SHRI MADHWA VADIRAJA INSTITUTE OF TECHNOLOGY & MANAGEMENT

An Autonomous Institution Affiliated to VTU, Belagavi

Approved by AICTE, New Delhi | Accredited by NAAC with 'A' Grade

(A unit of Shri Sode Vadiraja Mutt Education Trust®, Udupi)

Vishwothama Nagar, Bantakal – 574115, Udupi District, Karnataka

Course Title: PYTHON PROGRAMMING

1st Semester B.E

Academic Year: 2025–26

Submitted By:

Name:

Roll no./USN:

Department: Artificial Intelligence and Data Science

Section: D

Date: 01-01-2026

School Result Management: Create a Result class to manage student data. Use lists of marks, NumPy for mean and deviation, Pandas for tabular report, and file I/O for storage. Demonstrate shallow and deep copies of the result object

Course Instructor: Dr Deepika B.V

Signature:

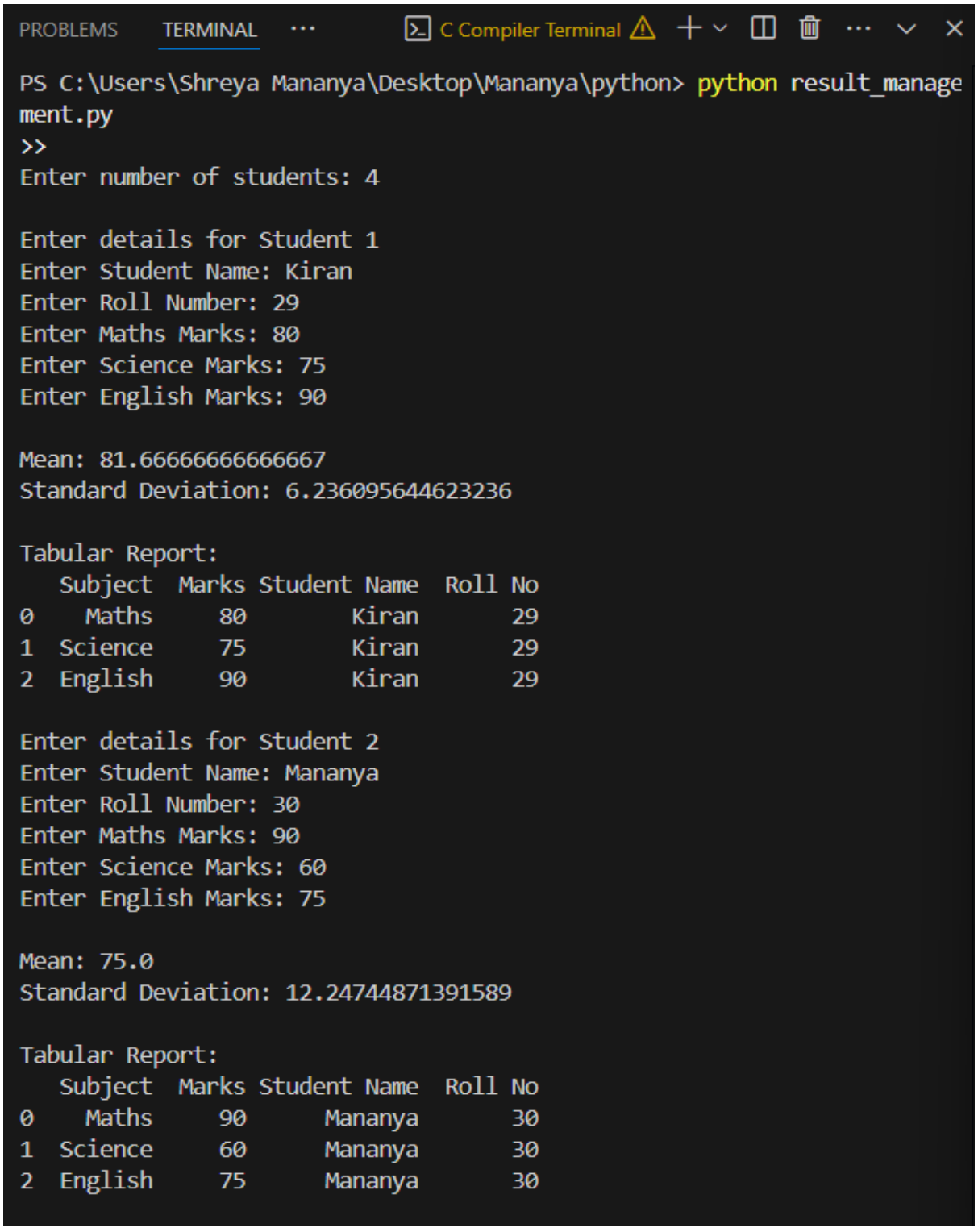
CODE:

```
# Importing NumPy library for numerical calculations
import numpy as np
# Importing Pandas library for creating tabular reports
import pandas as pd
# Importing copy module to demonstrate shallow and deep copy
import copy
# Defining Result class to manage student result details
class Result:
    # Constructor to initialize student name, roll number and marks
    def __init__(self, name, roll, marks):
        self.name = name       # Stores student name
        self.roll = roll       # Stores student roll number
        self.marks = marks     # Stores list of subject marks

    # Method to calculate mean (average) of marks using NumPy
    def calculate_mean(self):
        return np.mean(self.marks)
    # Method to calculate standard deviation of marks using NumPy
    def calculate_std(self):
        return np.std(self.marks)
    # Method to generate tabular report using Pandas DataFrame
    def generate_report(self):
        data = {
            "Subject": ["Maths", "Science", "English"],  # List of subjects
            "Marks": self.marks                    # Corresponding marks
        }
        df = pd.DataFrame(data)         # Creating DataFrame
        df["Student Name"] = self.name   # Adding student name column
```

