



RECHENTRAINER DOKUMENTATION

Featureliste, Ablaufbeispiele mit Belegungstabellen, Quellcode, Struktogramme



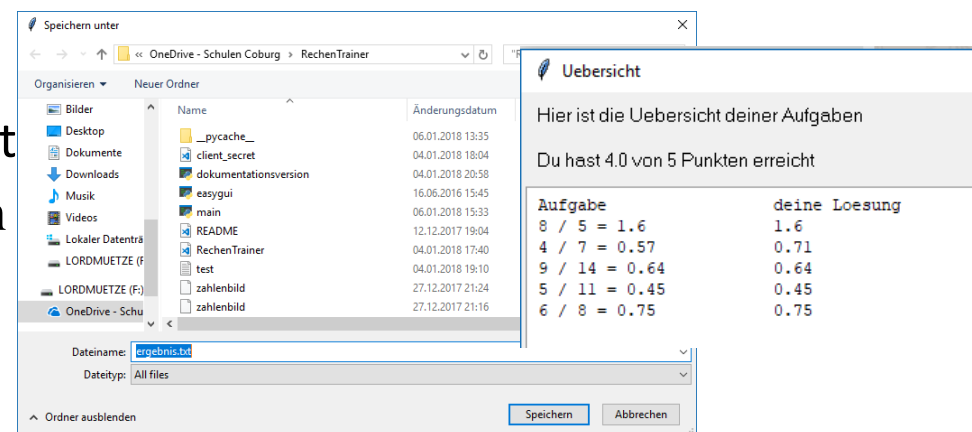
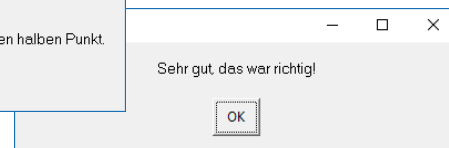
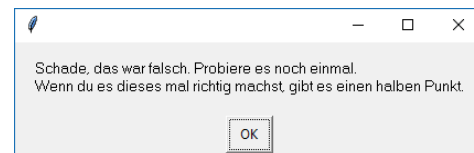
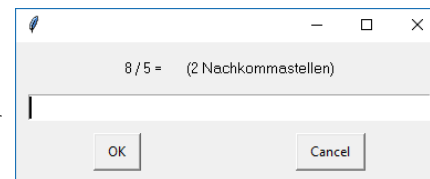
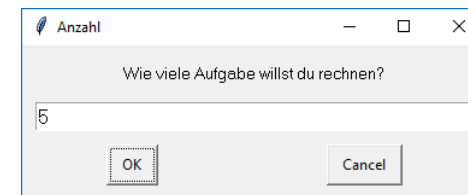
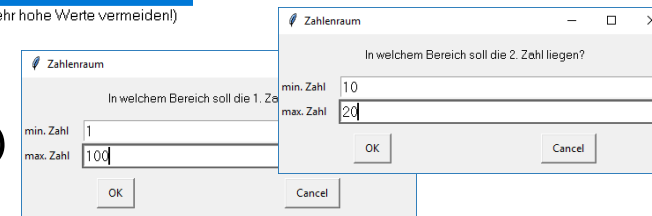
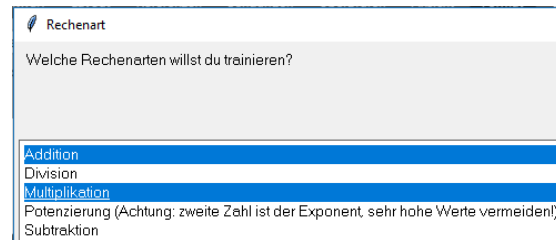
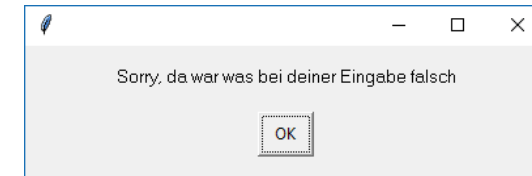
12/2017-01/2018

VON VALENTIN HERRMANN

Erstellt im Rahmen des Informatik-Unterricht
am Gymnasium Ernestinum Coburg bei Dr. Bernd Jakob

Features

- Startseite mit Start- & Verlassenschaftfläche und Titelbild
- Automatisches Prüfen jeder Eingabe auf ungültige Eingaben (bspw. Text statt Zahl)
- 5 Rechenarten in beliebiger Kombination wählbar
 - (Additieren, Subtrahieren, Multiplizieren, Dividieren, Potenzieren)
 - Warnung vor zu großen Zahlen beim Potenzieren
- Freie Wahl beider Komponenten der Rechnungen (natürliche Zahlen)
- Wahl der Anzahl der zu rechnenden Aufgaben (ausreichend hohes Limit)
- Angabe aller Ergebnisse auf 2 Nachkommastellen
- 2 Versuche pro Aufgabe
 - Ein Punkt im ersten Versuch
 - Ein halber Punkt im zweiten Versuch
 - Direktes Feedback
- Übersicht aller Aufgaben mit Nut
- Speichern der Übersicht möglich
- Direkte Rückkehr zur Startseite von überall mittels Cancel-Schaltfläche
- Verlassen ausschließlich durch Verlassen-Schaltfläche auf der Startseite möglich

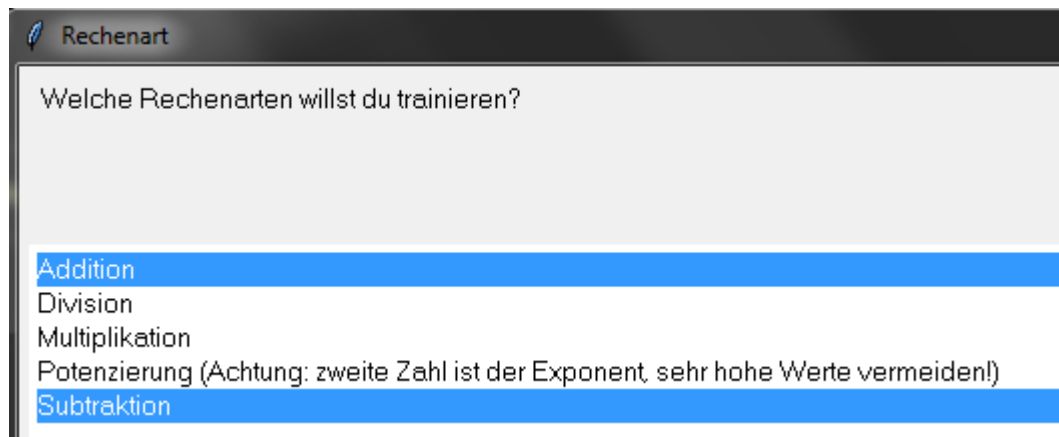


Ablaufbeispiele mit Belegungstabellen

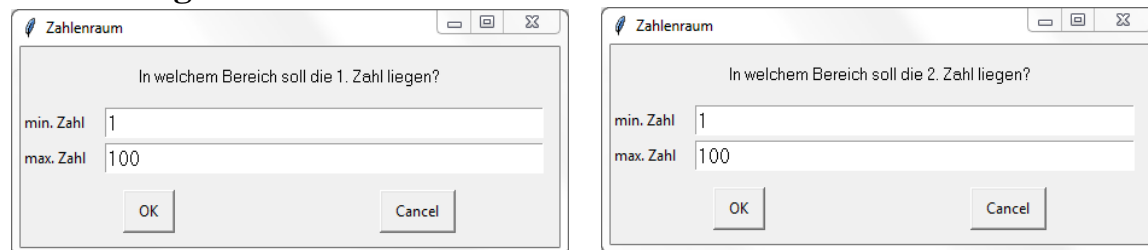
1. Johann öffnet den Rechentruainer und startet eine neue Runde



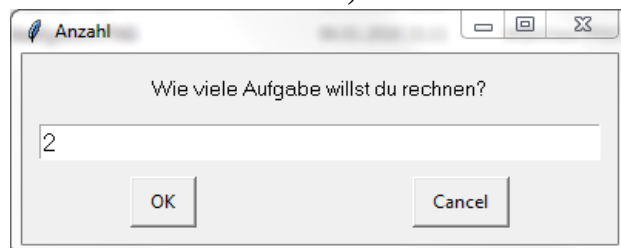
2. Da in seiner nächsten Arbeit Addition & Subtraktion bis 100 der Schwerpunkt sein werden wählt er diese beiden Rechenarten aus.



