ASSIGNMENT-6.4

NAME:-G.VARSHITHA

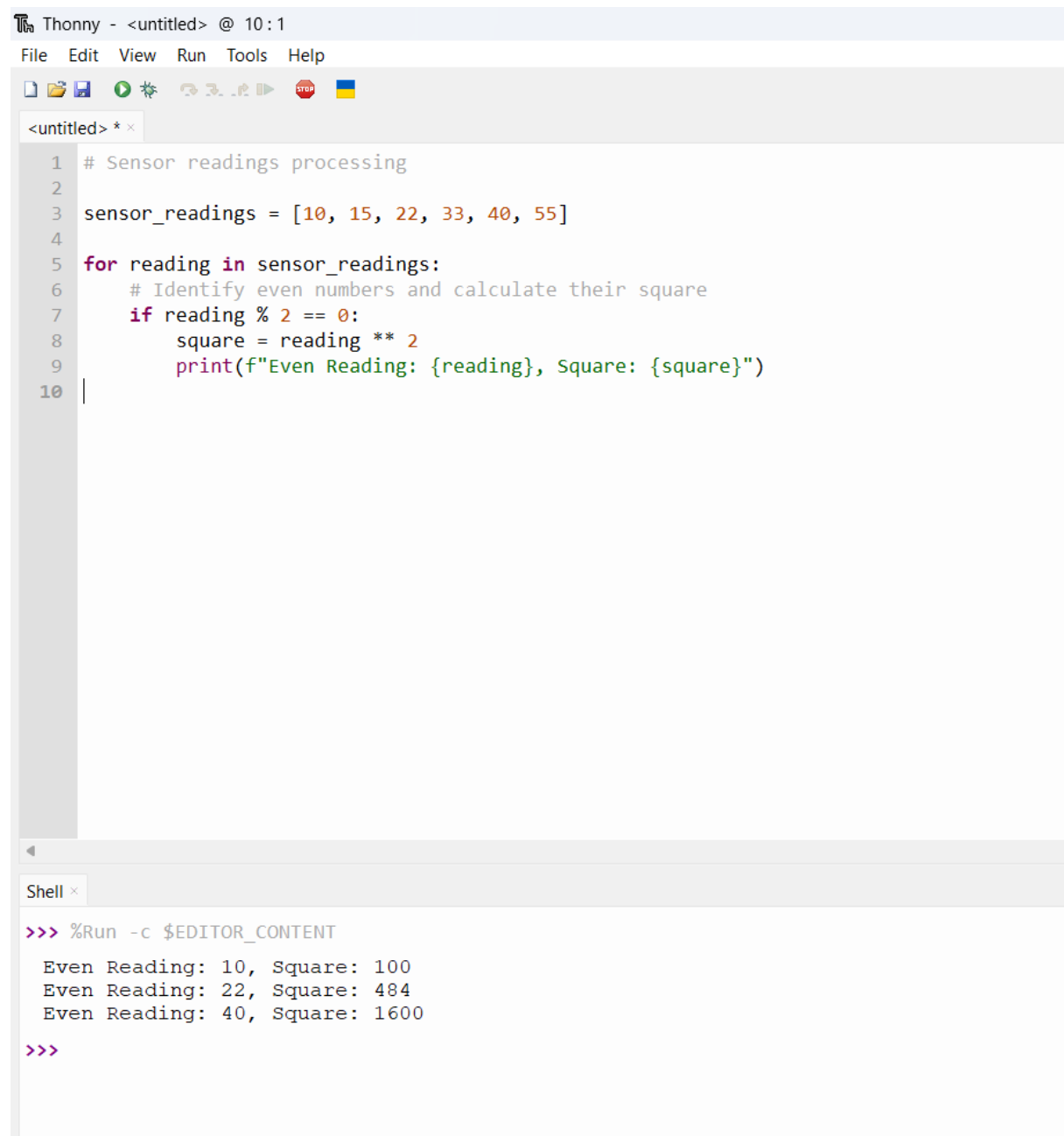BATCH-16

HTNO:-2303A51053

Task 1:StudentPerformanceEvaluationSystem

File   Edit   View   Run   Tools   Help

<untitled> * ×

```python
# Student class for performance evaluation

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

    # Display student details
    def display_details(self):
        print("Name:", self.name)
        print("Roll Number:", self.roll_number)
        print("Marks:", self.marks)

    # Check performance based on class average
    def check_performance(self, class_average):
        if self.marks > class_average:
            return "Student is performing above average"
        else:
            return "Student is performing below average"


# Sample usage
student1 = Student("Rohan", "23CS101", 94)
student1.display_details()
print(student1.check_performance(75))
```

