"Python Programming"

Assignment-2

Topic – Analysing and Reporting Student Grades

Submitted by – Meghna Kumar

Roll no - 2501730214

Course – B. Tech CSE (AI & ML)

Section – A

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

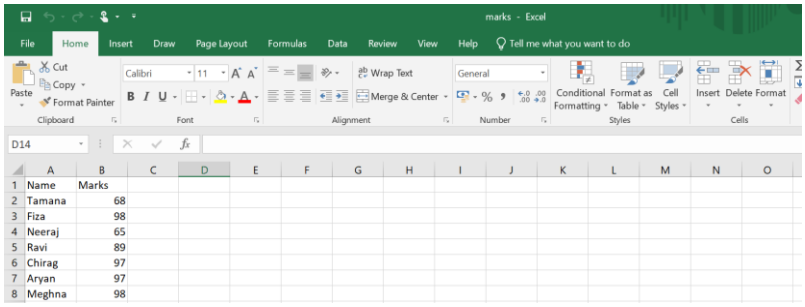
Program Description

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.

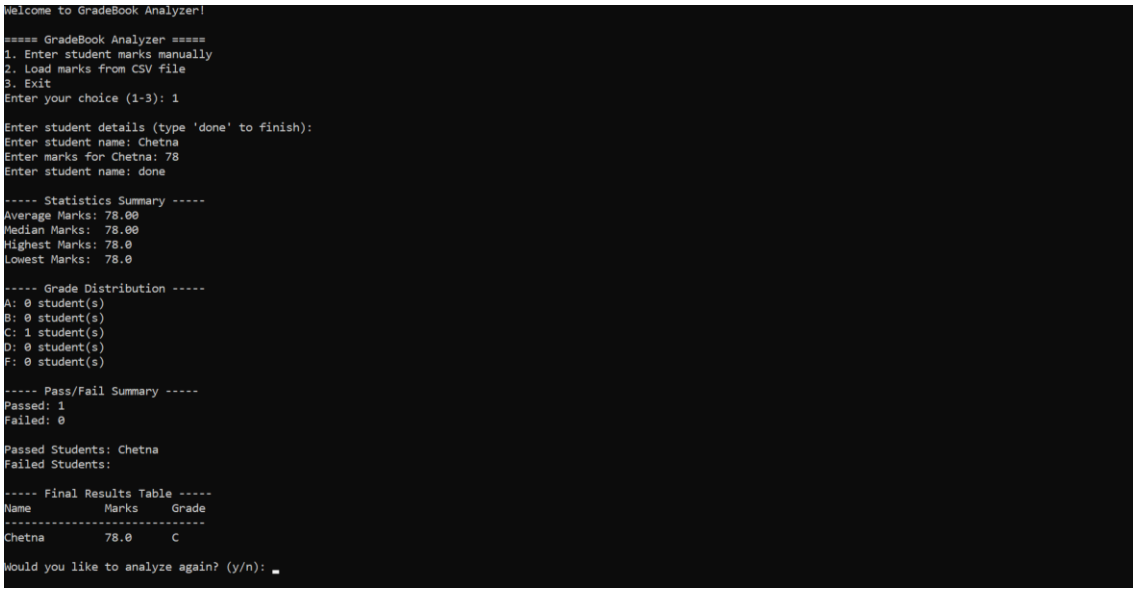
Program Code

```
# gradebook.py
# Author: Meghna Kumar
# Date: 30 November 2025
# Project Title: GradeBook Analyzer

import csv
import statistics


def display_menu():
    print("\n===== GradeBook Analyzer =====")
    print("1. Enter student marks manually")
    print("2. Load marks from CSV file")
    print("3. Exit")


def get_manual_input():
    marks = {}
    print("\nEnter student details (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 get_csv_input(filename):
    marks = {}
    try:
        with open(filename, 'r') as file:
            reader = csv.DictReader(file)
            for row in reader:
                marks[row['Name']] = float(row['Marks'])
        print(f"\n Loaded data for {len(marks)} students from {filename}.")
    except FileNotFoundError:
        print("File not found. Please check the file name.")
    return marks
    return marks


def calculate_average(marks_dict):
    return sum(marks_dict.values()) / len(marks_dict)

def calculate_median(marks_dict):
    return statistics.median(marks_dict.values())

def find_max_score(marks_dict):
    return max(marks_dict.values())

def find_min_score(marks_dict):
    return min(marks_dict.values())

def show_statistics(marks_dict):
    print("\n----- Statistics Summary -----")
    print(f"Average Marks: {calculate_average(marks_dict):.2f}")
    print(f"Median Marks:  {calculate_median(marks_dict):.2f}")
    print(f"Highest Marks: {find_max_score(marks_dict)}")
    print(f"Lowest Marks:  {find_min_score(marks_dict)}")


def assign_grades(marks_dict):
    grades = {}
    for name, score in marks_dict.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
    return grades

def grade_distribution(grades_dict):
    distribution = {'A': 0, 'B': 0, 'C': 0, 'D': 0, 'F': 0}
    for g in grades_dict.values():
```

```python
def grade_distribution(grades_dict):
    distribution = {'A': 0, 'B': 0, 'C': 0, 'D': 0, 'F': 0}
    for g in grades_dict.values():
        distribution[g] += 1
    print("\n----- Grade Distribution -----")
    for grade, count in distribution.items():
        print(f"{grade}: {count} student(s)")


def pass_fail_list(marks_dict):
    passed_students = [name for name, marks in marks_dict.items() if marks >= 40]
    failed_students = [name for name, marks in marks_dict.items() if marks < 40]

    print("\n----- Pass/Fail Summary -----")
    print(f"Passed: {len(passed_students)}")
    print(f"Failed: {len(failed_students)}")
    print("\nPassed Students:", ', '.join(passed_students))
    print("Failed Students:", ', '.join(failed_students))

    return passed_students, failed_students


def display_table(marks_dict, grades_dict):
    print("\n----- Final Results Table -----")
    print(f"{'Name':<15}{'Marks':<10}{'Grade':<5}")
    print("-" * 30)
    for name, marks in marks_dict.items():
        print(f"{name:<15}{marks:<10}{grades_dict[name]:<5}")


def main():
    print("Welcome to GradeBook Analyzer!")

    while True:
        display_menu()
        choice = input("Enter your choice (1-3): ").strip()

        if choice == '1':
            marks = get_manual_input()
        elif choice == '2':
            filename = input("Enter CSV filename (e.g. grades.csv): ").strip()
            marks = get_csv_input(filename)
        elif choice == '3':
            print(" Exiting... Thank you for using GradeBook Analyzer!")
            break
        else:
            print(" Invalid choice. Try again.")
            continue

        if not marks:
            print("No data found. Try again.")
            continue


        show_statistics(marks)
        grades = assign_grades(marks)
        grade_distribution(grades)
        pass_fail_list(marks)
        display_table(marks, grades)


        again = input("\nWould you like to analyze again? (y/n): ").strip().lower()
        if again != 'y':
            print(" Goodbye! Have a great day.")
            break

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



Sample Output



