

Übungen zu GUI

15. Der Algorithmus **ZINZESZINS** berechnet nach Eingabe des Anfangskapitals **K₀** (oder: des Preisindex **K₀** zu Anfang), des Zinsfußes **p** (oder: der Inflationsrate in Prozent) und der Laufzeit **n** (in Jahren) das Endkapital **K_n** (oder: den Preisindex **K_n**) nach **n** Jahren gemäß folgender Formel:

$$K_n = K_0 \cdot (1 + p/100)^n$$

Erstelle einen in Python geschriebenen Quelltext mit grafischer Benutzeroberfläche (**graphical user interface, GUI**), so daß die Eingabe der Daten und die Anzeige des Ergebnisses sich z. B. mit einer Benutzeroberfläche wie unten dargestellt gestalten.

Hinweise:

- Die Variablen **K_n**, **K₀** und **p** sind vom Typ **float**, **n** wahlweise **int** oder **float**.
- Falls das Ergebnis **kn** eine Gleitkommazahl (also vom Typ **float**) ist, rundet die Python-Anweisung **round(kn,2)** kaufmännisch auf zwei Nachkommastellen.
- Mit **str(kn)** wird die float-Zahl **kn** in eine Zeichenkette (vom Typ **string**) umgewandelt, um diese als Text im Label **ausgabe** anzeigen zu können.

Quelltext in Python als mögliche Lösung:

```
# Zinseszins

from tkinter import *

def berechnen():
    k0 = float(entry1.get())
    p = float(entry2.get())
    n = int(entry3.get())
    kn = k0 * (1 + p/100)**n
    kn = round(kn,2)
    ausgabe.config(text = 'Endkapital: ' + str(kn))

fenster = Tk()
fenster.title('Zinseszins')
fenster.geometry('500x300')
# fenster.resizable(True,0)

ausgabe = Label(fenster,bg='lightgrey', fg='purple',
                 text='Endkapital', font=('Arial', 30))
ausgabe.place(x=10, y=200, width=400, height=50)

label1 = Label(fenster, bg='lightgrey',
               fg='purple', text='Anfangskapital', font=('Arial', 18))
label1.place(x=10, y=10, width=180, height=50)

label2 = Label(fenster, bg='lightgrey',
               fg='purple', text='Zinsfuß', font=("Arial", 18))
label2.place(x=200, y=10, width=100, height=50)

label3 = Label(fenster, bg='lightgrey',
               fg='purple', text='Laufzeit', font=("Arial", 18))
label3.place(x=310, y=10, width=100, height=50)

entry1 = Entry(fenster, bg='red', font=("Arial", 30))
entry1.place(x=10, y=70, width=180, height=50)

entry2 = Entry(fenster, bg='orange', font=("Arial", 30))
entry2.place(x=200, y=70, width=100, height=50)

entry3 = Entry(fenster, bg='yellow', font=("Arial", 30))
entry3.place(x=310, y=70, width=100, height=50)

button = Button(fenster, bg = 'green', text = '=',
                font = ("Arial", 30), command = berechnen)
button.place(x=10, y=130, width=100, height=50)
```

Benutzeroberfläche:

The screenshot shows a web interface for calculating compound interest. It has three input fields: 'Anfangskapital' (1000), 'Zinsfuß' (4.3), and 'Laufzeit' (7). A green '=' button is below them. The result 'Endkapital: 1342.73' is displayed at the bottom.

Nach Eingabe der Werte für Anfangskapital, Zinsfuß und Laufzeit wird durch Anklicken des grünen Buttons „ = “ das Endkapital berechnet und angezeigt.

16. Der **Anhalteweg eines Fahrzeugs** ergibt sich als Summe von Reaktionsweg und Bremsweg:

Anhalteweg = Reaktionsweg + Bremsweg

Mit den Vereinbarungen

v_0 = Geschwindigkeit, aus der der Bremsvorgang eingeleitet wird (in m/s)

a = Bremsverzögerung (in m/s^2)

t_R = Reaktionszeit (in s)

gilt:

Reaktionsweg = $v_0 \cdot t_R$

Bremsweg = $\frac{v_0^2}{2 \cdot a}$

Schreibe ein Python-Programm mit grafischer Benutzeroberfläche GUI für den Algorithmus **ANHALTEWEG**, der nach Eingabe

- der Geschwindigkeit v_0 (in km/h; beachte: $1 \text{ m/s} = 3,6 \text{ km/h}$),
- der Reaktionszeit t_R (in s),
- des Straßenzustands

den Anhalteweg bei einer Gefahrenbremsung berechnet und ausgibt. Dabei ist die mögliche Bremsverzögerung a ein Maß für den Straßenzustand.

Annahmen:

trockene Fahrbahn: $a = g = 9,81 \text{ m/s}^2$ (oder näherungsweise 10 m/s^2)

regennasse Fahrbahn: $a = 0,7 g$

schneebedeckte Fahrbahn: $a = 0,2 g$

vereiste Fahrbahn: $a = 0,05 g$

17. Der Algorithmus **GRUNDRECHENARTEN** führt die „höheren“ Rechenarten Multiplizieren, Potenzieren und Dividieren durch geeignete Iteration auf die Grundrechenarten Addieren und Subtrahieren zurück.

Ergänze den in Python geschriebenen Quelltext Grundrechenarten.py durch eine graphische Benutzeroberfläche, so daß nach Eingabe der Operanden und nach Anklicken eines Buttons für die gewünschte Rechenoperation das Ergebnis berechnet und ausgegeben wird.