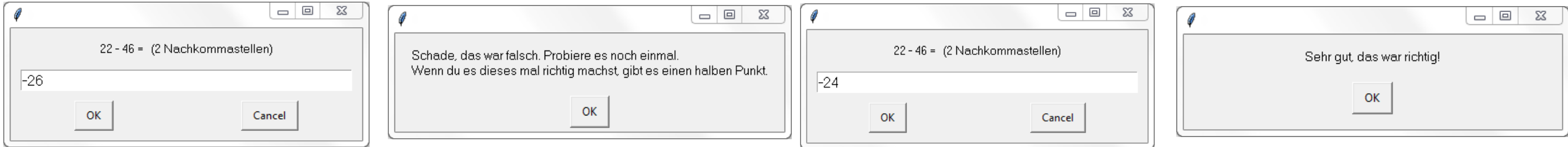
3. Danach legt er die Bereiche für beide Zahlen auf 1-100 fest.



4. Er hat nicht viel Zeit, daher möchte er nur 2 Aufgaben rechnen.



5. Seine erste Aufgabe (22-46) löst er zunächst falsch, im zweiten Versuch gibt er das richtige Ergebnis an. Er erhält einen halben Punkt.



Um zu prüfen, wie viele Versuche Johann noch hat, wird eine Schleife mit Eingangsbedingung gewählt, deren Zählervariable *i* kleiner als 2 sein muss um ausgeführt zu werden.

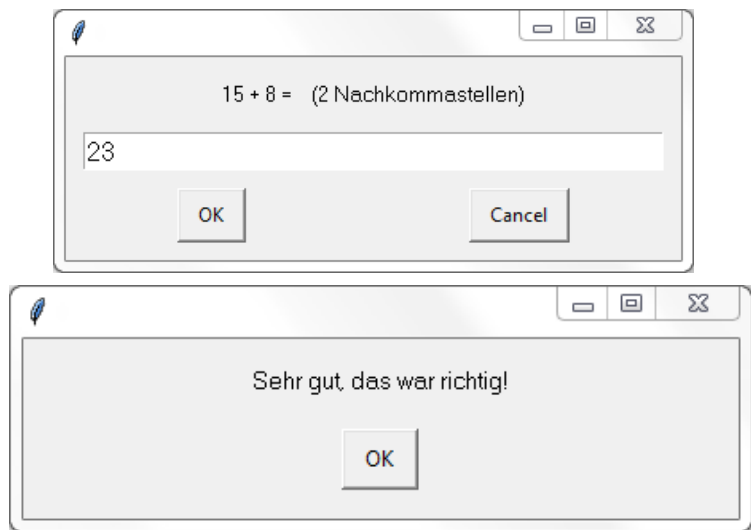
Ist die eingegebene Lösung falsch, wird *i* um 1 erhöht.

1. Bei zwei falschen Versuchen ist *i* also 2 und damit nicht kleiner als 2.
2. Bei einem richtigen Versuch wird *i* um 2 erhöht und ist somit direkt nicht mehr kleiner als 2.
3. Bei einem falschen und einem richtigen Versuch ist *i* 3 und somit auch nicht kleiner als 2.

In dieser Schleife befindet sich das Herzstück des Programms, die Anzeige der Aufgabe und die Eingabe des Ergebnisses.

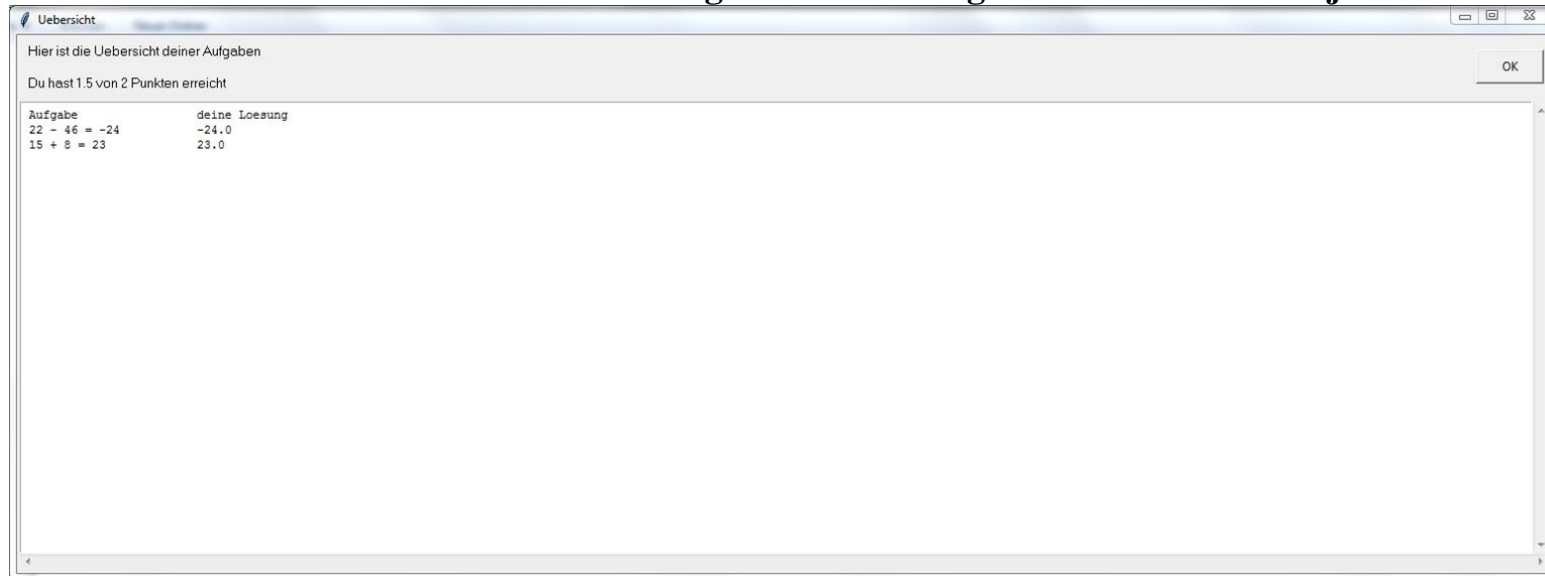
Zu Beginn wird die Punktzahl (*richtige_loesungen*) immer um 1 erhöht. Bei jeder falschen Eingabe wird ein halber Punkt abgezogen, sodass bei zwei falschen Ergebnissen 0 Punkte zur Gesamtpunktzahl addiert werden.

6. Die nächste Aufgabe (15+8) macht er auf Anhieb richtig und erhält einen Punkt. Die unten stehende Belegungstabelle soll die Fälle 5 und 6 verdeutlichen.

