National University of Science and Technology (NUST)

School of Mechanical & Manufacturing Engineering (SMME)

PROJECT REPORT

NAME: AFNAN ABBAS HAIDER

CMS I'D: 462250

Program: BE-Aerospace

Section: AE-01

Session: Fall 2024

Semester: 3rd

Instructor: Laiba Waheed

Contents

GPA Calculator and Visualizer

Abstract

This project introduces a **GPA Calculator and Visualizer**, a comprehensive tool designed to assist students in calculating and visualizing their academic performance. By accepting user inputs for course names, grades, and credit hours, the program calculates the Grade Point Average (GPA) and displays a bar chart that highlights the contribution of each course to the total quality points. The project leverages Python's data manipulation and visualization libraries, pandas and matplotlib, to provide a user-friendly and efficient solution. This report outlines the objectives, underlying theory, and implementation of the project, culminating in a detailed explanation of its functionality.

Introduction

In academic settings, GPA serves as a standardized metric to assess students' overall performance. However, manual GPA calculation can be error-prone and time-consuming, especially when handling multiple courses with different credit weights. Additionally, understanding how each course impacts the overall GPA is crucial for students aiming to improve their performance.

The **GPA Calculator and Visualizer** automates this process, offering a streamlined and accurate way to compute GPAs. By visualizing the quality points for each course, the project aids in identifying courses with the highest impact on a student's GPA. This project serves both as an educational tool and as a performance tracker, helping students make informed decisions regarding their academic priorities.

Theory

The **Grade Point Average (GPA)** is a key metric used to evaluate academic performance. It is calculated by dividing the total **Quality Points** earned by the total **Credit Hours** attempted:

$$GPA = Total\ Quality\ Points/\text{Total Credits}$$

The following concepts underpin the GPA calculation:

1. Grade Points:
   Each letter grade corresponds to a numeric value:

   A: 4.0

   B: 3.0

   C: 2.0

   D: 1.0

   F: 0.0

2. Quality Points:
   Quality points are calculated by multiplying the grade points by the credit hours for a course:

   Quality points = Grade points x credits

3. Total Credits and Total Quality Points:
   To compute the GPA, the total credits and total quality points for all courses are summed.

Explanation of Code

The program consists of several key components, each designed to fulfill a specific function:

