Serie 9B - Eigene Funktionen programmieren

> **Programmieren für Naturwissenschaften FS 2023**
> Gruppe: Sofia Kessler, Florian Mohaupt, Lukas Batschelet

Aufgabe 1: Parameter

Aufgabe

Was passiert im folgenden Code, wenn die funktion `greet("Hans", temp=10)` aufgerufen wird?

```
def greet(name, str1="Guten Morgen!", temp=20):
    print("Hallo {}!\n{}\nEs ist {} Grad draussen.\n".format(name, str1, temp))
```

Beschreiben Sie kurz was passiert und welche Ausgabe dieses Codefragment liefert!

Mögliche Lösung

Wenn die Funktion `greet("Hans", temp=10)` aufgerufen wird, passiert Folgendes:

1. Die Funktion `greet` wird mit dem Argument `name="Hans"` und dem Keyword-Argument `temp=10` aufgerufen.
2. Das Argument `str1` wird nicht explizit übergeben, also wird der Standardwert `"Guten Morgen!"` verwendet.
3. Der `print`-Befehl in der Funktion formatiert den String mit den übergebenen Argumenten und druckt ihn aus.

Die Ausgabe des Codefragments wäre:

```
Hallo Hans!
Guten Morgen!
Es ist 10 Grad draussen.
```

Die Platzhalter `{}` im String werden durch die übergebenen Argumente ersetzt: `name` wird durch "Hans" ersetzt, `str1` durch "Guten Morgen!" und `temp` durch 10.

Aufgabe 2: Higher-Lower-Game

Aufgabe

Im Code auf auf ILIAS finden Sie eine noch nicht fertiggestellte Version des Higher-Lower-Spieles.
Ihre Aufgabe ist es, das Spiel zu vervollständigen.

```
import random as rnd

# TODO: Aufgabe 2: Ihre Implementation der Funktion askguess:
def


# TODO: Aufgabe 3: Ihre Implementation für die Funktion validateguess:
def


# TODO: Aufgabe 1: Zufallszahl für die gesuchte Zahl
n =

## Start des Spielcodes:
over = False
while not over:
    guess = askguess() #Siehe Aufgabe 2
    h, l, c = validateguess(guess, n) # Siehe Aufgabe 3

    ## Folgender Code überprüft ob Sie die Werte h, l, c richtig gesetzt haben. Sollte ein Fehler auftreten so
können
    # Sie die Fehlermeldung nutzen um mögliche logische Fehler bei Ihrer Implementation aus Aufgabe 3 zu
überprüfen.
    # Hinweis: Ein Ausbleiben von Fehlermeldungen heisst nicht, dass der Code wie gewünscht funktioniert.
    assert h + l + c == 1
    assert h >= 0
    assert l >= 0
    assert c >= 0
    ##

    # TODO: Aufgabe 4: Geben Sie userfeedback basierend auf den Werten h, l, c:


    # Ende des Spielcodes
```

Aufgabe 1
Instanzieren Sie die zufindende Nummer `n`, indem Sie eine Zufallszahl in einem geeigneten Intervall zuweisen.

Aufgabe 2
Im Spiel wird eine Funktion `askguess` aufgerufen, welche den Spieler nach einer Eingabe fragen und überprüfen soll, ob die übergebene Zeichenkette eine Zahl ist. Falls nicht, soll der Spieler informiert und zu einer erneuten Eingabe aufgefordert werden! Schreiben Sie eine Funktion, welche diese Aufgabe löst.

Aufgabe 3
Anschließend wird die Funktion `validateguess` aufgerufen. Ergänzen Sie auch hier den fehlenden Code. Die Funktion soll überprüfen, ob die geratene Zahl größer, kleiner oder gleich der gesuchten Lösung ist. Anschließend soll die korrekte Variable `h` (higher), `l` (lower) oder `c` (correct) auf 1 gesetzt werden und alle drei Variablen sollen zurückgegeben werden.
Beispiel: Ist die geratene Zahl größer als die Gesuchte, so sollen die Variablen `h = 1` und `l` sowie `c` gleich 0 sein.
Tipp: Wenn Sie nicht wissen, wie Sie in einer Funktion mehrere Werte zurückgeben können, betrachten Sie Beispiel 76 im Skript auf Seite 102.

