“Python Programming”

Assignment-2

Topic – Analysing and Reporting Student Grades

Submitted by – Priyam Sharma

Roll no - 2501730184

Course – B. Tech CSE (AI & ML)

Section – B

Faculty name- Mr Sameer Farooq

Introduction

The objective of this Python Lab assignment is to develop a GradeBook Analyser, a program capable of reading student marks, performing statistical analysis, assigning grades, and presenting results clearly. The project demonstrates Python skills such as:

- File handling
- Functions and modular programming
- Data validation
- Statistics (mean, median)
- Grade assignment logic
- Table formatting for output

Objectives

-To develop a Python program that can efficiently analyse student marks.

-To implement file handling using CSV files

-To practice modular programming using functions

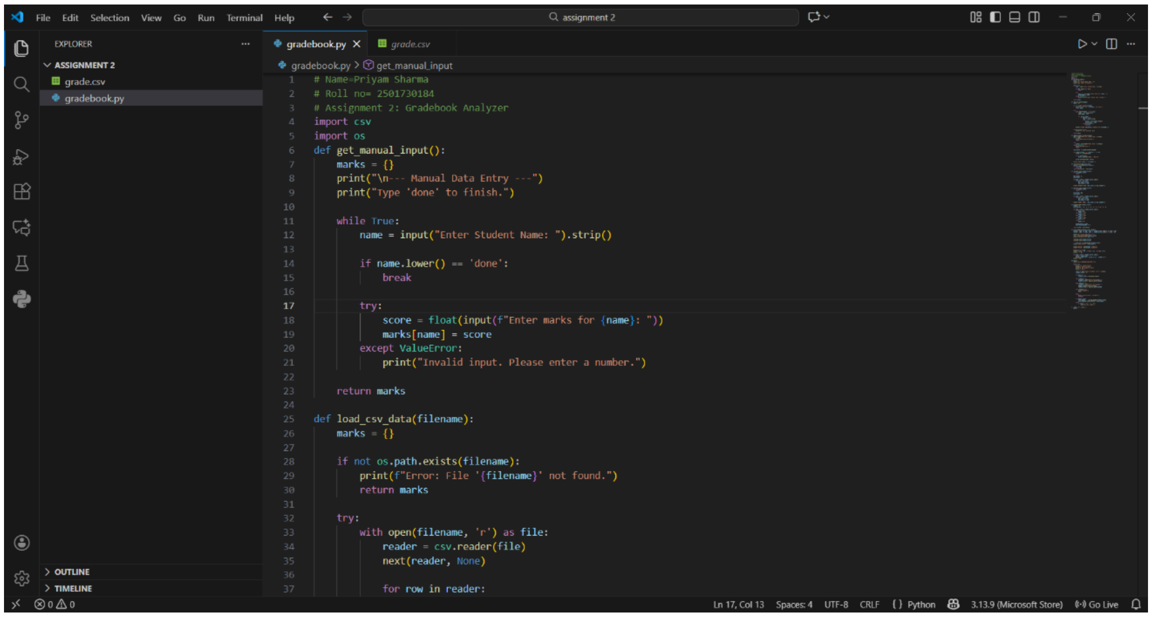
-To design a grading system based on score ranges