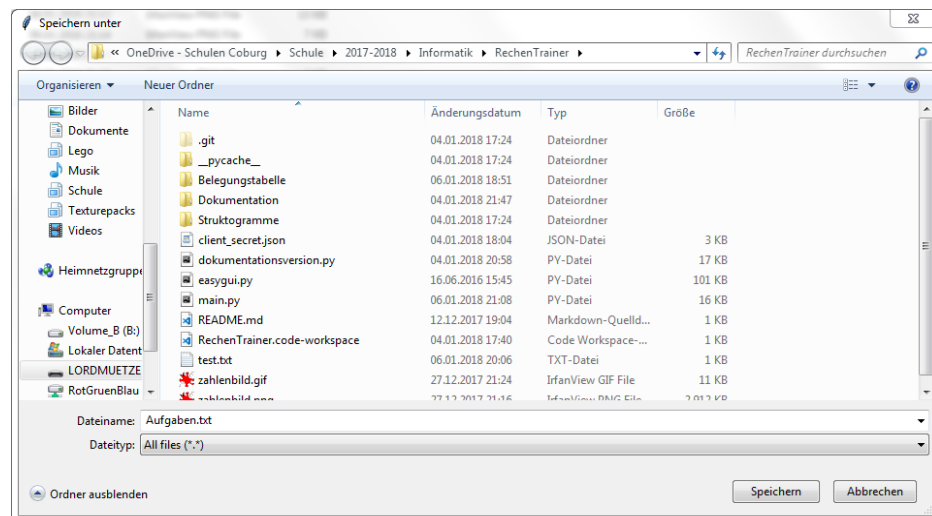


Zeile	i	i < 2	richtige_loesungen	Ausgabe	Eingabe
206			1		
215	0				
217		WAHR			
224				22 - 46 = (2 Nachkommastellen)	
224					-26
250				Schade, das war falsch. [...]	
251			0,5		
252	1				
217		WAHR			
224				22 - 46 = (2 Nachkommastellen)	
224					-24
244				Sehr gut, das war richtig!	
245	3				
217		FALSCH			
206			1,5		
215	0				
217		WAHR			
224				15 + 8 = (2 Nachkommastellen)	
224					23
244				Sehr gut, das war richtig!	
245	2				
217		FALSCH			

7. Nun bekommt er eine Übersicht der soeben gerechneten Aufgaben inklusive seiner jeweils letzten Lösung angezeigt. Außerdem wird seine erreichte Punktzahl (1,5 von 2) ausgegeben.



8. Danach speichert er diese Übersicht als Aufgaben.txt ab und kehrt zur Startseite zurück.



9. Als letztes klickt er die Verlassen-Schaltfläche an und beendet so das Programm.

Während des gesamten Programmablaufs wird kontrolliert, ob Johann irgendwann eine Cancellschaltfläche anklickt. Sollte dies der Fall sein, ist der Funktionswert der jeweiligen Funktion ein `NoneType`, besitzt also keinen Wert. Beim Gesamtablauf, wird bei jeder aufgerufenen Funktion mittels einer Variable namens `Funktionbezeichner_testen` geprüft, ob diese einen `NoneType` ausgibt. Sollte dies der Fall sein, so wird die Aufgabenschleife abgebrochen und zur Startseite zurück gekehrt. In diesem Beispiel klickt Johann keine Cancellschaltfläche an und `Funktionbezeichner_testen == None` ist somit immer `FALSCH`.

[illegible]

Sehen wir uns nun an, was passiert wäre und wie sich die obigen Belegungstabellen ändern, wenn Johann

1. Keine Rechenart ausgewählt hätte:

Rechenart

Welche Rechenarten willst du trainieren?

Addition

Division

Multiplikation

Potenzierung (Achtung: zweite Zahl ist der Exponent, sehr hohe Werte vermeiden!)

Subtraktion

Rechenart

Welche Rechenarten willst du trainieren?

Addition

Division

Multiplikation

Potenzierung (Achtung: zweite Zahl ist der Exponent, sehr hohe Werte vermeiden!)

Subtraktion

Das Auswahlfenster für die Rechenart öffnet sich erneut mit Standardauswahl (Addition).

2. Einen kleineren Maximalwert als Mindestwert eingegeben hätte:

Zahlenraum

In welchem Bereich soll die 1. Zahl liegen?

min. Zahl

10

max. Zahl

1

OK

Cancel

Sorry, da war was bei deiner Eingabe falsch

OK

Zahlenraum

In welchem Bereich soll die 1. Zahl liegen?

min. Zahl

1

max. Zahl

100

OK

Cancel

Es öffnet sich ein Fenster mit einem Hinweis, dass die eingegebenen Zahlen nicht möglich sind oder ein anderer Fehler aufgetreten ist. Danach öffnet sich erneut das Eingabefenster mit den Standardwerten 1 & 100.

3. Negative Werte für den Zahlenraum eingegeben hätte:

Zahlenraum

In welchem Bereich soll die 1. Zahl liegen?

min. Zahl

-1

max. Zahl

-100

OK

Cancel

Sorry, da war was bei deiner Eingabe falsch

OK

Zahlenraum

In welchem Bereich soll die 1. Zahl liegen?

min. Zahl

1

max. Zahl

100

OK

Cancel

Hier öffnet sich das gleiche Fenster, wie in Fall 2 & danach wird ebenfalls das Eingabefenster mit Standardwerten aufgerufen.

4. Eine negative Anzahl an Aufgaben eingegeben hätte:

Anzahl

Wie viele Aufgabe willst du rechnen?

-5

OK

Cancel

Error

The value that you entered is less than the lower bound of 1.

OK

Anzahl

Wie viele Aufgabe willst du rechnen?

5

OK

Cancel

Ein Fenster mit einem Hinweis öffnet sich, dann wird zum Eingabefeld mit Standardwert 5 zurückgekehrt.

5. Eine zu große Anzahl an Aufgaben eingegeben hätte:

Anzahl

Wie viele Aufgabe willst du rechnen?

1000000000001

OK

Cancel

Error

The value that you entered is greater than the upper bound of 1000000000000.

OK

Anzahl

Wie viele Aufgabe willst du rechnen?

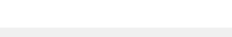

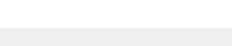
5

OK

Cancel

Es öffnet sich ebenfalls ein Hinweisfenster. Nun wird erneut das Eingabefeld mit dem Standardwert geöffnet.

6. Als Lösung für Aufgabe 1 aus Versehen Buchstaben eingegeben hätte:



Auch hier wird eine Warnung ausgegeben. Dieser Fehler wird auch verursacht, wenn bei einer Dezimalzahl ein Komma statt einem Punkt verwendet wird (z.B. 1,5 statt 1.5). Danach öffnet sich die Aufgabe erneut mit leerem Eingabefeld.

Zeile	i	i < 2	richtige_loesungen	Ausgabe	Eingabe
206			1		
215	0				
217		WAHR			
224				22 - 46 = (2 Nachkommastellen)	
224					abc
235				Sorry, da war etwas [...]	
224				22 - 46 = (2 Nachkommastellen)	
224					-26
...

7. Bei der zweiten Eingabeaufforderung von Aufgabe 1 Cancel gedrückt hätte:

22 - 46 = (2 Nachkommastellen)

OK Cancel

Johann landet ohne Umwege auf der Startseite und verlässt, wie im ersten Beispiel auch, die Programm.

Zeile	i	i < 2	richtige_loesungen	Ausgabe	Eingabe
206			1		
215	0				
217		WAHR			
224				22 - 46 = (2 Nachkommastellen)	
224					-26
250				Schade, das war falsch. [...]	
251			0,5		
252	1				
217		WAHR			
224				22 - 46 = (2 Nachkommastellen)	
224					None
60			0		

[illegible]

Quellcode & Struktogramme

Im folgenden, dunkel unterlegten Teil befindet sich der Programmcode des Rechentrainers. Im direkten Vergleich zum Originalcode (‘main.py’) enthält er einige formatbedingte Änderungen (Entfernen/Hinzufügen von Leerzeilen, verschieben von Kommentare, usw.), die sich aber nicht auf den Programmablauf auswirken.