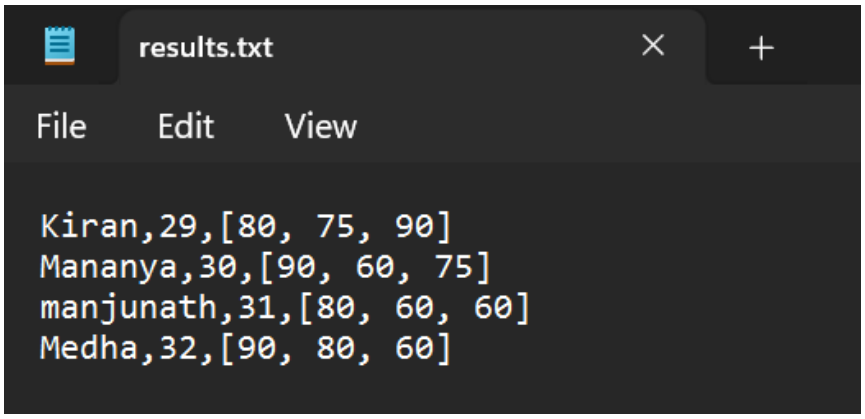
```
        df["Roll No"] = self.roll       # Adding roll number column
        return df

    # Method to save student result details into a file
    def save_to_file(self, filename):
        with open(filename, "a") as f:   # Opening file in append mode
            f.write(f"{self.name},{self.roll},{self.marks}\n")
# ---------- MAIN PROGRAM ----------

# List to store Result objects of multiple students
students = []
# Taking number of students as input
n = int(input("Enter number of students: "))
# Loop to accept details of each student
for i in range(n):
    print(f"\nEnter details for Student {i+1}")
    name = input("Enter Student Name: ")     # Input student name
    roll = int(input("Enter Roll Number: ")) # Input roll number
    # List to store marks of three subjects
    marks = []
    marks.append(int(input("Enter Maths Marks: ")))
    marks.append(int(input("Enter Science Marks: ")))
    marks.append(int(input("Enter English Marks: ")))
    # Creating Result object for the student
    student = Result(name, roll, marks)
    # Adding student object to students list
    students.append(student)
    # Displaying mean and standard deviation
    print("\nMean:", student.calculate_mean())
    print("Standard Deviation:", student.calculate_std())
```

```
    # Displaying tabular report
    print("\nTabular Report:")
    print(student.generate_report())
    # Saving student data to file
    student.save_to_file("results.txt")

# -------- SHALLOW & DEEP COPY DEMONSTRATION --------
# Checking if at least one student exists
if n > 0:
    print("\n--- Shallow and Deep Copy Demonstration (First Student) ---")
    # Original object reference
    original = students[0]
    # Creating shallow copy of object
    shallow_copy = copy.copy(original)
    # Modifying marks in shallow copy
    shallow_copy.marks[0] = 50
    # Showing effect of shallow copy
    print("\nAfter Shallow Copy Change:")
    print("Original Marks:", original.marks)
    print("Shallow Copy Marks:", shallow_copy.marks)
    # Creating deep copy of object
    deep_copy = copy.deepcopy(original)
    # Modifying marks in deep copy
    deep_copy.marks[1] = 40
    # Showing effect of deep copy
    print("\nAfter Deep Copy Change:")
    print("Original Marks:", original.marks)
    print("Deep Copy Marks:", deep_copy.marks)
```