-To generate summaries for academic performance evaluation

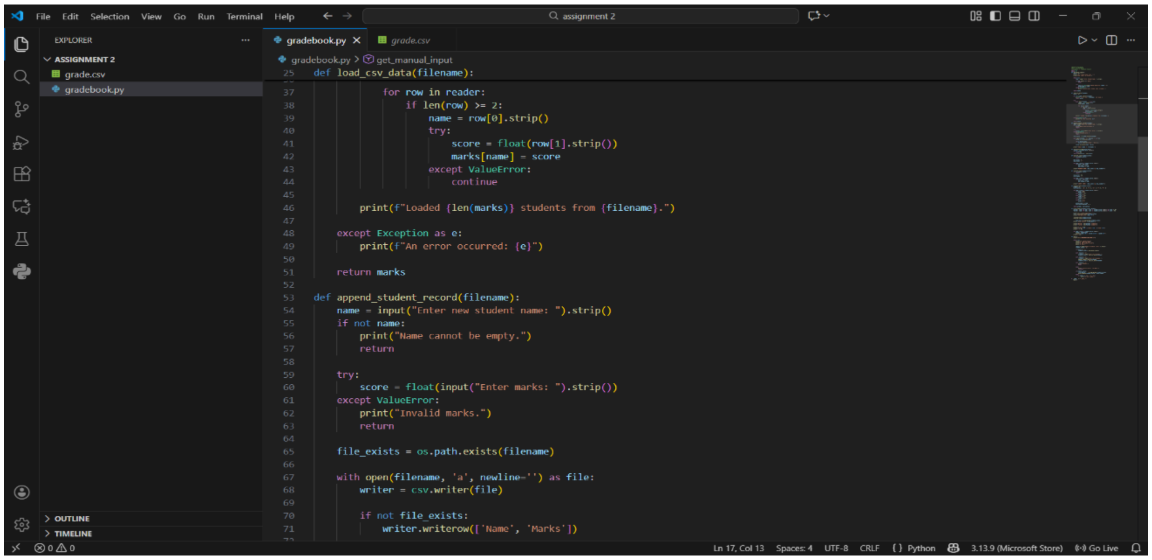
Program Description

The program developed (gradebook_analyzer.py) is designed to analyse student marks and generate performance summaries. It allows users to either enter marks manually or load them from a CSV file.