Neben den einzelnen Funktionen finden sich zudem Struktogramme, welche die Struktur dieser Funktionen erläutern. Wo nötig wurden außerdem ergänzende Verbindungen zwischen Struktogramm und Code eingezeichnet. Außerdem werden try-except-Statements in den Struktogrammen nicht beachtet, da diese dann den Rahmen sprengen würden.

```
#-----Modulimport-----

from easygui import * #Grafische Oberfläche, Dokumentation: http://easygui.sourceforge.net
from random import * #Zufallsgenerator
import sys

# wird bei einer Box (Element der GUI) der Cancel-Button gedrückt, wird ein NoneType ausgegeben, was später zur Überprüfung genutzt wird, ob dieser Button gedrückt wurde
#-----initialisieren von globalen Variablen, die später im Programm verwendet werden-----
#-----benötigte globale Variablen werden mit "global Variablenname" in die einzelnen Funktionen importiert-----

richtige_loesungen = 0.0
aufgabenliste = []
nutzer_loesungsliste = []
rechenart = []
min_zahl1 = None
max_zahl1 = None
min_zahl2 = None
max_zahl2 = None
anzahl_aufgaben = 0

#-----

#-----Definition der Startseite-----
def startseite():
    global richtige_loesungen
    global aufgabenliste
    global nutzer_loesungsliste
    global rechenart
    global min_zahl1
    global max_zahl1
    global min_zahl2
    global max_zahl2
    global anzahl_aufgaben

#Reset der wichtigsten globalen Variablen & Listen mit jedem neuen Durchlauf des Rechen Trainers
    richtige_loesungen = 0.0
    aufgabenliste = []
    nutzer_loesungsliste = []
    rechenart = []
    min_zahl1 = None
    max_zahl1 = None
    min_zahl2 = None
    max_zahl2 = None
    anzahl_aufgaben = 0

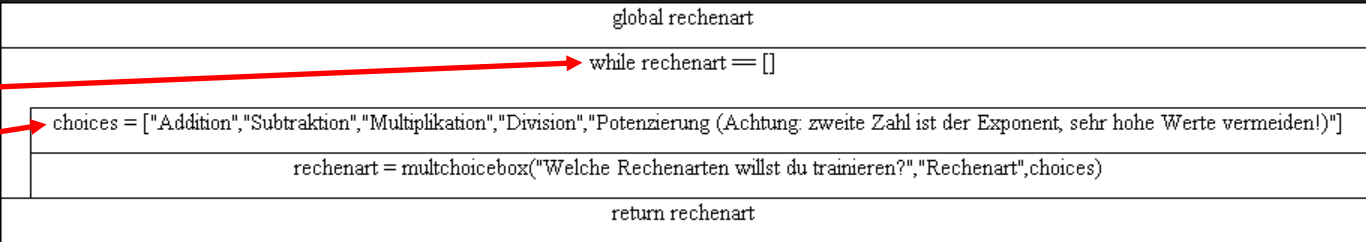
    choices = ["Los gehts", "Verlassen"] #Festlegen der Buttons
    start = buttonbox("Herzlich Willkommen beim Rechentrainer", "Willkommen", choices, "zahlenbild.gif")
    return start #angeklickten Button als Funktionswert ausgeben

#-----
```

global richtige_loesungen
global aufgabenliste
global nutzer_loesungsliste
global rechenart
global min_zahl1
global max_zahl1
global min_zahl2
global max_zahl2
global anzahl_aufgaben
richtige_loesungen = 0.0
aufgabenliste = []
nutzer_loesungsliste = []
rechenart = []
min_zahl1 = None
max_zahl1 = None
min_zahl2 = None
max_zahl2 = None
anzahl_aufgaben = 0
choices = ["Los gehts", "Verlassen"]
start = buttonbox("Herzlich Willkommen beim Rechentrainer", "Willkommen", choices, "zahlenbild.gif")
return start

startseite()-Funktion


```
#-----Definition, welche Rechenart trainiert werden soll-----
def rechenart_eingeben():
    global rechenart
    while rechenart == []: #Schleife läuft, solange keine Rechenart ausgewählt wurde
        #wählbare Rechenarten festlegen
        choices = ["Addition", "Subtraktion", "Multiplikation", "Division",
                  "Potenzierung (Achtung: zweite Zahl ist der Exponent, sehr hohe Werte vermeiden!)"]
        rechenart = multichoicebox("Welche Rechenarten willst du trainieren?", "Rechenart", choices) #Liste aus ausgewählte(n) Rechenart(en) erstellen
    return rechenart #ausgewählte Rechenarten [Liste] als Funktionswert zurückgeben
#-----
```



rechenart_eingeben()-Funktion

```
#-----Definition, welche Rechenart trainiert werden soll-----
def rechenart_abrufen():

    global rechenart

    rechentyp = SystemRandom().choice(rechenart) #zufälliges Rechenart von Liste auswählen

    #abhängig von der gewählten Rechenart "Rechnung(operator)" mit Operator
    #der jeweiligen Rechenart als Parameter aufrufen
    if rechentyp == "Addition":
        eingabe = Rechnung("+")

    elif rechentyp == "Subtraktion":
        eingabe = Rechnung("-")

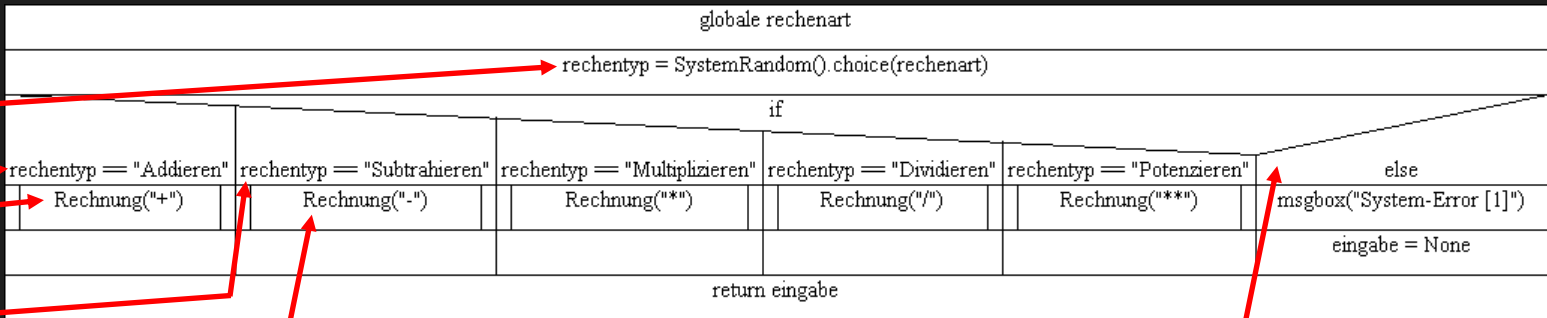
    elif rechentyp == "Multiplikation":
        eingabe = Rechnung("**")

    elif rechentyp == "Division":
        eingabe = Rechnung("/")

    elif rechentyp == "Potenzierung (Achtung: zweite Zahl ist der Exponent, sehr hohe Werte vermeiden!)":
        eingabe = Rechnung("**")

    #wird keine möglichen Rechenart ausgewählt, wird ein SystemError ausgegeben und "eingabe" als NoneType gesetzt
    else:
        msgbox("System-Error [1]")
        eingabe = None

    return eingabe #"eingabe" als Funktionswert ausgeben
#-----
```

