**1. Number Guessing Game**

In this I created a number guessing game. This program first picks a random number between 1 and 100 using the random module(build in modle). Then it asks the user to guess the number and gives reply after each guess. If the guess is too low, it tells the user to try a high number. if it’s too high, it suggests going low. This continues until the user finally guesses the correct number, at which point the game congratulates them . I used a while loop to keep the game running and if-elif-else statements to compare the guesses.

**2. Calculator**

In this a calculator that can perform arithmetic operations like add, sub, multi, and division. The program asks the user to enter two numbers and choose an operator (+, -, \*, or /). Based on the operator, it performs the correct calculation and shows the result. I used a function called calculate() to handle the logic, which makes the code more organized. I also added input validation(try-except and error cases) to make sure the user enters valid numbers and to prevent division by zero. The program runs inside a loop, so after each calculation, the user can choose whether to do another one or exit.

**3. To-Do List Manager**

It allows users to add new tasks, view the list of tasks, mark tasks as done, and remove tasks. All the tasks are stored in a Python list as dictionaries, where each task has a description and a "done" status. The program displays a menu with five options and keeps running in a loop until the user chooses to exit. When a user selects an option, the corresponding action is performed, such as adding a task to the list or changing its status to "done”. In this basic CRUD operations is used .

**4. Temperature Converter**

In this program, I built a simple temperature converter using Python. The user is asked to enter a temperature followed by a unit (like 100C or 212F). The program reads the input, separates the numeric value and the unit, and then converts it to the other temperature scale using standard formulas. If the input is in Celsius, it converts to Fahrenheit using (C \* 9/5) + 32. If the input is in Fahrenheit, it converts to Celsius using (F - 32) \* 5/9. I used string slicing and .upper() to handle input properly and added basic input validation to make sure the user types a valid number and unit. This task helped me understand how to process user input, apply mathematical formulas, and format output clearly.

**5. Even or Odd Checker**

In this i created a simple tool to check whether numbers in a given range are even or odd. The user is asked to enter a start and end number. Then the uses is a for loop to go through each number in that range, including start and end. Inside the loop, it uses the modulo operator % to check if the number is divisible by 2. If it is, the number is even or it’s odd. The result is line by line for every number.

**6. Contact Book**

In this i created a contact book using a Python dictionary . This allows users to add new contacts, view all saved contacts, search for a contact by name, and delete a contact. Each contact is stored as a key-value pair in the dictionary, where the name is the key and the phone number is the value. I used separate functions for each operation to keep the code organized and menu-driven logic to make it user friendly. This run inside a loop until the user chooses to exit.

**7. Password Generator**

In this Python program, I built a simple password generator that creates strong and random passwords. The user is asked to enter the desired length of the password. I then use built-in character sets from the string module — including letters, numbers, and symbols — to create a pool of characters. The password is generated by randomly selecting characters from this pool using a list comprehension and the random.choice() function. Finally, the result is printed out. This task helped me understand how to use the random and string modules, and how to generate secure passwords dynamically using loops and string operations.

8. Word Frequency

This Python program counts how often each word appears in a paragraph entered by the user. It starts by taking the paragraph input, converting it to lowercase, and removing punctuation using regular expressions. Then, it splits the cleaned text into a list of words and loops through each word to count its frequency using a dictionary. Finally, the program prints each word along with how many times it occurred. This task helped me practice basic text processing, string manipulation, regular expressions, and dictionary usage in Python.

9. Quiz Game

This Python program is a multiple-choice quiz game that tests the user's knowledge with three simple questions. Each question is stored in a dictionary containing the question text, four options, and the correct answer. The program displays one question at a time, lets the user input their answer (a/b/c/d), and checks if it matches the correct option. It gives immediate feedback after each answer and keeps track of the user’s score. At the end of the quiz, it shows the total number of correct answers. This task helped me practice using lists, dictionaries, loops, functions, and control flow in Python to build an interactive and structured application.

10. Bank Account Simulation

This Python program simulates a basic bank account system. It starts by storing a predefined account with a username, PIN, and balance. The user must log in by entering the correct username and PIN. Once logged in, they are presented with a menu that allows them to check their balance, deposit money, withdraw money, or exit the program. Each function handles one banking operation and ensures that invalid or negative amounts are not allowed. It also prevents users from withdrawing more than their current balance. This task helped me understand how to use functions to organize code, handle nested conditions, and simulate real-world logic like authentication and balance tracking.