To do-list

Statement

A To-Do List application is a useful project that helps users manage and organize their tasks efficiently. This project aims to create a command-line or GUI-based application using Python, allowing users to create, update, and track their to-do lists.

Code:

# Simple To-Do List Application

# Define an empty list to store tasks

tasks = []

def show\_menu():

    print("\nTo-Do List Application")

    print("----------------------")

    print("1. Add a task")

    print("2. View all tasks")

    print("3. Update a task")

    print("4. Delete a task")

    print("5. Exit")

def add\_task():

    task = input("\nEnter the task: ")

    tasks.append(task)

    print(f"'{task}' has been added to the to-do list.")

def view\_tasks():

    if len(tasks) == 0:

        print("\nNo tasks in the list.")

    else:

        print("\nTo-Do List:")

        for i, task in enumerate(tasks, 1):

            print(f"{i}. {task}")

def update\_task():

    view\_tasks()

    if len(tasks) > 0:

        task\_num = int(input("\nEnter the number of the task you want to update: "))

        if 1 <= task\_num <= len(tasks):

            updated\_task = input("Enter the updated task: ")

            tasks[task\_num - 1] = updated\_task

            print(f"Task {task\_num} has been updated.")

        else:

            print("Invalid task number.")

def delete\_task():

    view\_tasks()

    if len(tasks) > 0:

        task\_num = int(input("\nEnter the number of the task you want to delete: "))

        if 1 <= task\_num <= len(tasks):

            deleted\_task = tasks.pop(task\_num - 1)

            print(f"Task '{deleted\_task}' has been deleted.")

        else:

            print("Invalid task number.")

def main():

    while True:

        show\_menu()

        choice = input("\nChoose an option (1-5): ")

        if choice == '1':

            add\_task()

        elif choice == '2':

            view\_tasks()

        elif choice == '3':

            update\_task()

        elif choice == '4':

            delete\_task()

        elif choice == '5':

            print("Exiting the application.")

            break

        else:

            print("Invalid option, please try again.")

if \_\_name\_\_ == "\_\_main\_\_":

    main()

Calculator

Statement:

Design a simple calculator with basic arithmetic operations. Prompt the user to input two numbers and an operation choice. Perform the calculation and display the result.

Code:

# Simple Calculator Program

def add(x, y):

    return x + y

def subtract(x, y):

    return x - y

def multiply(x, y):

    return x \* y

def divide(x, y):

    if y == 0:

        return "Error! Division by zero."

    else:

        return x / y

def calculator():

    print("Simple Calculator")

    print("------------------")

    print("Select operation:")

    print("1. Add")

    print("2. Subtract")

    print("3. Multiply")

    print("4. Divide")

    # Get user input

    choice = input("Enter choice (1/2/3/4): ")

    if choice in ['1', '2', '3', '4']:

        num1 = float(input("Enter first number: "))

        num2 = float(input("Enter second number: "))

        if choice == '1':

            print(f"{num1} + {num2} = {add(num1, num2)}")

        elif choice == '2':

            print(f"{num1} - {num2} = {subtract(num1, num2)}")

        elif choice == '3':

            print(f"{num1} \* {num2} = {multiply(num1, num2)}")

        elif choice == '4':

            print(f"{num1} / {num2} = {divide(num1, num2)}")

    else:

        print("Invalid Input. Please enter a valid choice.")

# Run the calculator function

Calculator()

Rock-Paper-Scissors Game

Statement:

User Input: Prompt the user to choose rock, paper, or scissors. Computer Selection: Generate a random choice (rock, paper, or scissors) for the computer. Game Logic: Determine the winner based on the user's choice and the computer's choice. Rock beats scissors, scissors beat paper, and paper beats rock. Display Result: Show the user's choice and the computer's choice. Display the result, whether the user wins, loses, or it's a tie. Score Tracking (Optional): Keep track of the user's and computer's scores for multiple rounds. Play Again: Ask the user if they want to play another round. User Interface: Design a user-friendly interface with clear instructions and feedback.

Code:

import random

# Function to get computer's choice

def get\_computer\_choice():

    choices = ['rock', 'paper', 'scissors']

    return random.choice(choices)

# Function to determine the winner

def determine\_winner(user\_choice, computer\_choice):

    if user\_choice == computer\_choice:

        return "tie"

    elif (user\_choice == 'rock' and computer\_choice == 'scissors') or \

         (user\_choice == 'scissors' and computer\_choice == 'paper') or \

         (user\_choice == 'paper' and computer\_choice == 'rock'):

        return "user"

    else:

        return "computer"

# Function to display result

def display\_result(user\_choice, computer\_choice, winner):

    print(f"\nYou chose: {user\_choice}")

    print(f"Computer chose: {computer\_choice}")

    if winner == "tie":

        print("It's a tie!")

    elif winner == "user":

        print("You win!")

    else:

        print("Computer wins!")

# Main function to play the game

def play\_game():

    user\_score = 0

    computer\_score = 0

    while True:

        print("\n--- Rock, Paper, Scissors ---")

        user\_choice = input("Enter your choice (rock/paper/scissors): ").lower()

        if user\_choice not in ['rock', 'paper', 'scissors']:

            print("Invalid choice! Please enter rock, paper, or scissors.")

            continue

        computer\_choice = get\_computer\_choice()

        winner = determine\_winner(user\_choice, computer\_choice)

        # Display the result of the round

        display\_result(user\_choice, computer\_choice, winner)

        # Update scores

        if winner == "user":

            user\_score += 1

        elif winner == "computer":

            computer\_score += 1

        # Show scores

        print(f"\nScore - You: {user\_score}, Computer: {computer\_score}")

        # Ask if the user wants to play again

        play\_again = input("Do you want to play again? (yes/no): ").lower()

        if play\_again != 'yes':

            print("\nThanks for playing!")

            print(f"Final Score - You: {user\_score}, Computer: {computer\_score}")

            break

# Start the game

play\_game()