6.4-Assignment

R.Nikhil Kumar

2303A52260

Batch – 44

Task 1: Student Performance Evaluation System

Scenario :

You are building a simple academic management module for a university system where student performance needs to be evaluated automatically.

Task Description :

Create the skeleton of a Python class named Student with the attributes:

- ***name***
- ***roll_number***
- ***marks***

Code :

```
class Student:
    def __init__(self, name, roll_number, marks):
        self.name = name
        self.roll_number = roll_number
        self.marks = marks

    def display_details(self):
        print(f"Name: {self.name}")
        print(f"Roll Number: {self.roll_number}")
        print(f"Marks: {self.marks}")

    def check_performance(self, class_average):
        if self.marks > class_average:
            return f"{self.name} has performed above the class average."
        else:
            return f"{self.name} has performed below the class average."
student1 = Student("Nikhil", "2260", 85)
student2 = Student("Induvardhan", "2030", 75)
class_average = 80
student1.display_details()
print(student1.check_performance(class_average))
student2.display_details()
print(student2.check_performance(class_average))
```

Output :

```
Name: Nikhil
Roll Number: 2260
Marks: 85
Nikhil has performed above the class average.
Name: Induvardhan
Roll Number: 2030
Marks: 75
Induvardhan has performed below the class average.
```

Task 2: Data Processing in a Monitoring System

Scenario

You are working on a basic data monitoring script where sensor readings are collected as numbers. Only even readings need further processing.

Task Description

Write the initial part of a for loop to iterate over a list of integers representing sensor readings.

Code :

```python
sensor_readings = [10, 15, 22, 33, 40, 55, 60]
for reading in sensor_readings:
    if reading % 2 == 0:
        squared_value = reading ** 2
        print(f"The square of the even reading {reading} is {squared_value}.")
```

Output :

```
The square of the even reading 10 is 100.
The square of the even reading 22 is 484.
The square of the even reading 40 is 1600.
The square of the even reading 60 is 3600.
```

Task 3: Banking Transaction Simulation

Scenario :

You are developing a basic banking module that handles deposits and withdrawals for customers.

Task Description :

Create the structure of a Python class named BankAccount with attributes:

- ***account_holder***
- ***balance***

Code :

```
class BankAccount:
    def __init__(self, account_holder, balance=0):
        self.account_holder = account_holder
        self.balance = balance

    def deposit(self, amount):
        """Method to deposit money into the account."""
        if amount > 0:
            self.balance += amount
            print(f"Deposited {amount}. New balance is {self.balance}.")
        else:
            print("Deposit amount must be positive.")

    def withdraw(self, amount):
        """Method to withdraw money from the account."""
        if amount > self.balance:
            print("Insufficient balance for this withdrawal.")
        elif amount > 0:
            self.balance -= amount
            print(f"Withdrew {amount}. New balance is {self.balance}.")
        else:
            print("Withdrawal amount must be positive.")
account = BankAccount("Alice", 100)
account.deposit(50)
account.withdraw(30)
account.withdraw(150)
```

Output :

```
Deposited 50. New balance is 150.
Withdrew 30. New balance is 120.
Insufficient balance for this withdrawal.
```

Task 4: Student Scholarship Eligibility Check

Scenario :

A university wants to identify students eligible for a merit-based scholarship based on their scores.

Task Description :

Define a list of dictionaries where each dictionary represents a student with:

- ***name***
- ***score***

Write the initialization and list structure yourself.

Then, prompt GitHub Copilot to generate a while loop that:

- ***Iterates through the list***
- ***Prints the names of students who scored more than 75***

Code :

```
students = [
    {"name": "Nikhil", "score": 82},
    {"name": "Induvardhan", "score": 74},
    {"name": "Sourish", "score": 91},
    {"name": "Shiva Teja", "score": 68},
    {"name": "Abhinav", "score": 77}
]
index = 0
while index < len(students):
    if students[index]["score"] > 75:
        print(f"{students[index]['name']} is eligible for the scholarship.")
    index += 1
```

Output :

```
Nikhil is eligible for the scholarship.
Sourish is eligible for the scholarship.
Abhinav is eligible for the scholarship.
```

Task 5: Online Shopping Cart Module

Scenario :

You are designing a simplified shopping cart system for an e-commerce website that supports item management and discount calculation.

Task Description :

Begin writing a Python class named ShoppingCart with:

• An empty list to store items (each item may include name, price,

quantity)

Code :

```python
class ShoppingCart:
    def __init__(self):
        self.items = []

    def add_item(self, name, price, quantity):
        item = {"name": name, "price": price, "quantity": quantity}
        self.items.append(item)
        print(f"Added {quantity} of {name} at ${price} each to the cart.")

    def remove_item(self, name):
        for item in self.items:
            if item["name"] == name:
                self.items.remove(item)
                print(f"Removed {name} from the cart.")
                return
        print(f"Item {name} not found in the cart.")

    def calculate_total(self):
        total = 0
        for item in self.items:
            total += item["price"] * item["quantity"]
        return total

    def apply_discount(self, total):
        discount_threshold = 100
        discount_rate = 0.1
        if total > discount_threshold:
            discount_amount = total * discount_rate
            total -= discount_amount
            print(f"Applied a discount of ${discount_amount:.2f}.")
        return total
cart = ShoppingCart()
cart.add_item("Laptop", 999.99, 1)
cart.add_item("Mouse", 25.50, 2)
cart.add_item("Keyboard", 45.00, 1)
total = cart.calculate_total()
print(f"Total before discount: ${total:.2f}")
final_total = cart.apply_discount(total)
print(f"Final total after discount: ${final_total:.2f}")
```

Output :

```
Added 1 of Laptop at $999.99 each to the cart.
Added 2 of Mouse at $25.5 each to the cart.
Added 1 of Keyboard at $45.0 each to the cart.
Total before discount: $1095.99
Applied a discount of $109.60.
Final total after discount: $986.39
```