OUTPUT:


PROBLEMS
TERMINAL
C Compiler Terminal
PS C:\Users\Shreya Mananya\Desktop\Mananya\python> python result_manage
ment.py
>>
Enter number of students: 4
Enter details for Student 1
Enter Student Name: Kiran
Enter Roll Number: 29
Enter Maths Marks: 80
Enter Science Marks: 75
Enter English Marks: 90
Mean: 81.66666666666667
Standard Deviation: 6.236095644623236
Tabular Report:
Subject Marks Student Name Roll No
0 Maths 80 Kiran 29
1 Science 75 Kiran 29
2 English 90 Kiran 29
Enter details for Student 2
Enter Student Name: Mananya
Enter Roll Number: 30
Enter Maths Marks: 90
Enter Science Marks: 60
Enter English Marks: 75
Mean: 75.0
Standard Deviation: 12.24744871391589
Tabular Report:
Subject Marks Student Name Roll No
0 Maths 90 Mananya 30
1 Science 60 Mananya 30
2 English 75 Mananya 30

```
Enter Student Name: manjunath
Enter Roll Number: 31
Enter Maths Marks: 80
Enter Science Marks: 60
Enter English Marks: 60

Mean: 66.66666666666667
Standard Deviation: 9.428090415820634

Tabular Report:
   Subject  Marks Student Name  Roll No
0    Maths     80    manjunath       31
1  Science     60    manjunath       31
2  English     60    manjunath       31

Enter details for Student 4
Enter Student Name: Medha
Enter Roll Number: 32
Enter Maths Marks: 90
Enter Science Marks: 80
Enter English Marks: 60

Mean: 76.66666666666667
Standard Deviation: 12.472191289246473

Tabular Report:
   Subject  Marks Student Name  Roll No
0    Maths     90        Medha       32
1  Science     80        Medha       32
2  English     60        Medha       32

--- Shallow and Deep Copy Demonstration (First Student) ---

After Shallow Copy Change:
Original Marks: [50, 75, 90]
Shallow Copy Marks: [50, 75, 90]

After Deep Copy Change:
Original Marks: [50, 75, 90]
Deep Copy Marks: [50, 40, 90]
```



1. Variables Used, Their Types, and Purpose

1. **name (string)**
   - Stores the name of the student.
   - Used to identify the student in the report and file storage.
2. **roll (integer)**
   - Stores the roll number of the student.
   - Used as a unique identifier for each student.
3. **marks (list of integers)**
   - Stores marks of subjects (Maths, Science, English).
   - Used to calculate mean and standard deviation.
4. **students (list of objects)**
   - Stores multiple `Result` objects.
   - Used to manage data of more than one student.
5. **n (integer)**
   - Stores number of students entered by the user.
   - Used to control loop execution.
6. **filename (string)**
   - Stores the file name (`results.txt`).
   - Used for saving student data permanently.

---

2. Data Structures Used and Reason

List

- Used to store marks of subjects.
- Allows storing multiple values in a single variable.
- Easy to pass to NumPy functions for calculations.

Object (Class Result)

- Used to combine student data and related functions.
- Helps in organizing code and reusability.
- Represents a real-world entity (student result).

---

3. Other Main Concepts Used

Class and Object

- `Result` class is used to define student result structure.
- Objects represent individual students.

NumPy

- Used to calculate **mean** and **standard deviation**.
- Provides fast and accurate mathematical operations.

Pandas

- Used to create a **tabular report** using DataFrame.
- Makes data easy to read and present.

File Handling

- Used to store student results in a text file.
- Ensures data is saved permanently.

Shallow Copy

- Copies object reference.
- Changes in copied object affect original object.

Deep Copy

- Creates an independent copy.
- Changes in copied object do not affect original object.

Looping and Input

- `for` loop is used to take details of multiple students.
- `input()` is used for dynamic data entry.