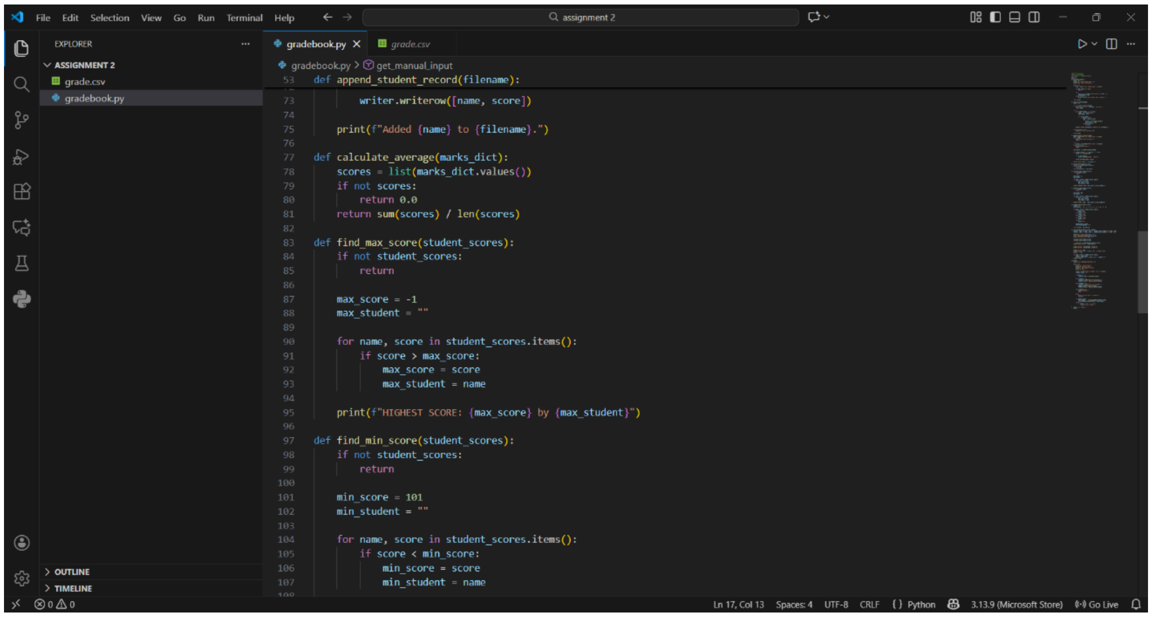
Program Code



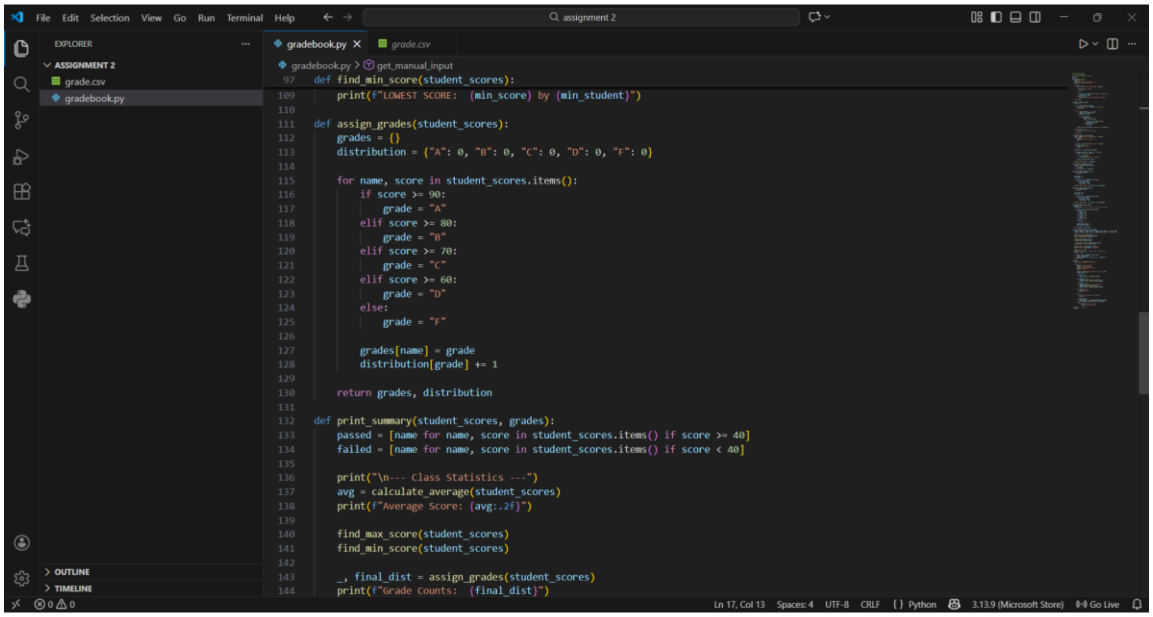
```python
# Name=Priyam Sharma
# Roll no= 2501730184
# Assignment 2: Gradebook Analyzer
import csv
import os
def get_manual_input():
    marks = {}
    print("\n--- Manual Data Entry ---")
    print("Type 'done' to finish.")

    while True:
        name = input("Enter Student Name: ").strip()

        if name.lower() == 'done':
            break

        try:
            score = float(input(f"Enter marks for {name}: "))
            marks[name] = score
        except ValueError:
            print("Invalid input. Please enter a number.")

    return marks

def load_csv_data(filename):
    marks = {}

    if not os.path.exists(filename):
        print(f"Error: File '{filename}' not found.")
        return marks

    try:
        with open(filename, 'r') as file:
            reader = csv.reader(file)
            next(reader, None)

            for row in reader:
```