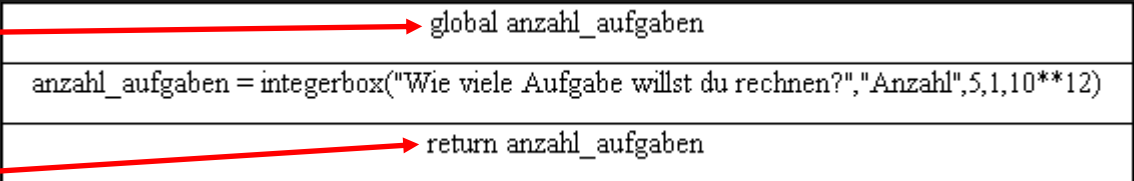


rechenart_abrufen()-Funktion

```
#-----Anzahl der Aufgaben-----
def anzahl_aufgaben_eingeben():
    global anzahl_aufgaben

    #Anzahl der zu rechnenden Aufgaben an "anzahl_aufgaben" binden, automatische Prüfung auf natürliche Zahl, maximal: 10**12
    anzahl_aufgaben = integerbox("Wie viele Aufgabe willst du rechnen?", "Anzahl", 5, 1, 10**12)

    return anzahl_aufgaben #Anzahl als Funktionswert ausgeben
#-----
```



anzahl_aufgaben_eingeben()-Funktion

```
#-----Festlegung des Zahlenraums-----
def zahlenraum():

    global min_zahl1
    global max_zahl1
    global min_zahl2
    global max_zahl2

    for i in range (1,3): #for-Schleife mit zwei Durchläufen: i=1 & i=2

        while True: #Endlosschleife

            try: #try-Teil wird ausgeführt, bis ein Fehler auftritt
                fields = ["min. Zahl","max. Zahl"] #Namen der Felder festlegen
                values = [1,100] #Standardwerte für die Felder festlegen

                zahlenraum = multenterbox("In welchem Bereich soll die "+str(i)+" . Zahl liegen?",
                                           "Zahlenraum",fields,values)

                if zahlenraum == None: #Cancel-Button wurde gedrückt
                    break #Endlosschleife abbrechen

            else:
                max_zahl = int(zahlenraum.pop())
                min_zahl = int(zahlenraum.pop())
                #letzten Listenwert als Minimal-/Maximalwert festlegen & von Liste entfernen
                #wird ein nicht-Integer (Float,Text,...) eingegeben, wird ein Fehler auftreten

                #prüfen, ob der Maximalwert größer als der Minimalwert ist
                #oder ob eine der beiden Zahlen kleiner als 0 ist (keine natürliche Zahl)
                if max_zahl < min_zahl or max_zahl < 0 or min_zahl < 0:
                    msgbox("Sorry, da war was bei deiner Eingabe falsch")

                else: #Zahlen sind ok
                    break #Endlosschleife abbrechen

            except: #tritt ein Fehler auf, so wird der except-Teil ausgeführt
                msgbox("Sorry, da war was bei deiner Eingabe falsch")

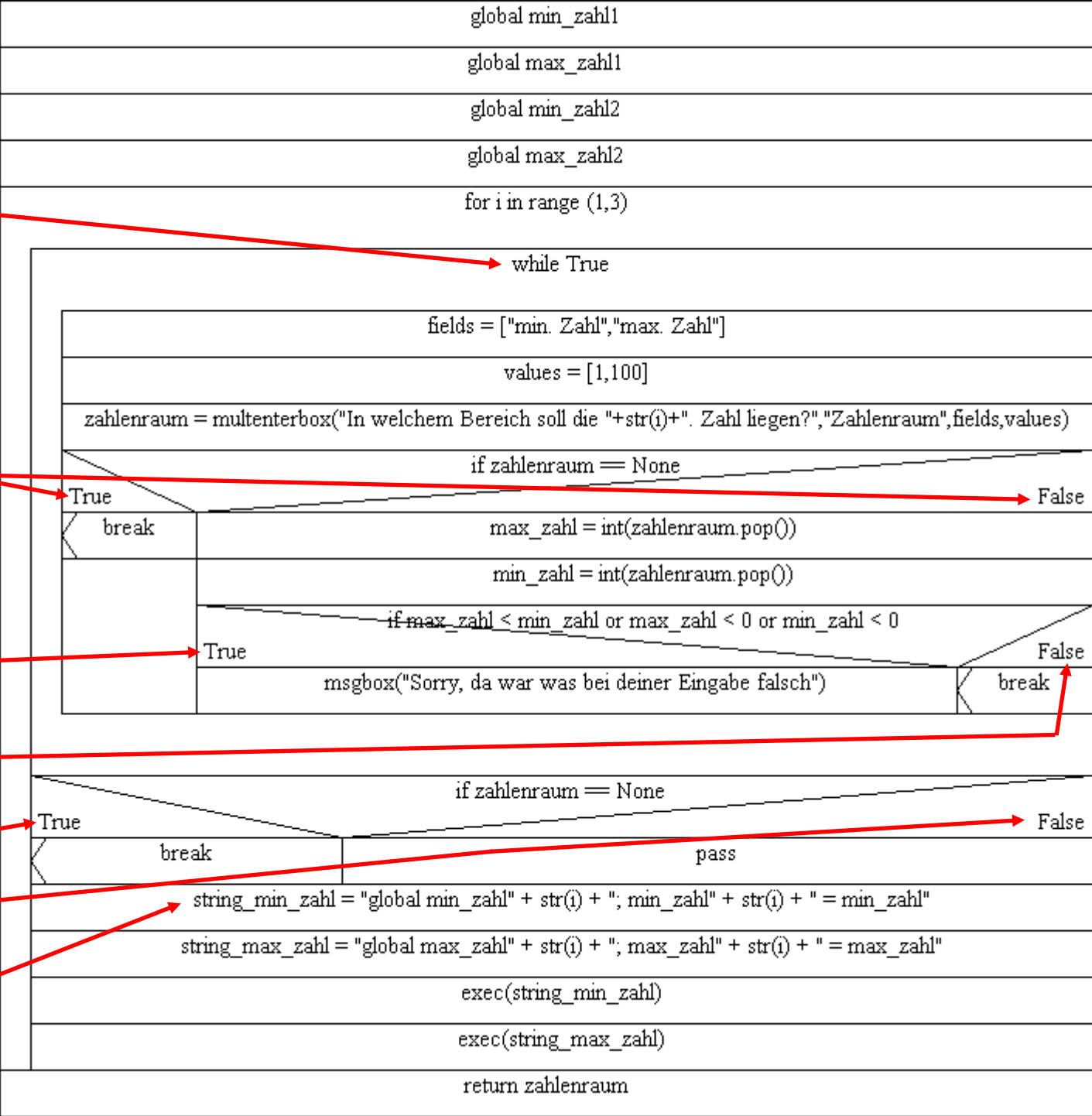
        if zahlenraum == None: #Cancel-Button wurde gedrückt
            break #for-Schleife abbrechen
        else:
            pass #nichts tun

    #Strings zum späteren Ausführen erstellen, Nummer von min_zahl & max_zahl abhängig von
    #Schleifendurchlauf
    string_min_zahl = "global min_zahl" + str(i) + "; min_zahl" + str(i) + " = min_zahl"
    string_max_zahl = "global max_zahl" + str(i) + "; max_zahl" + str(i) + " = max_zahl"

    #min_zahl & max_zahl, min_zahl1 & max_zahl1 bzw min_zahl2 & max_zahl2 zuweisen
    exec(string_min_zahl)
    exec(string_max_zahl)

    return zahlenraum #die Liste "zahlenraum" wird als Funktionswert ausgegeben

#-----
```



