HWS 2024
Moritz Grävinghoff

STADS | Students' Association for Data Analytics & Statistics

Sample Certificate Exam

"Python Programming Course"

Important Notes:

- You have **45 minutes** to complete the exam. Additionally, you have an extra **10 minutes** before the exam to download the tasks and **10 minutes** afterward to upload your solutions. The exam tasks will be sent to your email address at 18:50. Please note:

  The submission must be made no later than 19:55 to `pythonkurs@stads.de`.

- The exam consists of 3 tasks.
- Please create **one Python file** for **all tasks combined**, named

  `<LastName>_<FirstName>_exam.py` (e.g., `mustermann_max_exam.py`)

  and complete the exam only in this file. Then, send your solutions as a **ZIP file** named

  `<LastName>_<FirstName>_exam.zip` (e.g., `mustermann_max_exam.zip`)

  to the above email address.
- The code must run on Python ≥3.11.5 with pandas ≥1.5.3, matplotlib ≥3.7.1, or plotly ≥5.15.0. If other packages or versions are used, the respective versions must be specified.

Good Luck!

Task 1 (15 Points)

a) [2 Points] Create a variable called `favorite_color` and assign your favorite color to the variable. Print the text "My favorite color is `favorite_color`".

b) [6 Points] Create a dictionary called `result_b`, in which for each key "i" to "vi," a tuple is given as the value. Each tuple should have the data type of the expression as the first element and the value of the expression as the second element. Print the dictionary.

(i) `7 % 4`

(ii) `"Python"[1:5]`

(iii) `bool(None)`

(iv) `{2, 4, 6, 8} - {4, 8}`

(v) `[z**3 for z in range(5)]`

(vi) `len("Welt") < 5`

c) [3 Points] Create a list called `fibonacci_numbers` with the first 10 Fibonacci numbers. Print the fourth and second-to-last elements of the list.

d) [4 Points] You are given the following code, which generates a list of 300 random numbers in the range of 10 to 500.

```
import random
random.seed(42)
random_numbers = [random.randint(100, 500) for _ in range(300)]
```

Print the count of numbers from `random_numbers` that are divisible by 9.

Task 2 (15 Points)

a) [7 Points] Write a function called `carbon_footprint_category`, which takes the annual CO2 emissions (in tons) of several individuals in a list and returns an assessment of their carbon footprint category.
Categories:

- "Low": CO2 emissions < 5 tons/year
- "Average": 5 tons/year $\leq$ CO2 emissions ≤ 15 tons/year
- "High": CO2 emissions > 15 tons/year

The return value should be a dictionary where the CO2 value is the key and the category is the value.
Call the function `carbon_footprint_category` with the list `[3, 20, 12, 16, 7]` and print the result.

b) [8 Points] Create

- a class called `BookCollection` with no attributes,
- an empty constructor that initializes only `self`,
- a method called `add_book`, which takes two strings (book title and author) and returns a string in the format "Title by Author,"
- a method called `list_collection`, which takes a list of tuples (title, author) and returns a formatted string with all book titles in the collection. The method should internally use the `add_book` method.

Call the method `list_collection` with the list

```
[("The Hobbit", "J.R.R. Tolkien"),
("1984", "George Orwell"),
("The Metamorphosis", "Franz Kafka")]
```

and print the result.

Task 3 (15 Points)

You are given the following code, which imports the dataset to be processed.

```
import pandas as pd
import numpy as np
np.random.seed(42)
df = pd.DataFrame({
    'sensor_id': range(1, 11),
    'measurement': np.random.uniform(50, 100, 10),
    'calibration': np.random.uniform(0.8, 1.2, 10),
    'error_rate': np.random.uniform(0.01, 0.1, 10),
    'temperature': np.random.uniform(20, 30, 10)
})
```

The DataFrame contains five columns:

- sensor_id $\mapsto$ Unique ID of the sensor
- measurement $\mapsto$ Measured value of the sensor
- calibration $\mapsto$ Calibration factor of the sensor
- error_rate $\mapsto$ Error rate of the sensor in percent
- temperature $\mapsto$ Temperature during the measurement

a) [3 Points] Print the average measurement value and the highest error rate.

b) [4 Points] Create a new column called `adjusted_value`, which calculates the adjusted measurement (measurement $*$ calibration). Print the last five rows of the DataFrame.

c) [3 Points] Print all sensors where the calibration factor is greater than 1.0.

d) [5 Points] Create a scatter plot that shows the relationship between adjusted measurement, error rate, and temperature. Use the adjusted measurement as the x-axis, the error rate as the y-axis, and the temperature for the coloring of points. The title of the plot should be "Relationship between Adjusted Measurement, Error Rate, and Temperature." Label the x-axis as "Adjusted Measurement" and the y-axis as "Error Rate (%)." Display the plot.