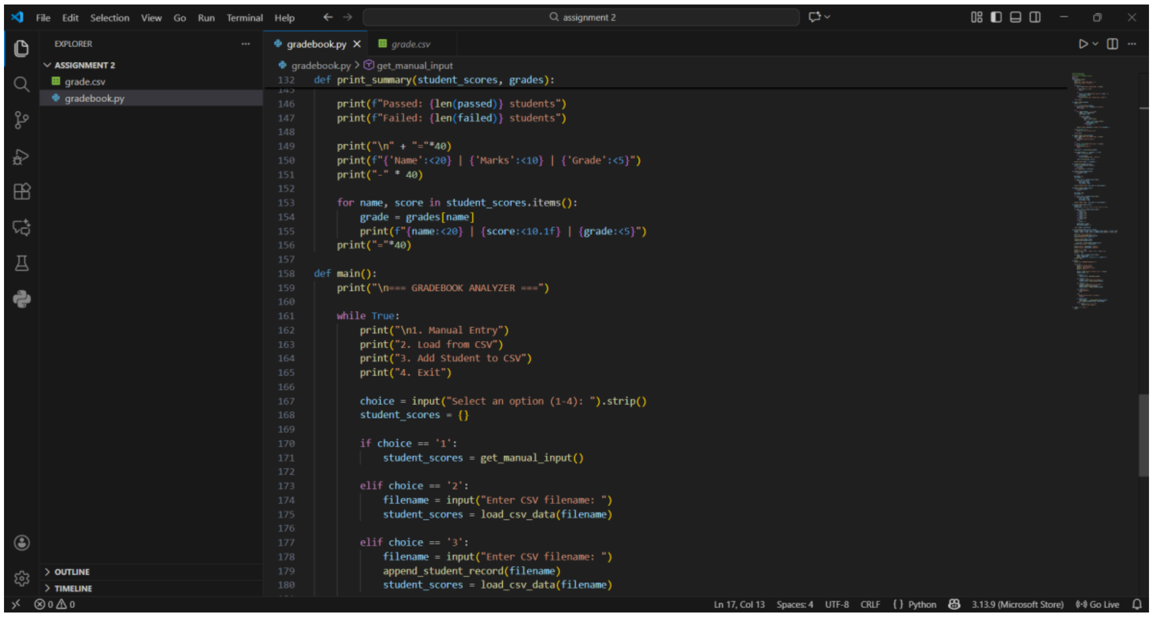
```python
            for row in reader:
                if len(row) >= 2:
                    name = row[0].strip()
                    try:
                        score = float(row[1].strip())
                        marks[name] = score
                    except ValueError:
                        continue

        print(f"Loaded {len(marks)} students from {filename}.")

    except Exception as e:
        print(f"An error occurred: {e}")

    return marks

def append_student_record(filename):
    name = input("Enter new student name: ").strip()
    if not name:
        print("Name cannot be empty.")
        return

    try:
        score = float(input("Enter marks: ").strip())
    except ValueError:
        print("Invalid marks.")
        return

    file_exists = os.path.exists(filename)

    with open(filename, 'a', newline='') as file:
        writer = csv.writer(file)

        if not file_exists:
            writer.writerow(['Name', 'Marks'])
```


File Edit Selection View Go Run Terminal Help
assignment 2
EXPLORER
ASSIGNMENT 2
grade.csv
gradebook.py
gradebook.py
grade.csv
gradebook.py > get_manual_input
OUTLINE
TIMELINE
Ln 17, Col 13 Spaces: 4 UTF-8 CRLF {} Python 3.13.9 (Microsoft Store) Go Live

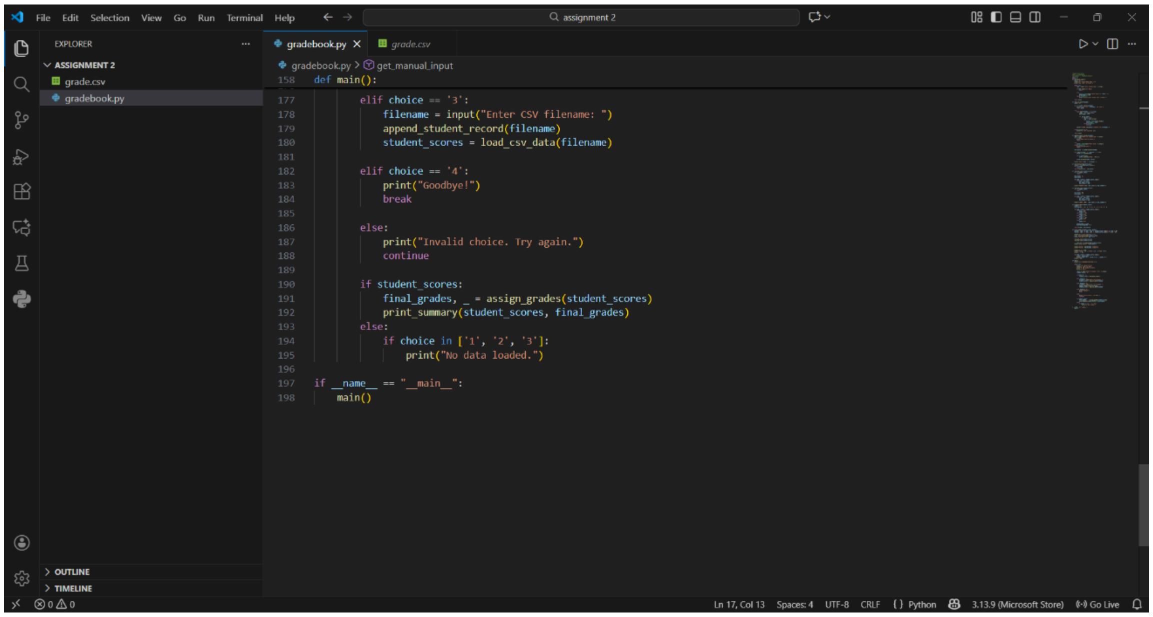
```python
def append_student_record(filename):
            writer.writerow([name, score])

    print(f"Added {name} to {filename}.")

def calculate_average(marks_dict):
    scores = list(marks_dict.values())
    if not scores:
        return 0.0
    return sum(scores) / len(scores)

def find_max_score(student_scores):
    if not student_scores:
        return

    max_score = -1
    max_student = ""

    for name, score in student_scores.items():
        if score > max_score:
            max_score = score
            max_student = name

    print(f"HIGHEST SCORE: {max_score} by {max_student}")

def find_min_score(student_scores):
    if not student_scores:
        return

    min_score = 101
    min_student = ""

    for name, score in student_scores.items():
        if score < min_score:
            min_score = score
            min_student = name
```


