You can make use of the slicing operator, for example, to get the first two characters of a string you could use:

```
# Example of how to use the slicing operator
word = "Apple"
print(word[0:2])
```

```
Ap
```

Your solution script should print each sub-string on a separate line. For example: if you use `word = "Hello"` as word, then the output should look similar to:

```
H
He
Hel
Hell
Hello
```

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

Question 2: Verify email address

Make the script to verify an email address. Prompt the user to input an email address. Automatically verify whether the format of the student email address ends with `@std.iyte.edu.tr`. Allow the user to try again if his/her input was invalid until the user fills in a valid email address. You can use the `endswith()` method to verify whether the email address is valid:

```
# Example showing how to use the "endswith()" method
my_string = "hello.txt"

# The endswith() method returns True if the string ends in a
# specific substring (here "txt"), otherwise it returns False
print(my_string.endswith("txt"))
print(my_string.endswith("png"))
```

```
True
False
```

This is example output of a correct script:

```
Please enter a valid IYTE student email address: harry@hotmail.com
The provided email address is invalid, please try again.

Please enter a valid IYTE student email address: harry at std.iyte.edu.tr
The provided email address is invalid, please try again.

Please enter a valid IYTE student email address: harry@std.iyte.edu.tr
Thank you, your contact information will be updated.
```

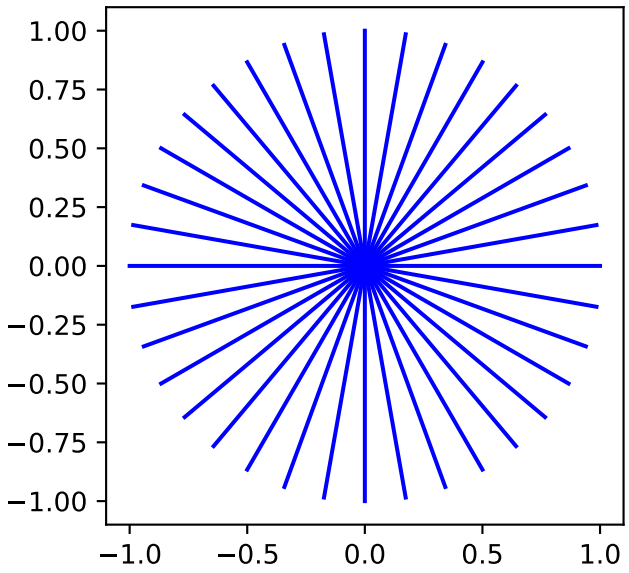
Save your solution as a script with file name: `solution_2.py`.

Question 3: Correct a Python script

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

The corrected script plots a graph where lines are plotted outwards from the origin. There is a line every 10 degrees. This graph is then saved as a png-file with file name `output_script_with_errors.png` to the hard disk.

The output graph should look as the plot below:



Question 4: Creating and using modules

Create a **module** (a separate script file) with file name `module_conversion.py` which provides the conversion between units of Watt (W) and Horsepower (hp) contains two functions:

- `value_hp = watt_to_horsepower(value_watt)`: which converts values in kiloWatt to horsepower.
- `value_watt = horsepower_to_watt(value_hp)`: which converts values in Horsepower to values in unit of kiloWatt.

For the definition of the conversion assume that kiloWatt (kW) and Horsepower (hp) are related as follows:

$$1 \text{ kiloWatt} = 1.34 \text{ hp}$$

Afterwards, import and use this module in a script (with file name: `solution_4.py`) that prompts a user to enter his/her car's power in horsepower and then converts it to kiloWatt, and prints the result. See the following example output of a correct script:

```
Please enter the power of your car (in hp): 105
The power of your car in kiloWatt: 78.4 kW
```

Question 5: Plot Gaussian probability density functions

Consider the Gaussian probability density function $g(x)$, with parameters mean μ and standard deviation σ, defined as:

$$g(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

Plot this Gaussian function for two different parameter sets of mean and standard deviation, where: ($\mu_1 = 0$, $\sigma_1 = 0.5$) and ($\mu_2 = 2$, $\sigma_2 = 1$). Plot both these probability density functions as a smooth curve. Then add a curve representing the sum of the two Gaussian functions. Plot both the separate functions and the summed function using a line plot.

For each of the curves you can use the same x-values within an interval $[-3, 5]$ (take a sufficiently high number of x values to obtain smooth curves). **Save the plot** under the file name: `output_plot_gauss.png`. Save your solution as a script with file name: `solution_5.py`.