1. Grade Mapping Function

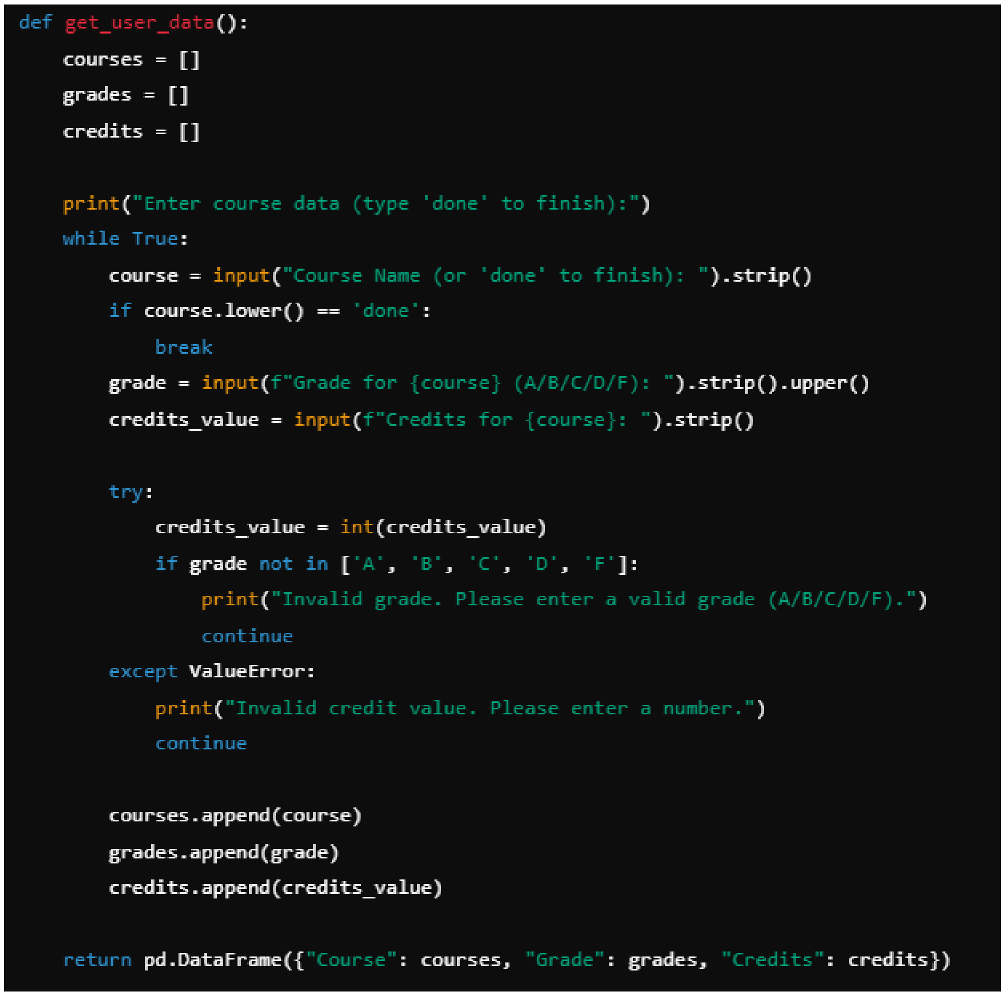
The grade_to_points function maps user-provided letter grades to their corresponding numeric grade points using a dictionary. This ensures a consistent and efficient way to translate grades:

```python
def grade_to_points(grade):
    grade_points = {
        "A": 4.0,
        "B": 3.0,
        "C": 2.0,
        "D": 1.0,
        "F": 0.0
    }
    return grade_points.get(grade.upper(), 0.0)
```

The function returns 0.0 for invalid or unrecognized grades, ensuring the program handles unexpected input gracefully.

2. User Input Function

The get_user_data function collects data from the user about their courses, grades, and credit hours. The function ensures the input is valid and stores the data in a structured pandas DataFrame:



```python
def get_user_data():
    courses = []
    grades = []
    credits = []

    print("Enter course data (type 'done' to finish):")
    while True:
        course = input("Course Name (or 'done' to finish): ").strip()
        if course.lower() == 'done':
            break
        grade = input(f"Grade for {course} (A/B/C/D/F): ").strip().upper()
        credits_value = input(f"Credits for {course}: ").strip()

        try:
            credits_value = int(credits_value)
            if grade not in ['A', 'B', 'C', 'D', 'F']:
                print("Invalid grade. Please enter a valid grade (A/B/C/D/F).")
                continue
        except ValueError:
            print("Invalid credit value. Please enter a number.")
            continue

        courses.append(course)
        grades.append(grade)
        credits.append(credits_value)

    return pd.DataFrame({"Course": courses, "Grade": grades, "Credits": credits})
```

This function validates input for grades and credits, ensuring that only valid data is processed.

3. GPA Calculation Function

The calculated gpa function calculates the GPA by adding columns for grade points and quality points to the DataFrame. It then computes the total quality points and total credits, applying the GPA formula:

```python
def calculate_gpa(df):
    df['Grade Points'] = df['Grade'].apply(grade_to_points)
    df['Quality Points'] = df['Grade Points'] * df['Credits']

    total_quality_points = df['Quality Points'].sum()
    total_credits = df['Credits'].sum()

    gpa = total_quality_points / total_credits if total_credits > 0 else 0.0
    return gpa, total_credits, total_quality_points
```

The function provides both the calculated GPA and the supporting data for quality points and credits.