File Edit Selection View Go Run Terminal Help
assignment 2
EXPLORER
ASSIGNMENT 2
grade.csv
gradebook.py
gradebook.py
grade.csv
gradebook.py > get_manual_input
OUTLINE
TIMELINE
Ln 17, Col 13 Spaces: 4 UTF-8 CRLF {} Python 3.13.9 (Microsoft Store) Go Live

```python
def find_min_score(student_scores):
    print(f"LOWEST SCORE:  {min_score} by {min_student}")

def assign_grades(student_scores):
    grades = {}
    distribution = {"A": 0, "B": 0, "C": 0, "D": 0, "F": 0}

    for name, score in student_scores.items():
        if score >= 90:
            grade = "A"
        elif score >= 80:
            grade = "B"
        elif score >= 70:
            grade = "C"
        elif score >= 60:
            grade = "D"
        else:
            grade = "F"

        grades[name] = grade
        distribution[grade] += 1

    return grades, distribution

def print_summary(student_scores, grades):
    passed = [name for name, score in student_scores.items() if score >= 40]
    failed = [name for name, score in student_scores.items() if score < 40]

    print("\n--- Class Statistics ---")
    avg = calculate_average(student_scores)
    print(f"Average Score: {avg:.2f}")

    find_max_score(student_scores)
    find_min_score(student_scores)

    _, final_dist = assign_grades(student_scores)
    print(f"Grade Counts:  {final_dist}")
```

