White Exam: Corrections

Master 222: Introduction to Python for Finance

September 24, 2024

Examination Instructions

- **Duration**: This exam is 2 hours long.
- **No Devices**: Use of computers, smartphones, or any internet-enabled devices is prohibited.
- **Code Language**: All code must be written in Python.
- **Indentation**: Proper indentation is crucial. It counts towards your grade.
- **Package Imports**: Any packages used should be imported at the start of each question.
- **Correction Indulgence**: Since this is a written code exam, some leniency will be given during correction.

Please read questions carefully and do your best. Good luck!

1 Basics and Fundamentals (6 points)

Problem 1. [1 point] Write a function called `occurrence` to count the occurrence of each vowel in a string.

Consider both uppercase and lowercase. It's advisable to use the 'in' keyword, as demonstrated below:

```python
my_list = [1, 2, 3, 4, 5]
if 3 in my_list:
    print("3 is in the list!")
```

Listing 1: Using 'in' keyword

Correction:

```python
def occurrence(s):
    vowels = "AEIOUaeiou"
    vowel_count = {vowel: 0 for vowel in vowels}

    for char in s:
        if char in vowels:
            vowel_count[char] += 1
    return vowel_count
```

Problem 2. [1 point] Create a list called `squared_numbers` containing the squares of numbers from 1 to 15.

Correction:

```python
squared_numbers = [i**2 for i in range(1, 16)]
```

Problem 3. [2 points] Create a function named `is_prime` to verify if a number is prime. Then, generate a list of prime numbers between 1 and 50.

A prime number is divisible only by 1 and itself. It's recommended to use Python's integer division rest `%`.

Correction:

```python
def is_prime(n):
    if n <= 1:
        return False
    for i in range(2, n):
        if n % i == 0:
            return False
    return True

def generate_primes(limit):
    return [num for num in range(1, limit+1) if is_prime(num)
   ]

prime_numbers = generate_primes(50)
```

Problem 4. [2 points] Develop a function named `string_alternate` which, given a string, produces a new string containing every alternate character.

For the input 'PythonExam', the function should return 'PtoEa'.

Correction:

```
def string_alternate(s):
    result = ""
    for i in range(len(s)):
        if i % 2 == 0:
            result += s[i]
    return result
```

2 Numpy and Intermediate Data Analysis (6 points)

Problem 5. [1 point] Create a 4×4 numpy matrix `M` with numbers spanning from 1 to 16. Subsequently, print the matrix and its diagonal.

Correction:

```
import numpy as np
M = np.array([
    [1, 2, 3, 4],
    [5, 6, 7, 8],
    [9, 10, 11, 12],
    [13, 14, 15, 16]
])
print("Matrix M:")
print(M)
# Print the diagonal of the matrix
print("\nDiagonal of M:")
print(np.diag(M))
```

Problem 6. [1 point] Construct two random 3×3 matrices `X` and `Y`. Then, print the result of their matrix multiplication. **Correction:**

```
import numpy as np
X = np.random.rand(3,3)
Y = np.random.rand(3,3)
print(np.dot(X, Y))
```

Problem 7. [2 points] Generate a numpy array `D` containing 200 random numbers between -1 and 1. Subsequently, compute and display its variance.

Consider using the ***`np.var()`*** *function from numpy to compute the variance.*

Correction:

```
import numpy as np
D = np.random.uniform(-1, 1, 200)
print(np.var(D))
```

Problem 8. [1 point] Identify and print the indices in array `D` where values exceed 0.5.

Correction:

```
print([i for i in range(200) if D[i] > 0.5])
```

Problem 9. [1 point] Adjust the values in array `D` to be rounded to two decimal places.

Hint: Refer to the official ***`numpy`*** *documentation excerpt below.*

```
numpy.round
------------
numpy.round(a, decimals=0, out=None)
    Evenly round to the given number of decimals.

    Parameters:
    - a (array_like): Input data.
    - decimals (int, optional): Number of decimal places to round to (default: 0).
      Negative values specify positions to the left of the decimal point.

    Returns:
    - ndarray: An array containing the rounded values.
```

Correction:

```
print(np.round(D, 2))
```

3 Pandas, Matplotlib, and yfinance (8 points)

Problem 10. [2 points] Create a DataFrame named `df` using the dictionary data provided below:

```
data = {
    'Name': ['Alice', 'Bob', 'Charlie', 'David'],
    'Age': [25, 30, 35, 40],
    'Salary': [50000, 55000, 60000, 65000]
}
```

Subsequently, compute and display the average age and salary from `df`.

Hint: To compute the average of a column in a DataFrame, consider using the ***`mean()`*** *method. For instance,* ***`df['ColumnName'].mean()`*** *would give the average of the specified column.*

Correction:

```
import pandas as pd
df = pd.DataFrame({
    'Name': ['Alice', 'Bob', 'Charlie', 'David'],
    'Age': [25, 30, 35, 40],
    'Salary': [50000, 55000, 60000, 65000]
})
print(df['Age'].mean())
print(df['Salary'].mean())
```

Problem 11. [2 points] Utilizing `matplotlib`, construct a bar graph representing the salary of each individual from the DataFrame `df`. Ensure the following elements are incorporated:

- A title for the graph.
- Label for the x-axis.
- Label for the y-axis.
- Names of the individuals displayed on the x-axis.

Hint: To create a bar graph using ***`matplotlib`****, you can use the* ***`bar()`*** *function from the* ***`pyplot`*** *module.*

Correction:

```
import matplotlib.pyplot as plt
plt.bar(df['Name'], df['Salary'])
plt.xlabel('Name')
plt.ylabel('Salary')
plt.title('Salary for each person')
```

Problem 12. [2 points] Using `yfinance` library, obtain the closing prices of the "AAPL" stock ticker for the recent 30 days. Store the retrieved data in a DataFrame.

Correction:

```
import yfinance as yf
data = yf.download("AAPL", period="1mo")
closing_prices = data['Close']
```

Problem 13. [2 points] Using `matplotlib`, plot the 30-day closing prices of "AAPL". Ensure the graph incorporates the following details:

- x-axis labeled as "Date".
- y-axis labeled as "Closing Price".
- A title: "AAPL 30-day Closing Price".

Correction:

```
import matplotlib.pyplot as plt

plt.plot(closing_prices)
plt.xlabel('Date')
plt.ylabel('Closing Price')
plt.title('AAPL 30-day Closing Price'
```