PROJECTS' DESCRIPTION

1. Student Grade Tracker (Using Dictionaries)

- **Concepts Covered**: Dictionaries, Lists, Functions, JSON Conversion.
- **Description**:
  - Create a program allowing users to enter student names and scores in multiple subjects.
  - Store the data in a nested dictionary format.
  - Implement functions to:
    - Add new students and update scores.
    - Calculate and display the average grade per student.
    - Convert the dictionary to a JSON string and save it to a file.
- **Extensions**:
  - Load and save data from a JSON file for persistence.

Advanced Student Grade Tracker (Using Dictionaries, JSON, File Handling, and Data Analysis)

- **Concepts Covered**: Dictionaries, Lists, Functions, JSON Serialization, File Handling, and Data Analysis.
- **Enhancements**:
  - Store student records with:
    - **ID, Name, Courses, Scores, Attendance, and Remarks**.
  - Implement the following features:
    - **Add a new student** and automatically generate a unique **student ID**.
    - **Modify scores** and calculate the GPA.
    - **Analyze performance trends** using **Matplotlib** (e.g., grade distribution).
    - Save student records in a **JSON file** and retrieve data when reopening the program.
- **Extensions**:
  - Rank students based on GPA.

Integrate an attendance system that deducts marks for low attendance.

---

2. Word Frequency Analyzer (Using Strings & Dictionaries)

- **Concepts Covered**: String Manipulation, Dictionaries, Loops.
- **Description**:
  - Ask the user for a text input.
  - Count the occurrences of each word in the text.
  - Display the top N most frequent words.
- **Extensions**:
  - Use **file handling** to analyze word frequency from a text file.

Text Analyzer & Sentiment Detector (Using NLP & File Processing)

- **Concepts Covered**: String Manipulation, Dictionaries, File Handling, NLP (Natural Language Processing).
- **Enhancements**:
  - Ask users to input a **text file**.
  - Perform:
    - **Word frequency analysis**.
    - **Sentence segmentation** (split into sentences).
    - **Named Entity Recognition (NER)** (find names, locations, dates).
    - **Sentiment analysis** using **VADER (NLTK)**.
  - Visualize results (word cloud, bar chart).
- **Extensions**:
  - Detect spam or offensive language.

Summarize the text automatically.

---

3. To-Do List with Priority (Using Lists & Sets)

- **Concepts Covered**: Lists, Sets, User Input, File Handling.
- **Description**:
  - Create a command-line to-do list where users can:
    - Add, remove, and list tasks.
    - Assign priority levels (e.g., High, Medium, Low).
  - Store tasks in a **set** to prevent duplicates.
  - Save the tasks to a file and load them upon program start.
- **Extensions**:
  - Add deadlines and sort tasks by priority.

AI-Powered To-Do List with Smart Reminders

- **Concepts Covered**: Lists, Sets, Datetime, Scheduler, Notifications, AI.
- **Enhancements**:
  - Allow users to:
    - **Set deadlines and reminders**.
    - **Categorize tasks** into work, personal, urgent, etc.
    - **Prioritize** based on urgency and due date.
  - Use **AI (GPT or Chatbot API)** to:
    - Suggest the best **time slot** for tasks based on existing schedule.
    - Generate **summaries and motivational messages**.
  - Integrate **notifications (Desktop Alerts or SMS via Twilio)**.
- **Extensions**:
  - Add a **progress tracker**.
  - Export tasks to a **Google Calendar or CSV file**.

---

4. Math Quiz Game (Using Control Flow & Functions)

- **Concepts Covered**: Loops, If-Else, Functions, Random Module.
- **Description**:
  - Create a simple math quiz game.
  - Generate random arithmetic questions (addition, subtraction, multiplication).
  - Take user input and validate the answer.
  - Keep track of correct/incorrect responses and display the final score.
- **Extensions**:
  - Add different difficulty levels (easy, medium, hard).
  - Use a timer to limit answer time.

Math Quiz with AI-Generated Questions & Adaptive Difficulty

- **Concepts Covered**: Randomization, Control Flow, File Handling, AI APIs.
- **Enhancements**:
  - Generate **math problems dynamically** based on difficulty levels:
    - **Easy**: Basic arithmetic.
    - **Medium**: Algebra, fractions.
    - **Hard**: Calculus, trigonometry.
  - Implement:
    - **Timer-based scoring system**.
    - **Leaderboard** (save high scores in a file).
    - **AI-generated word problems** using OpenAI API.
  - Adaptive difficulty: If the user **gets 3 correct answers in a row**, increase the difficulty.
- **Extensions**:
  - Multiplayer mode (two users can compete).
  - Speech-to-text input for solving problems.

---

5. Simple Banking System (Using Functions & Dictionaries)

- **Concepts Covered**: Functions, Dictionaries, File Handling.
- **Description**:
  - Implement basic banking operations:
    - Create an account.
    - Deposit money.
    - Withdraw money.
    - Check balance.
  - Store account information in a **dictionary** (e.g., `{account_number: balance}`).
  - Save and load account details from a file.

- **Extensions**:
    - Implement user authentication (simple username-password check).

Secure Banking System with Multi-Level Authentication

- **Concepts Covered**: Functions, Dictionaries, File Handling, Encryption.
- **Enhancements**:
    - Users can:
        - **Register with a username, password, and 4-digit PIN**.
        - **Login securely** (password hashing using `bcrypt`).
        - **Perform transactions** (deposit, withdraw, transfer money).
        - **Generate monthly statements** in **CSV format**.
    - Implement:
        - **2FA (Two-Factor Authentication)** (send OTP to email).
        - **Encryption for sensitive data** using `cryptography` library.
    - Store user details securely in a **database (SQLite/MySQL)** instead of a plain file.
- **Extensions**:
    - Add **currency conversion** using an API.
    - Implement a **simple chatbot assistant** for FAQs.