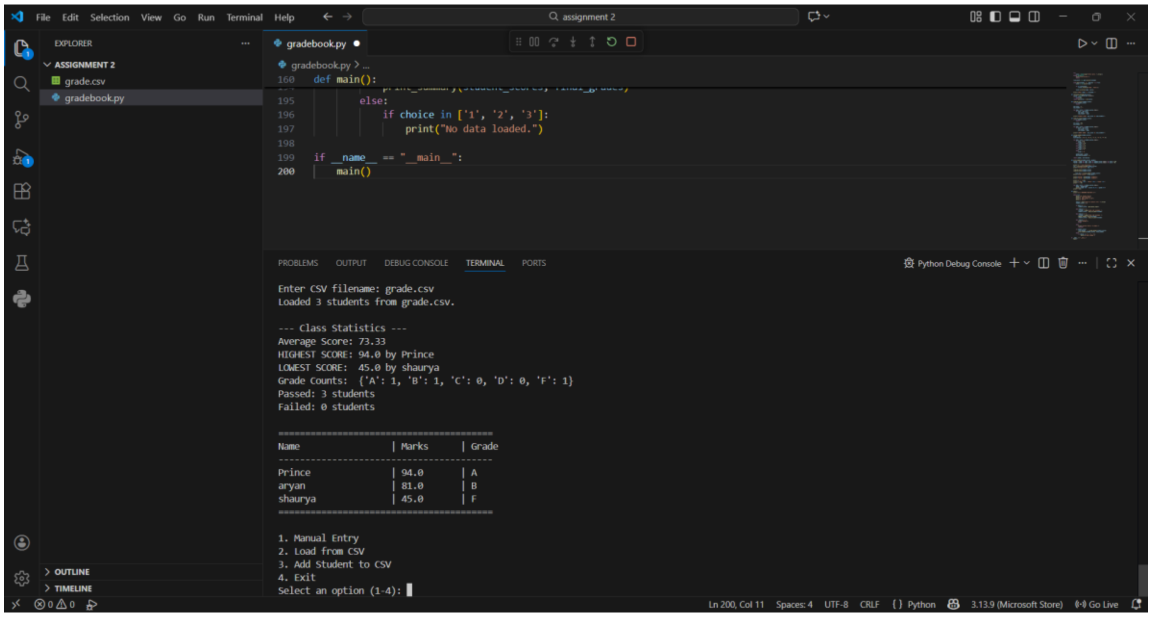


```python
def print_summary(student_scores, grades):

    print(f"Passed: {len(passed)} students")
    print(f"Failed: {len(failed)} students")

    print("\n" + "="*40)
    print(f"{'Name':<20} | {'Marks':<10} | {'Grade':<5}")
    print("-" * 40)

    for name, score in student_scores.items():
        grade = grades[name]
        print(f"{name:<20} | {score:<10.1f} | {grade:<5}")
    print("="*40)

def main():
    print("\n=== GRADEBOOK ANALYZER ===")

    while True:
        print("\n1. Manual Entry")
        print("2. Load from CSV")
        print("3. Add Student to CSV")
        print("4. Exit")

        choice = input("Select an option (1-4): ").strip()
        student_scores = {}

        if choice == '1':
            student_scores = get_manual_input()

        elif choice == '2':
            filename = input("Enter CSV filename: ")
            student_scores = load_csv_data(filename)

        elif choice == '3':
            filename = input("Enter CSV filename: ")
            append_student_record(filename)
            student_scores = load_csv_data(filename)
```



```python
def main():

        elif choice == '3':
            filename = input("Enter CSV filename: ")
            append_student_record(filename)
            student_scores = load_csv_data(filename)

        elif choice == '4':
            print("Goodbye!")
            break

        else:
            print("Invalid choice. Try again.")
            continue

        if student_scores:
            final_grades, _ = assign_grades(student_scores)
            print_summary(student_scores, final_grades)
        else:
            if choice in ['1', '2', '3']:
                print("No data loaded.")

if __name__ == "__main__":
    main()
```

Sample Output


File Edit Selection View Go Run Terminal Help
assignment 2
EXPLORER
ASSIGNMENT 2
grade.csv
gradebook.py
OUTLINE
TIMELINE
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Python Debug Console
Ln 200, Col 11 Spaces: 4 UTF-8 CRLF Python 3.13.9 (Microsoft Store) Go Live

```python
def main():
        else:
            if choice in ['1', '2', '3']:
                print("No data loaded.")

if __name__ == "__main__":
    main()
```

```
Enter CSV filename: grade.csv
Loaded 3 students from grade.csv.

--- Class Statistics ---
Average Score: 73.33
HIGHEST SCORE: 94.0 by Prince
LOWEST SCORE:  45.0 by shaurya
Grade Counts:  {'A': 1, 'B': 1, 'C': 0, 'D': 0, 'F': 1}
Passed: 3 students
Failed: 0 students

========================================
Name                 | Marks      | Grade
----------------------------------------
Prince               | 94.0       | A
aryan                | 81.0       | B
shaurya              | 45.0       | F
========================================

1. Manual Entry
2. Load from CSV
3. Add Student to CSV
4. Exit
Select an option (1-4):
```

Conclusion

The *GradeBook Analyser* is an effective Python program that reads student marks, performs statistical analysis, assigns grades, and displays results clearly. It demonstrates essential programming skills such as file handling, functions, and data processing, providing a practical solution for evaluating student performance.