zahlenraum()-Funktion

#-----Aufgabenstellung mit Eingabe für die Lösung-----

```
def Rechnung(operator):
    global richtige_loesungen
    global aufgabenliste
    global nutzer_loesungsliste
    global min_zahl1
    global max_zahl1
    global min_zahl2
    global max_zahl2

    richtige_loesungen += 1 #Punktzahl um 1 erhöhen
    #erste Zahl für die Rechnung mit kleinster & größter Zahl zufällig festlegen
    #zweite Zahl für die Rechnung mit kleinster & größter Zahl zufällig festlegen
    nummer_1 = randint(min_zahl1, max_zahl1)    nummer_2 = randint(min_zahl2, max_zahl2)
    aufgabe = str(nummer_1) + " " + str(operator) + " " + str(nummer_2) #Aufgabe in der Form "1 + 1" erstellen
    ergebnis = round(eval(aufgabe),2) #zuvor erstellte Aufgabe auf 2 Nachkommastellen berechnen
```

```
    aufgabenloesung = aufgabe + " = " + str(ergebnis) #Aufgabe mit Lösung in der Form "1 + 1 = 2" zusammensetzen
    i = 0 #Zählervariable für Versuchs-Schleife initialisieren
    while i < 2: #Schleife mit Maximalwert i<2 für Versuche
        while True: #Endlosschleife
            try: #try-Teil wird ausgeführt, bis ein Fehler auftritt
                #Aufgabenstellung mit Eingabefeld wird angezeigt, Form "1 + 1 =  (2 Nachkommastellen)"
                eingabe = enterbox(aufgabe+" = \t(2 Nachkommastellen)")

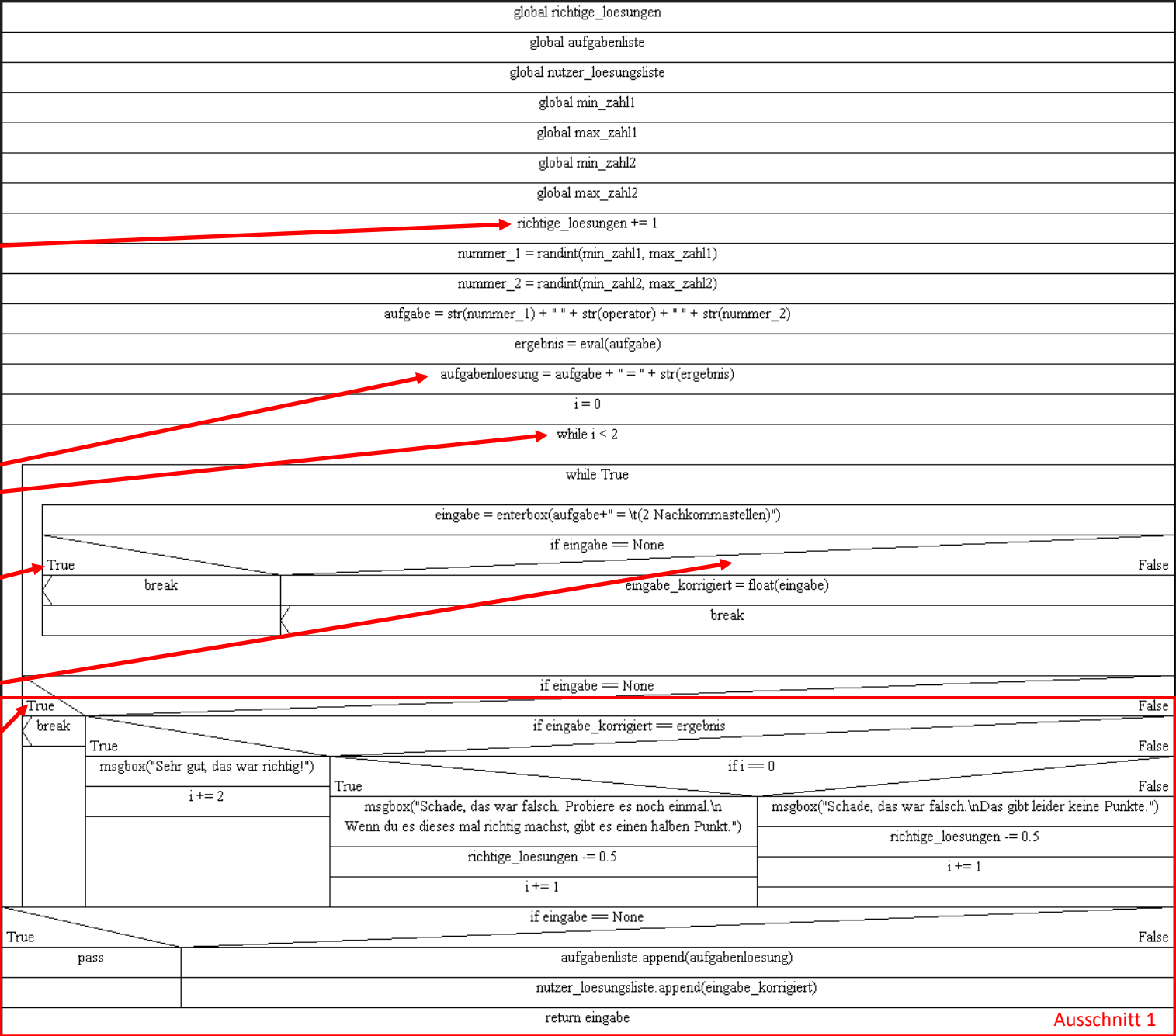
                if eingabe == None: #Cancel-Button wurde gedrückt
                    break #Endlosschleife wird unterbrochen
                else:
                    #Fehler wird ausgegeben, wenn sich die Eingabe nicht
                    #in einen Float mit 2 Nachkommastellen umwandeln lässt (zB Text)
                    eingabe_korrigiert = round(float(eingabe),2)

                    break #Endlosschleife wird unterbrochen
            #tritt ein Fehler auf, wird der except-Teil ausgeführt
            except:
                msgbox("""Sorry da was bei deiner Eingabe falsch\n
                    (z.B. falsche Zahlenart oder Komma statt Punkt bei Kommazahlen)""")
                #Zurückspringen zum Schleifenanfang

            if eingabe == None: #Cancel-Button wurde ggedrückt
                break #Abbruch der Schleife, welche die Anzahl der Versuche zählt und
                #in der die Aufgabenstellung angezeigt wird

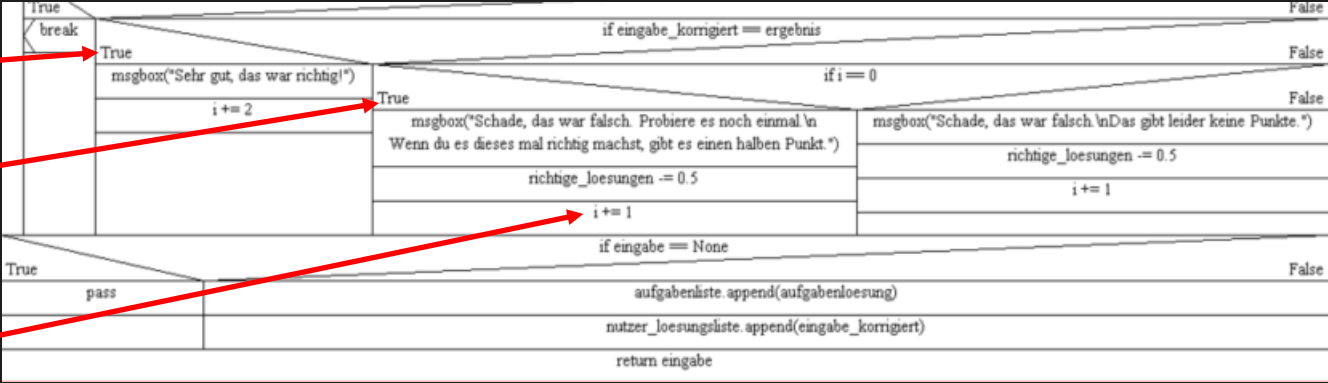
        else:
            if eingabe_korrigiert == ergebnis: #Eingabe ist richtig
                msgbox("Sehr gut, das war richtig!")
                i += 2 #Zähler der Schleife auf 2 erhöhen und dadurch die Schleife beenden

            else: #Eingabe ist falsch
                if i == 0: #prüfen, ob es der erste Versuch ist
                    msgbox("""Schade, das war falsch. Probiere es noch einmal.\n
                        Wenn du es dieses mal richtig machst,
                        gibt es einen halben Punkt.""")
                    richtige_loesungen -= 0.5 #halben Punkt abziehen =>0,5 Punkte
                    i += 1 #Zähler der Schleife um 1 (auf 1) erhöhen
```



