# Python Projekte

## Projekt 1: Taschenrechner

Erstellen Sie ein Programm, das die grundlegenden Rechenoperationen durchführt und einen benutzerfreundlichen Taschenrechner simuliert.

### Anforderungen:

1. Zeigen Sie ein Menü mit den Optionen:
   * Addition
   * Subtraktion
   * Multiplikation
   * Division
   * Beenden
2. Der Benutzer wählt eine Option.
3. Fragen Sie den Benutzer nach zwei Zahlen.
4. Führen Sie die ausgewählte Operation durch und geben Sie das Ergebnis aus.
5. Fehlerbehandlung:
   * Verhindern Sie Division durch Null.
   * Fangen Sie ungültige Eingaben ab (z. B. Buchstaben statt Zahlen).
6. Wiederholen Sie das Menü, bis der Benutzer "Beenden" auswählt.

### Erweiterung:

* Implementieren Sie erweiterte Funktionen wie Potenzieren, Quadratwurzel und Prozentrechnung.
* Speichern Sie alle Berechnungen in einer Liste und zeigen Sie eine Historie an.

## Projekt 2: To-Do-Listen-Anwendung

Erstellen Sie ein Programm, mit dem der Benutzer Aufgaben verwalten kann. Die Anwendung sollte Aufgaben hinzufügen, anzeigen, bearbeiten und löschen können.

### Anforderungen:

1. Erstellen Sie ein Menü mit den Optionen:
   * Aufgabe hinzufügen
   * Alle Aufgaben anzeigen
   * Aufgabe bearbeiten
   * Aufgabe löschen
   * Beenden
2. Speichern Sie die Aufgaben in einer Liste.
3. Beim Bearbeiten oder Löschen einer Aufgabe:
   * Zeigen Sie alle Aufgaben mit Indexnummern an, damit der Benutzer die gewünschte Aufgabe auswählen kann.
4. Speichern Sie die Aufgaben persistent in einer Datei (todo.txt) und laden Sie sie beim Start des Programms.
5. Fangen Sie alle möglichen Fehler ab, z. B.:
   * Ungültige Eingaben.
   * Auswahl eines Indexes, der nicht existiert.

### Erweiterung:

* Fügen Sie ein Fälligkeitsdatum für Aufgaben hinzu.
* Implementieren Sie eine Suchfunktion, um Aufgaben anhand von Stichwörtern zu finden.

## Projekt 3: Quiz-Spiel

Erstellen Sie ein interaktives Quiz-Spiel, bei dem der Benutzer Fragen beantwortet und Punkte basierend auf den richtigen Antworten erhält.

### Anforderungen:

1. Definieren Sie eine Liste von mindestens 10 Fragen, jede mit:
   * Einer Frage.
   * Vier Antwortmöglichkeiten.
   * Der richtigen Antwort.
2. Stellen Sie dem Benutzer nacheinander Fragen.
3. Lassen Sie den Benutzer eine Antwort auswählen.
4. Geben Sie Feedback, ob die Antwort richtig oder falsch war.
5. Am Ende zeigen Sie die Gesamtpunktzahl an.

### Erweiterung:

* Speichern Sie Highscores in einer Datei und zeigen Sie die Bestenliste an.
* Fügen Sie Schwierigkeitsgrade hinzu (z. B. leichte, mittlere und schwere Fragen).

## Projekt 4: Mini-Bibliotheksverwaltung

Erstellen Sie ein Programm, mit dem Bücher in einer Bibliothek verwaltet werden können.

### Anforderungen**:**

1. Erstellen Sie ein Menü mit folgenden Funktionen:
   * Buch hinzufügen.
   * Buch ausleihen.
   * Buch zurückgeben.
   * Alle verfügbaren Bücher anzeigen.
   * Alle ausgeliehenen Bücher anzeigen.
   * Beenden.
2. Speichern Sie die Bücher in einer Liste oder einem Dictionary.
3. Beim Hinzufügen eines Buches:
   * Geben Sie den Titel und den Autor an.
4. Beim Ausleihen eines Buches:
   * Überprüfen Sie, ob das Buch verfügbar ist.
   * Verschieben Sie das Buch in eine separate Liste für ausgeliehene Bücher.
5. Fehlerbehandlung:
   * Fangen Sie ungültige Eingaben ab.
   * Zeigen Sie eine Fehlermeldung, wenn ein nicht verfügbares Buch ausgeliehen werden soll.