Shell ×

```
>>> %Run -c $EDITOR_CONTENT
 Name: Rohan
 Roll Number: 23CS101
 Marks: 94
 Student is performing above average
>>>
```

Task2:DataProcessinginaMonitoringSystem

<untitled> * ×

```python
1  # Sensor readings processing
2
3  sensor_readings = [10, 15, 22, 33, 40, 55]
4
5  for reading in sensor_readings:
6      # Identify even numbers and calculate their square
7      if reading % 2 == 0:
8          square = reading ** 2
9          print(f"Even Reading: {reading}, Square: {square}")
10 |
```

Shell ×

```
>>> %Run -c $EDITOR_CONTENT
 Even Reading: 10, Square: 100
 Even Reading: 22, Square: 484
 Even Reading: 40, Square: 1600
>>>
```

Task3: BankingTransaction Simulation

File   Edit   View   Run   Tools   Help

<untitled> * ×

```python
 1  # BankAccount class for deposit and withdrawal
 2
 3  class BankAccount:
 4      def __init__(self, account_holder, balance):
 5          self.account_holder = account_holder
 6          self.balance = balance
 7
 8      # Deposit money
 9      def deposit(self, amount):
10          self.balance += amount
11          print(f"Deposited {amount}. New Balance: {self.balance}")
12
13      # Withdraw money with balance check
14      def withdraw(self, amount):
15          if amount <= self.balance:
16              self.balance -= amount
17              print(f"Withdrawn {amount}. Remaining Balance: {self.balance}")
18          else:
19              print("Insufficient balance. Withdrawal not allowed.")
20
21
22  # Sample usage
23  account = BankAccount("Rohan", 5000)
24  account.deposit(2000)
25  account.withdraw(3000)
26  account.withdraw(5000)
27
```

Shell ×

```
>>> %Run -c $EDITOR_CONTENT
 Deposited 2000. New Balance: 7000
 Withdrawn 3000. Remaining Balance: 4000
 Insufficient balance. Withdrawal not allowed.
>>>
```

Task4:StudentScholarshipEligibilityCheck

```
# List of students with scores

students = [
    {"name": "Amit", "score": 80},
    {"name": "Sneha", "score": 72},
    {"name": "Kiran", "score": 90},
    {"name": "Priya", "score": 65}
]

index = 0

# While loop to check scholarship eligibility
while index < len(students):
    if students[index]["score"] > 75:
        print(students[index]["name"], "is eligible for scholarship")
    index += 1

```

Shell

```
>>> %Run -c $EDITOR_CONTENT
 Amit is eligible for scholarship
 Kiran is eligible for scholarship
>>>
```

Task5:OnlineShoppingCartModule

<untitled> * ×

```
10            print(f"{name} added to cart")
11
12        # Remove item from cart
13        def remove_item(self, name):
14            self.items = [item for item in self.items if item["name"] != name]
15            print(f"{name} removed from cart")
16
17        # Calculate total bill with discount
18        def calculate_total(self):
19            total = 0
20            for item in self.items:
21                total += item["price"] * item["quantity"]
22
23            if total > 2000:
24                total *= 0.9   # 10% discount
25
26            return total
27
28
29    # Sample usage
30    cart = ShoppingCart()
31    cart.add_item("Laptop Bag", 1500, 1)
32    cart.add_item("Mouse", 500, 2)
33    cart.remove_item("Mouse")
34
35    print("Total Bill:", cart.calculate_total())
36
```

Shell ×

```
>>> %Run -c $EDITOR_CONTENT
 Laptop Bag added to cart
 Mouse added to cart
 Mouse removed from cart
 Total Bill: 1500
>>>
```