Ausschnitt 1

rechnung()-Funktion



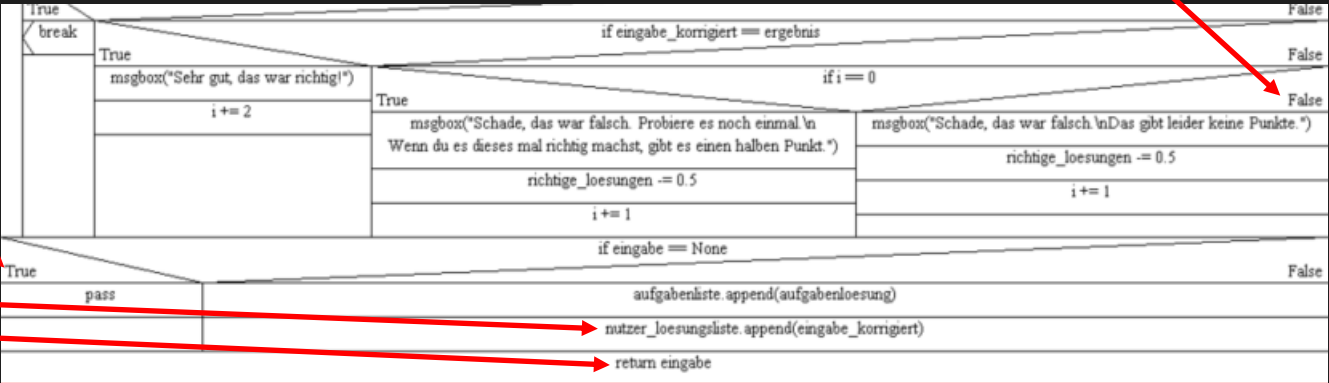
rechnung()-Funktion (Ausschnitt 1)

```

else: #zweiter (letzter) Versuch
    msgbox("Schade, das war falsch.\nDas gibt leider keine Punkte.")
    richtige_loesungen -= 0.5 #einen weiteren halben Punkt abziehen
    #--> 0 Punkte

    i += 1 #Zähler der Schleife um 1 (auf 2) erhöhen

if eingabe == None: #Cancel-Button wurde gedrückt
    pass #nichts tun
else:
    aufgabenliste.append(aufgabenloesung) #Aufgabe mit Lösung zur Liste für die Übersicht hinzufügen
    nutzer_loesungsliste.append(eingabe_korrigiert) #Lösung des Nutzers zur Liste für die Übersicht hinzufügen
return eingabe #"eingabe" als Funktionswert ausgegeben
#-----
```



rechnerung()-Funktion (Ausschnitt 1)

```

#-----erstellen der Abschlussuebersicht-----
def uebersicht():

    global aufgabenliste
    global nutzer_loesungsliste
    global anzahl_aufgaben
    globale richtige_loesungen
    uebersicht = "" #"uebersicht" als leeren String initialisieren

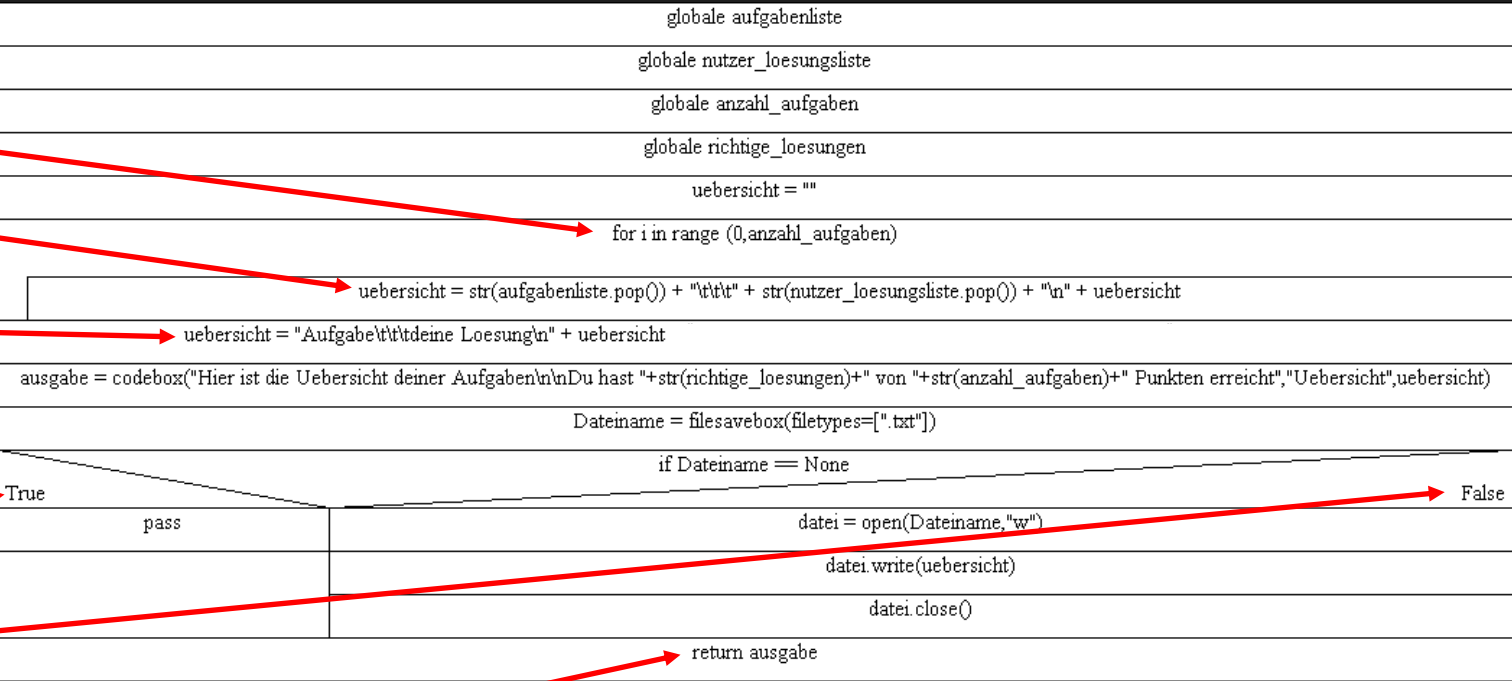
    for i in range (0,anzahl_aufgaben): #Schleife mit so vielen Durchläufen wie Aufgaben
        #bei jedem Durchlauf werden das letzte Element der Aufgabenliste & das letzte Element der Nutzerlösungsliste
        #mit drei Tabulatoren Abstand und einer neuen Zeile am Ende an den Anfang der Übersicht gesetzt
        uebersicht = str(aufgabenliste.pop()) + "\t\t\t" + str(nutzer_loesungsliste.pop()) + "\n" + uebersicht

    #Überschriften für die beiden Spalten an den Anfang der Übersicht ergänzen
    uebersicht = "Aufgabe\t\t\tdeine Loesung\n" + uebersicht
    #Übersicht in einer Scrollbaren Codebox ausgeben
    ausgabe = codebox("Hier ist die Uebersicht deiner Aufgaben\n\nDu hast "+str(richtige_loesungen)+" von "+
        str(anzahl_aufgaben)+" Punkten erreicht","Uebersicht",uebersicht)

    #Übersicht als Datei speichern
    #Speicherort und Dateiname festlegen
    Dateiname = filesavebox(filetypes=[".txt"])
    if Dateiname == None: #Cancel-Button gedrückt
        pass #nichts tun & nicht speichern

    else:
        #Datei mit obigem Namen erstellen/öffnen und leeren
        datei = open(Dateiname,"w")
        datei.write(uebersicht) #"uebersicht" in Datei einfügen
        datei.close() #Datei schließen

    return ausgabe #"ausgabe" als Funktionswert ausgegeben
#-----
```