Aufgabe 4
Als letzter Schritt sollen Sie dem Spieler Rückmeldung geben, ob die geratene Zahl zu hoch, zu tief oder richtig war. Falls die

Zahl richtig war, soll das Spiel beendet werden! Beispiel: Falls die Variable `l = 1` ist, dann sollte ein Text wie folgendes Beispiel ausgegeben werden:

```
your guess was too low! guess again!
```

Aufgabe 5: Zusatz
Diese Aufgabe ist freiwillig. Implementieren Sie einige der folgenden Ideen:

- Fügen Sie dem Spiel eine maximale Anzahl an Versuchen hinzu. Sollte der/die Spieler/-Spielerin nach x Versuchen die Zahl immer noch nicht erraten haben, ist das Spiel verloren.
- Implementieren Sie ein Hinweissystem, welches den Spielern einen besseren Hinweis geben kann.
- Fügen Sie dem Spiel ein Scoreboard hinzu, welches die Anzahl Versuche und die durchschnittlichen Versuche, sowie einen Highscore festhält.
- Weitere Additionen nach Wahl.

Mögliche Lösung

```python
import random as rnd

# Aufgabe 5: Funktion fragt nach dem Bereich für die Zufallszahl
def ask_range():
    start = input("Bitte gib den Anfang des Bereichs ein (Standard ist 1): ")
    end = input("Bitte gib das Ende des Bereichs ein (Standard ist 100): ")
    return int(start) if start.isnumeric() else 1, int(end) if end.isnumeric() else 100

# Aufgabe 2: Die Funktion fragt den Spieler nach einer Eingabe und überprüft, ob es eine Zahl ist.
def askguess():
    while True:
        guess = input("Bitte gib eine Zahl ein: ")
        if guess.isnumeric():
            return int(guess)
        else:
            print("Das war keine Zahl. Bitte versuche es erneut.")

# Aufgabe 3: Die Funktion überprüft, ob die geratene Zahl größer, kleiner oder gleich der gesuchten Zahl ist.
def validateguess(guess, n):
    h, l, c = 0, 0, 0
    if guess > n:
        h = 1
    elif guess < n:
        l = 1
    else:
        c = 1
    return h, l, c

print("Willkommen zum Zahlenraten!")
print("Du hast 10 Versuche, um die gesuchte Zahl zu erraten.")

# Aufgabe 5: Fragen nach dem Bereich für die Zufallszahl
start, end = ask_range()

# Aufgabe 1: Eine Zufallszahl zwischen start und end wird generiert.
n = rnd.randint(start, end)

# Zusatz Aufgabe: Maximale Anzahl an Versuchen
max_attempts = 10
attempts = 0

# Start des Spielcodes
over = False
while not over and attempts < max_attempts:
    # Aufgabe 2
    guess = askguess()
    attempts += 1
    # Aufgabe 3
    h, l, c = validateguess(guess, n)

    # Überprüfung der Werte h, l, c
    assert h + l + c == 1
    assert h >= 0
    assert l >= 0
    assert c >= 0

    # Aufgabe 4: User-Feedback basierend auf den Werten h, l, c
    if h:
        print("Deine Schätzung war zu hoch! Versuch es erneut.")
    elif l:
        print("Deine Schätzung war zu niedrig! Versuch es erneut.")
    else:
        print("Richtig geraten! Das Spiel ist beendet.")
        over = True
```

```python
# Zusatz Aufgabe: Spiel verloren, falls maximale Anzahl an Versuchen erreicht ist
if not over:
    print(f"Das Spiel ist verloren. Die gesuchte Zahl war {n}.")
```