4. Visualization Function

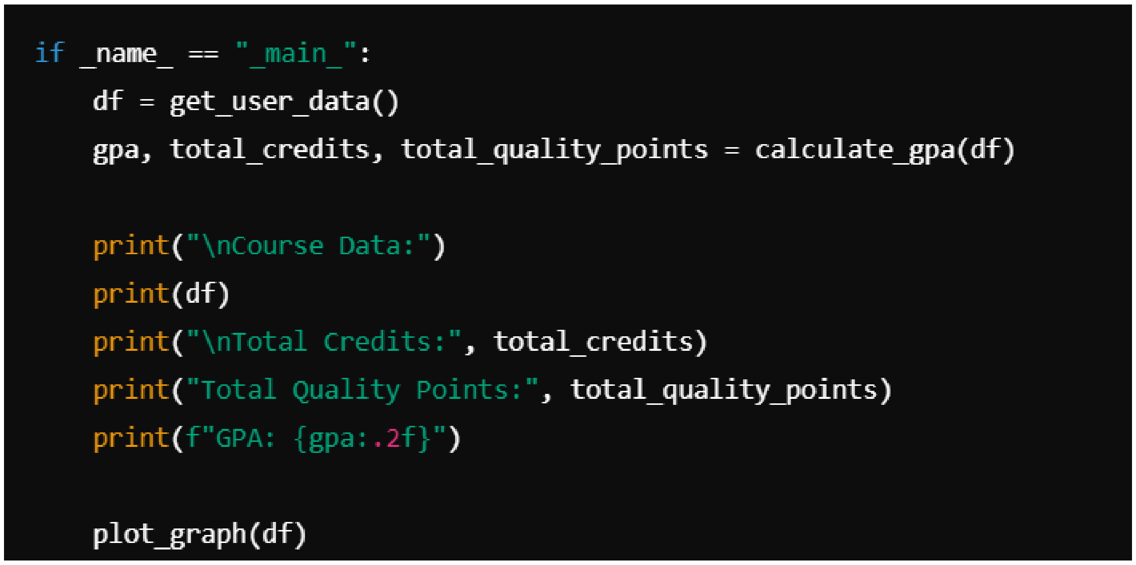
The plot_graph function uses matplotlib to create a bar chart displaying quality points for each course. This visualization helps users understand how each course contributes to their GPA:

```python
def plot_graph(df):
    plt.figure(figsize=(10, 6))
    plt.bar(df['Course'], df['Quality Points'], color='skyblue', edgecolor='black')
    plt.title('Quality Points by Course', fontsize=16)
    plt.xlabel('Courses', fontsize=12)
    plt.ylabel('Quality Points', fontsize=12)
    plt.xticks(rotation=45, fontsize=10)
    plt.yticks(fontsize=10)
    plt.grid(axis='y', linestyle='--', alpha=0.7)
    plt.tight_layout()
    plt.show()
```

The graph provides a clear, visual summary of the data.

5. Main Execution Flow

The program's main section orchestrates the user interaction, calculation, and visualization:



```
if _name_ == "_main_":
    df = get_user_data()
    gpa, total_credits, total_quality_points = calculate_gpa(df)

    print("\nCourse Data:")
    print(df)
    print("\nTotal Credits:", total_credits)
    print("Total Quality Points:", total_quality_points)
    print(f"GPA: {gpa:.2f}")

    plot_graph(df)
```

This section ties all the components together, guiding the user through the input process, displaying results, and showing the graph.

Conclusion

The **GPA Calculator and Visualizer** efficiently automates the GPA calculation process, ensuring accuracy and providing an intuitive visual summary of academic performance. It simplifies the task for students, offering clear insights into their academic strengths and areas needing improvement. Future enhancements could include advanced features like data storage and a graphical user interface, broadening its usability and appeal.

References

- Python Official Documentation: https://docs.python.org/
- pandas Documentation: https://pandas.pydata.org/docs/
- matplotlib Documentation: https://matplotlib.org/stable/contents.html