uebersicht()-Funktion

```
#-----Erstellen der main-Funktion-----
#-----Gesamtablauf und Zusammenfuehren vorheriger Funktionen-----
def main():
    while True: #Endlosschleife
        start = startseite() #Startseite aufrufen

        if start == "Los gehts": #"Los gehts"-Button gedrückt

            rechenart_testen = rechenart_eingeben()
            if rechenart_testen == None: #Cancel-Button wurde in dieser Funktion gedrückt
                break #Endlosschleife und damit weiteren Programmablauf abbrechen
            else:
                pass #nichts tun

            zahlenraum_testen = zahlenraum()
            if zahlenraum_testen == None: #Cancel-Button wurde in dieser Funktion gedrückt
                break #Endlosschleife und damit weiteren Programmablauf abbrechen
            else:
                pass #nichts tun

            anzahl_aufgaben_testen = anzahl_aufgaben_eingeben()
            if anzahl_aufgaben_testen == None: #Cancel-Button wurde in dieser Funktion gedrückt
                break #Endlosschleife und damit weiteren Programmablauf abbrechen
            else:
                pass #nichts tun

            for i in range(0,anzahl_aufgaben): #untere Funktion so oft aufrufen, wie Aufgaben gerechnet werden sollen
                rechenart_abrufen_testen = rechenart_abrufen()
                if rechenart_abrufen_testen == None: #Cancel-Button wurde in dieser Funktion gedrückt
                    break #Endlosschleife und damit weiteren Programmablauf abbrechen
                else:
                    pass #nichts tun

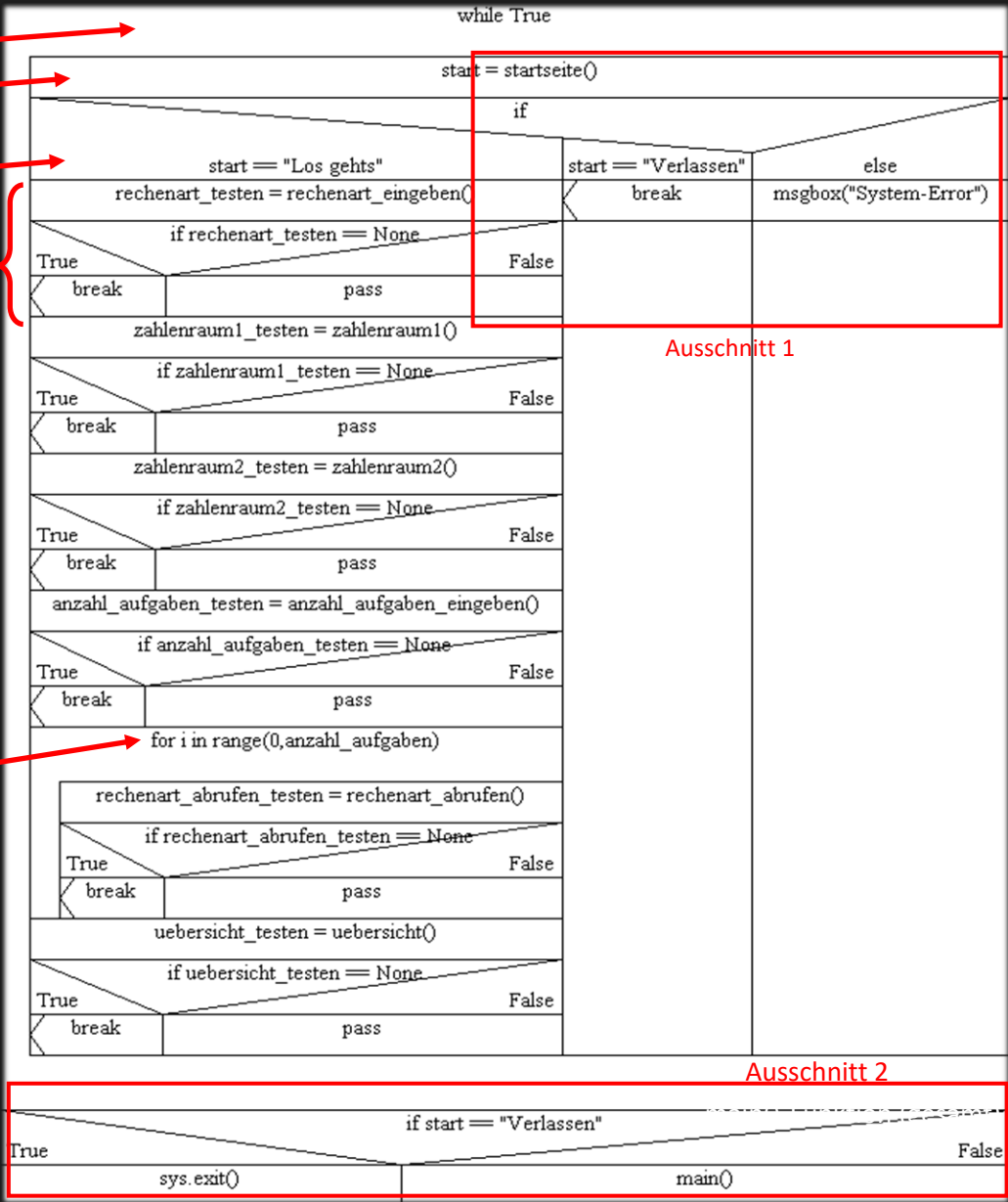
            try: #try-Teil wird solange ausgeführt bis ein Fehler auftritt
                uebersicht_testen = uebersicht()
                if uebersicht_testen == None: #Cancel-Button wurde in dieser Funktion gedrückt
                    break #Endlosschleife und damit weiteren Programmablauf abbrechen
                else:
                    pass #nichts tun
            except: #tritt ein Fehler auf, wird der except-Teil ausgeführt
                break #Endlosschleife und damit weiteren Programmablauf abbrechen

        elif start == "Verlassen": #"Verlassen"-Button wurde gedrückt
            break #Endlosschleife und damit weiteren Programmablauf abbrechen

        else: #es wurde keiner der beiden Buttons gedrückt aber es entsteht trotzdem eine Eingabe
            msgbox("System-Error")

    if start == "Verlassen": #"Verlassen"-Button wurde gedrückt
        sys.exit() #Programm schließen
    else:
        main() #Programm von Anfang an starten

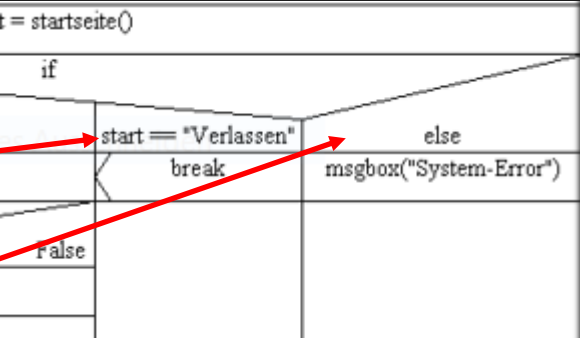
#-----
main() #Aufrufen der main()-Funktion → Starten des Programms
```



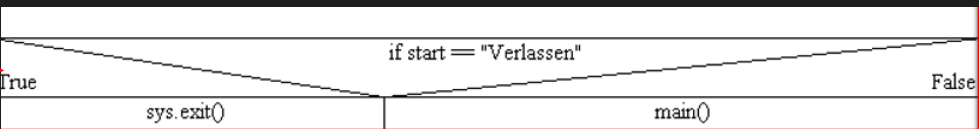
Ausschnitt 1

Ausschnitt 2

main()-Funktion



main()-Funktion (Ausschnitt 1)



main()-Funktion (Ausschnitt 2)