### Erweiterung:

* Speichern Sie die Buchdaten persistent in einer Datei.
* Implementieren Sie eine Suchfunktion, um Bücher nach Titel oder Autor zu finden.

## Projekt 5: Passwort-Manager

Erstellen Sie einen Passwort-Manager, der Anmeldedaten für verschiedene Dienste sicher speichert.

### Anforderungen:

1. Zeigen Sie ein Menü mit den Optionen:
   * Passwort speichern.
   * Passwörter anzeigen.
   * Passwort suchen.
   * Beenden.
2. Speichern Sie Anmeldedaten (Dienst, Benutzername, Passwort) in einer Datei (passwoerter.txt).
3. Beim Speichern eines Passworts:
   * Überprüfen Sie, ob der Dienst bereits existiert.
4. Beim Anzeigen der Passwörter:
   * Zeigen Sie alle gespeicherten Anmeldedaten an.
5. Beim Suchen eines Passworts:
   * Finden Sie den entsprechenden Dienst und geben Sie die Daten aus.

### Erweiterung:

* Verschlüsseln Sie die gespeicherten Passwörter mit einer Bibliothek wie cryptography oder hashlib.
* Implementieren Sie eine Passwortgeneratorfunktion.

## Projekt 6: Hangman-Spiel

Erstellen Sie ein Text-basiertes Hangman-Spiel, bei dem der Benutzer ein Wort erraten muss.

### Anforderungen**:**

1. Definieren Sie eine Liste von Wörtern, aus denen ein Wort zufällig ausgewählt wird.
2. Zeigen Sie dem Benutzer eine Reihe von Unterstrichen an, die die Buchstaben des Wortes repräsentieren.
3. Lassen Sie den Benutzer Buchstaben raten.
4. Aktualisieren Sie den Stand nach jedem Versuch:
   * Fügen Sie den Buchstaben hinzu, wenn er korrekt ist.
   * Ziehen Sie einen Versuch ab, wenn er falsch ist.
5. Zeigen Sie nach jedem Versuch die aktuelle Darstellung des Wortes und die verbleibenden Versuche an.
6. Das Spiel endet, wenn:
   * Der Benutzer das Wort errät (gewonnen).
   * Die Versuche aufgebraucht sind (verloren).

### Erweiterung:

* Speichern Sie eine Bestenliste mit der Anzahl der Versuche in einer Datei.
* Fügen Sie einen Mehrspielermodus hinzu.

## Projekt 7: Wetter-Statistik-Tool

Erstellen Sie ein Programm, das Wetterdaten analysiert und Statistiken berechnet.

### Anforderungen:

1. Laden Sie Wetterdaten (z. B. Temperatur, Niederschlag, Windgeschwindigkeit) aus einer Datei (wetterdaten.csv).
2. Zeigen Sie ein Menü mit folgenden Optionen:
   * Durchschnittstemperatur berechnen.
   * Höchste und niedrigste Temperatur anzeigen.
   * Tage mit Niederschlag anzeigen.
   * Wetterdaten für einen bestimmten Tag anzeigen.
   * Beenden.
3. Implementieren Sie Funktionen für jede Option.
4. Fehlerbehandlung:
   * Fangen Sie ungültige Eingaben ab.
   * Zeigen Sie eine Fehlermeldung, wenn ein nicht existierender Tag abgefragt wird.

### Erweiterung:

* Visualisieren Sie die Wetterdaten mit einer Bibliothek wie matplotlib.
* Erstellen Sie Vorhersagen basierend auf den Durchschnittsdaten.

## Projekt 8: Schere-Stein-Papier

Erstellen Sie ein interaktives Spiel "Schere-Stein-Papier".

### Anforderungen:

1. Der Benutzer wählt Schere, Stein oder Papier.
2. Der Computer trifft eine zufällige Wahl.
3. Bestimmen Sie den Gewinner anhand der Regeln:
   * Schere schlägt Papier.
   * Papier schlägt Stein.
   * Stein schlägt Schere.
4. Zeigen Sie das Ergebnis und aktualisieren Sie die Punkte für den Benutzer und den Computer.
5. Lassen Sie den Benutzer entscheiden, ob er eine weitere Runde spielen möchte.

### Erweiterung:

* Implementieren Sie eine Statistikfunktion, die die Siege, Niederlagen und Unentschieden anzeigt.
* Fügen Sie eine Mehrspielerversion hinzu.