Python Programming Assignments

These assignments are based on a full Python course covering the following topics:

- Print Statements, Comments, Escape Sequences
- Variables, Datatypes, Typecasting
- Taking Input, Strings, String Methods
- Lists, Tuples, Dictionaries, Sets
- Mutable vs Immutable Concepts
- If-Else, Loops (For, While), Operators
- Functions, Recursion
- Try-Except Exception Handling
- File System Handling
- Modules and Packages
- Object-Oriented Programming (Classes, Objects, Inheritance, etc.)
- GUI Programming (e.g., using Tkinter)

Below is a collection of **20+ assignments** arranged from beginner to advanced level. Each assignment also includes **methodology hints** and **expected console behavior** (for terminal-based tasks) to guide the student.

Beginner Level Assignments

1. Student Report Card Generator

Methodology: Use dictionary to store subjects and marks, use functions for calculation, and file I/O to save results. **Expected Console Behavior:** Prompt for 5 subject marks, show total, percentage, and grade. Display formatted report and save to a file.

2. ATM Simulator

Methodology: Use a loop-based menu system, file I/O for balance and history, and functions for modularity. **Expected Console Behavior:** Show menu with options like Deposit, Withdraw, Balance. Handle incorrect options and insufficient balance.

3. Contact Book

Methodology: Store contacts in a dictionary, use functions for each operation, and save/load using file. **Expected Console Behavior:** Menu-driven CLI showing options like Add, Delete, Search. Confirm actions and handle invalid input.

4. Word Frequency Counter

Methodology: Read file, use string methods and dictionary to count words, sort and display top results. **Expected Console Behavior:** Prompt user for filename, show most frequent words with counts.

5. Rock-Paper-Scissors Game

Methodology: Use random module, loop for multiple rounds, and conditionals for win/lose logic. **Expected Console Behavior:** Prompt user for R/P/S, show computer's choice, declare round result, display score at end.

6. Even-Odd Range Splitter

Methodology: Use range() and for loop, classify numbers into separate lists. **Expected Console Behavior:** Prompt for start and end, show even and odd lists separately.

7. Number Guessing Game

Methodology: Use while loop, random number generation, and if-else for hinting. **Expected Console Behavior:** Prompt for guess, respond with "Too High", "Too Low", or "Correct!", show attempts taken.

8. Basic Calculator

Methodology: Use input(), if-else, and functions for operations. **Expected Console Behavior:** Prompt for two numbers and an operator, show result.

9. Multiplication Table Generator

Methodology: Use a loop, string formatting, and optional file writing. **Expected Console Behavior:** Input a number, display its multiplication table (1–10).

10. Simple Interest and EMI Calculator

Methodology: Use formulae in functions, get input values from user. **Expected Console Behavior:** Input principal, rate, and time. Show calculated SI or EMI clearly.

Intermediate Level Assignments

11. To-Do List Manager

Methodology: Use list of dictionaries to store tasks, loop menu, and file for persistence. **Expected Console Behavior:** Menu with add/remove/mark complete. Display task list clearly.

12. Quiz App

Methodology: Store questions and answers in dictionary or list, loop through questions, use scoring. **Expected Console Behavior:** Show one question at a time, prompt for answer, show score at end.

13. Tic Tac Toe (2 Player Game)

Methodology: Use 2D list or simple 1–9 grid, turn-based logic, and win detection. **Expected Console Behavior:** Display grid after each turn, prompt for player input, show winner or draw.

14. Expense Tracker

Methodology: Store expenses in list of dictionaries, categorize by date/type, and file I/O for reports. **Expected Console Behavior:** Menu to add/view/summary, show expense breakdown.

15. File Organizer Script

Methodology: Use `os` module to list and move files by extension. **Expected Console Behavior:** Prompt for folder path, confirm file sorting, list of organized files.

16. CSV Data Reader & Analyzer

Methodology: Use `csv` module, loops and conditions to analyze values. **Expected Console Behavior:** Load and display summary like average, max, min, total for selected column.

17. Unit Converter

Methodology: Use function-based modular logic for each unit type. **Expected Console Behavior:** Menu to select conversion type, input value, display converted result.

18. Library Management System

Methodology: Use list of dictionaries or classes to store books, functions for borrow/return. **Expected Console Behavior:** Menu to add, issue, return, search books. Display book status.

Advanced Level Assignments

19. Banking System with OOP

Methodology: Create classes for Customer, Account, and Transaction. Encapsulation and file persistence. **Expected Console Behavior:** CLI menu with object-based operations. Display account status and transaction history.

20. Inventory Management System (OOP)

Methodology: Use classes for Product, Inventory, and Supplier. Implement add/update/remove features. **Expected Console Behavior:** Inventory dashboard, alerts for low stock.

21. Weather App using API

Methodology: Use `requests` module to call weather API, parse and display JSON data. **Expected Output:** GUI or terminal output with city name, temperature, condition, etc.

22. Password Manager (File + Encryption)

Methodology: Use file I/O for data, encrypt with external module, and decrypt on demand. **Expected Output:** GUI or CLI interface to add/view passwords securely.

23. Desktop GUI Calculator (Tkinter)

Methodology: Use Tkinter for layout, button events for input, and logic for calculations. **Expected Output:** Simple GUI with buttons for digits/operators and display area.

24. Digital Clock GUI

Methodology: Use `tkinter` and `time` module to show live clock. **Expected Output:** GUI showing current time updating every second.

25. Login/Register GUI App

Methodology: Use `tkinter`, file for storing credentials, and input validation. **Expected Output:** GUI with login/register screen, error handling, and session logic.

26. Chat App (Local GUI + Socket Programming)

Methodology: Use `socket` for communication, `tkinter` for GUI interface. **Expected Output:** Basic chat box where messages appear in real-time.

27. Mini Project: Notes App with GUI + File Save

Methodology: Use `tkinter` text widgets, file save/load functions. **Expected Output:** GUI with rich-text area, Save and Load buttons.

These assignments are designed to be progressive, starting from basic syntax to full application development using GUI and OOP principles. Each task includes guidance on how to approach the problem and what kind of console or GUI behavior is expected. This ensures clarity and direction for students when implementing their solutions.