Python Practice 13

Imagine you have a question about Python code or statistics and ask a large language model (LLM), like ChatGPT, for help. The model responds to your prompt by producing a code example. However, you are quite experienced with LLMs and know that these models can sometimes *hallucinate*, meaning they produce an answer that seems meaningful but is actually incorrect. You will be asked to check the LLM's answers and correct any mistakes. Note that some answers may actually be correct, so you only need to correct incorrect answers and indicate correct answers.

1 Sum of two arrays

Prompt: "How to sum two data arrays in python element-wise?"

LLM Response:

```
data_1 = [1,2,3]
data_2 = [2,3,4]
data_sum = data_1 + data_2
```

2 Random data generation

Prompt: "I want to generate a random set of 1000 points samples from a standard normal distribution. How can I do that?"

LLM Response:

```
sample = np.random.normal(loc=0.0, scale=1.0, size=1000)
```

3 Sample mean

Prompt: "I have 10 samples drawn from a target distribution with 100 points in each. How can I compute the mean of each sample?"

LLM Response:

```
data = np.array([...]) # A data sample with a shape of (10, 100)
means = np.mean(data)
```

4 Masking the data

Prompt: "I have an impression that my array has an outlier. I want to check if there are any values in the array that are less that 3.5"

LLM Response:

```
data = np.array([...]) # A data sample to test
threshold = 3.5
outliers = data[data < threshold]
```

5 Data reshaping

Prompt: "I have an array of 10000 points that contains the experimental data of 10 measurement rounds with 1000 points in each. How can I reshape the array, so it contains 10 rows with 1000 elements (columns) in each?"

LLM Response:

```
data = np.array([...]) # A data sample with 10000 points
reshaped_data = np.reshape(data, (1000, 10))
```

6 Unbiased estimator

Prompt: "How can I compute the unbiased estimator of the standard deviation?"

LLM Response:

```
data = np.array([1,2,3])
s = np.std(data)
```

7 Z-score

Prompt: "I have a set of 25 measurements from my thickness measuring device. It has an error of 10 μm. I want to test if the thickness of my sample is larger than 100 μm with a 95 % confidence."

LLM Response:

```
measurements = np.array([101.3, 102.0, ]) # 25 measurements of the sample thinkness
threshold = 100 # target thickness in micrometers
sigma = 10 # a standart deviation of the measuring device
mean = np.mean(measurements)
z_score = (mean - threshold) / sigma

z_threshold = 1.65 # A 95% confidence interval from Z-tables

if z_score > z_threshold:
    print("Sample is thicker than 100 micrometer")
else:
    print("Sample is not thicker than 100 micrometer")
```

8 SSE calculation in ANOVA

Prompt: "I have an array with 3 data samples with 10 points in each. These numbers represent the production quality score of 10 test examples of the same product made by 3 different partner manufacturers. Before passing the contract to the sales department I need to figure out, which one of these three manufacturers I should keep partnering with. First, I want to find if there is any statistically meaningful difference between the manufacturers. I did my statistics course well, and I remember that there is the ANOVA test that can help me with that. First, how can compute the errors sum of squares (SSE)?"

LLM Response:

```
scores = np.array([...]) # the array of the shape (3,10) with the scores
mean_scores = np.mean(scores, axis=1) # mean values for each manufacturer
sse = np.sum((scores - mean_scores.reshape(3, 1))**2)
```

9 SSB calculation in ANOVA (continuation)

Prompt: "Okay, good, and how can I compute the sum of squares between the samples (SSB)?"

LLM Response:

```
scores = np.array([...]) # the array of the shape (3,10) with the scores
mean_scores = np.mean(scores, axis=1) # mean values for each manufacturer
total_mean_score = np.mean(mean_scores) # The total mean
ssb = np.sum((mean_scores - total_mean_score)**2)
```

10 F calculation in ANOVA (continuation)

Prompt: "Good, now how can I compute if there is any statistically meaningful difference at the level of confidence of 95 %?"

LLM Response:

```
num_samples = 3
num_points_per_sample = 10
dfb = num_samples - 1 # num. DOF between the samples
dfe = num_samples * (num_points_per_sample - 1) # num. DOF within the samples
msb = ssb / dfb
mse = sse / dfe
F = msb / mse
F_threshold = 3.35 # an approximate value of the F(2,27) at 95% confidence

if F > F_threshold:
    print("There is a difference between manufacturers!")
else:
    print("There is no difference between manufacturers")
```