

ASSIGNMENT-6.4

NAME:-N.ROHAN

BATCH-16

HTNO:-2303A51064

Task 1: Student Performance Evaluation System

The screenshot shows the Thonny Python IDE interface. The top bar displays the title "Thonny - <untitled> @ 26:36". Below the title is a menu bar with File, Edit, View, Run, Tools, and Help. A toolbar follows with icons for file operations like Open, Save, and Run. The main area is titled "<untitled> *". The code editor contains the following Python script:

```
1 # Student class for performance evaluation
2
3 class Student:
4     def __init__(self, name, roll_number, marks):
5         self.name = name
6         self.roll_number = roll_number
7         self.marks = marks
8
9     # Display student details
10    def display_details(self):
11        print("Name:", self.name)
12        print("Roll Number:", self.roll_number)
13        print("Marks:", self.marks)
14
15    # Check performance based on class average
16    def check_performance(self, class_average):
17        if self.marks > class_average:
18            return "Student is performing above average"
19        else:
20            return "Student is performing below average"
21
22
23 # Sample usage
24 student1 = Student("Rohan", "23CS101", 94)
25 student1.display_details()
26 print(student1.check_performance(75))
```

Below the code editor is a "Shell" tab. The shell output shows the execution of the script and its results:

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

Task 2: Data Processing in a Monitoring System

The screenshot shows the Thonny Python IDE interface. The top bar displays "Thonny - <untitled> @ 10:1" and a menu bar with File, Edit, View, Run, Tools, Help. Below the menu is a toolbar with icons for file operations and run/stop. The main editor window titled "<untitled> * x" contains the following Python code:

```
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
```

The bottom pane is a "Shell" window showing the output of running the script:

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

Task 3: Banking Transaction Simulation

The screenshot shows the Thonny Python IDE interface. The top menu bar includes File, Edit, View, Run, Tools, and Help. Below the menu is a toolbar with icons for file operations like Open, Save, and Run, along with a Stop button and a British flag icon.

The code editor window contains a script named 'untitled'. The code defines a `BankAccount` class with methods for depositing and withdrawing money, including a balance check. A sample usage of the class is shown at the bottom.

```
# BankAccount class for deposit and withdrawal
class BankAccount:
    def __init__(self, account_holder, balance):
        self.account_holder = account_holder
        self.balance = balance

    # Deposit money
    def deposit(self, amount):
        self.balance += amount
        print(f"Deposited {amount}. New Balance: {self.balance}")

    # Withdraw money with balance check
    def withdraw(self, amount):
        if amount <= self.balance:
            self.balance -= amount
            print(f"Withdrawn {amount}. Remaining Balance: {self.balance}")
        else:
            print("Insufficient balance. Withdrawal not allowed.")

# Sample usage
account = BankAccount("Rohan", 5000)
account.deposit(2000)
account.withdraw(3000)
account.withdraw(5000)
```

The shell window below shows the output of running the script:

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

>>>
```

Task 4: Student Scholarship Eligibility Check

The screenshot shows the Thonny Python IDE interface. The top window is titled "Thonny - <untitled> @ 17:1" and contains a code editor with the following Python script:

```
1 # List of students with scores
2
3 students = [
4     {"name": "Amit", "score": 80},
5     {"name": "Sneha", "score": 72},
6     {"name": "Kiran", "score": 90},
7     {"name": "Priya", "score": 65}
8 ]
9
10 index = 0
11
12 # While loop to check scholarship eligibility
13 while index < len(students):
14     if students[index]["score"] > 75:
15         print(students[index]["name"], "is eligible for scholarship")
16     index += 1
17 |
```

The bottom window is titled "Shell" and displays the output of running the script:

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

Task 5: Online Shopping Cart Module

Thonny - <untitled> @ 36:1

File Edit View Run Tools Help

<untitled> * x

```
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 x

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