```
Welcome to GradeBook Analyzer!

===== GradeBook Analyzer =====
1. Enter student marks manually
2. Load marks from CSV file
3. Exit
Enter your choice (1-3): 1

Enter student details (type 'done' to finish):
Enter student name: Chetna
Enter marks for Chetna: 78
Enter student name: done

----- Statistics Summary -----
Average Marks: 78.00
Median Marks:  78.00
Highest Marks: 78.0
Lowest Marks:  78.0

----- Grade Distribution -----
A: 0 student(s)
B: 0 student(s)
C: 1 student(s)
D: 0 student(s)
F: 0 student(s)

----- Pass/Fail Summary -----
Passed: 1
Failed: 0

Passed Students: Chetna
Failed Students:

----- Final Results Table -----
Name            Marks     Grade
------------------------------
Chetna          78.0      C

Would you like to analyze again? (y/n): _
```

```
===== GradeBook Analyzer =====
1. Enter student marks manually
2. Load marks from CSV file
3. Exit
Enter your choice (1-3): 2
Enter CSV filename (e.g. grades.csv): marks.csv

 Loaded data for 7 students from marks.csv.

----- Statistics Summary -----
Average Marks: 87.43
Median Marks:  97.00
Highest Marks: 98.0
Lowest Marks:  65.0

----- Grade Distribution -----
A: 4 student(s)
B: 1 student(s)
C: 0 student(s)
D: 2 student(s)
F: 0 student(s)

----- Pass/Fail Summary -----
Passed: 7
Failed: 0

Passed Students: Tamana, Fiza, Neeraj, Ravi, Chirag, Aryan, Meghna
Failed Students:

----- Final Results Table -----
Name            Marks     Grade
------------------------------
Tamana          68.0      D
Fiza            98.0      A
Neeraj          65.0      D
Ravi            89.0      B
Chirag          97.0      A
Aryan           97.0      A
Meghna          98.0      A

Would you like to analyze again? (y/n):
```

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.