7.2

Programmierkonventionen

- Die Zeilenbegrenzung auf 99 Zeichen wurde sehr genau eingehalten.
- Bei mehrzeilige Docstrings steht zum Teil vor und nach dem `"""` keine Leerzeile.
- Nach einem Komma steht kein Leerzeichen (dies ist zwar nicht explizit im Styleguide aber in PEP8 vorgegeben).
- Es existiert keine `main()` Funktion und auch keine, wie im Header vorgegeben, Abfrage auf `__name__ == '__main__'`. Die `sixteen_is_dead` Funktion wird stattdessen direkt am Ende der Datei aufgerufen.

Verbesserungsvorschläge

Die Aufteilung in mehrere Funktionen ist gut gelungen, den Namen `user_interface_2` könnte man aber z.B. durch `ask_user_which_dice` ersetzen und dadurch die Aufgabe der Funktion besser wiedergeben.

Bei Nutzereingaben wurde `int(input(...))` verwendet, wenn der Nutzer keine Zahl eingibt, wird dies einen Fehler werfen, der nirgends im Programm aufgefangen wird, dies kann z.B. durch Nutzung von `.isdigit()` umgangen werden.

Die `sixteen_is_dead` Funktion ist relativ komplex, hier wäre eine weitere Unterteilung in Funktionen sinnvoll. Insbesondere wird das Code-Stück (Zeile 87 und 98)

```
1 total_points += dice_number
2 print()
3 print("Deine aktuelle Punktzahl betraegt:",total_points)
4 print()
5 if total_points == 10:
6     time.sleep(3)
7     continue
8 else:
9     break
```

zweimal verwendet.

Die Funktionen `set_players`, `setup_number_of_dices` und `setup_number_of_faces` sind sich sehr ähnlich, hier könnte man eine allgemeine `setup_numeric_value` Funktion schreiben, um den dreifachen Code zu reduzieren.

Nutzereingaben werden zum Teil in den beiden `user_input` Funktion geparsed aber auch in der `sixteen_is_dead` Funktion. Hier würde es sich anbieten, Nutzereingaben nur in den `user_input` Funktionen zu parsen und z.B. `True` oder `False` zurückzugeben, je nach dem, ob der Nutzer noch einmal werfen möchte oder nicht.

Der Docstring der Funktion `restart` ist inkorrekt, die Funktion fragt den Nutzer, ob er das Spiel neustarten will.

Es wäre schön, die gewürfelten Zahlen zu sehen und nicht nur die Summe aller Zahlen.

Die Dopplung des Nutzerinterfaces ist nicht notwendig, der Nutzer könnte auch bei der ersten Frage `"c"` eingeben, um den "Cheating dice" zu verwenden.

Funktionsumfang

Geforderte Funktionen

1. Die Anzahl der Spieler, die Anzahl der Würfel und die Anzahl der Würfelseiten kann eingestellt werden.
2. Das Spiel lässt jeden Spieler so oft würfeln, wie der Spieler möchte, und geht dann zum nächsten Spieler über.
3. Bei einem Spielstand von 16 oder mehr endet das Spiel.
4. Bei einem Spielstand von 9 muss der Spieler aufhören zu würfeln.
5. Bei einem Spielstand von 10 muss der Spieler noch einmal würfeln.
6. Hat am Ende kein Spieler 16 oder mehr Punkte, zeigt das Programm die Spieler mit den wenigsten Punkten an.
7. Das Programm zeigt dem Spieler seine möglichen Eingaben an.
8. Das Programm gibt dem Spieler die Möglichkeit, das Spiel jederzeit zu beenden oder neuzustarten.

Test des Programms auf die geforderten Funktionen

1. Die Anzahl der Spieler, die Anzahl der Würfel und die Anzahl der Würfelseiten kann zu Beginn eines Spiels eingestellt werden.
2. Das Spiel lässt jeden Spieler so oft würfeln, wie der Spieler möchte, und geht dann zum nächsten Spieler über.
3. Bei einem Spielstand von 16 oder mehr beendet des Programm alle `while` Schleifen in `sixteen_is_dead` und beendet das Spiel.
4. Ebenso muss der Spieler bei einem Spielstand von 9 aufhören zu würfeln.
5. Bei einem Spielstand von 10 muss der Spieler nach 3 Sekunden noch einmal würfeln.
6. Hat am Ende kein Spieler 16 oder mehr Punkte, listet das Programm alle Spieler auf, die die wenigsten Punkten haben.
7. Das Programm zeigt dem Spieler seine möglichen Eingaben in jeder Runde an.
8. Das Programm gibt dem Spieler die Möglichkeit durch die Eingaben "q" bzw. "r", das Spiel jederzeit zu beenden oder neuzustarten.

Weitere Funktionen

Das Programm ermöglicht zusätzlich den Einsatz des "Cheating Dice", welcher zwar an dem Würfelergebnis nicht in spielentscheidenden Umfang etwas ändert, die Option ihn zu benutzen ist jedoch praktisch.

Fazit

Das Programm umfasst alle geforderten Funktionen und ist für Fremde relativ gut zu verstehen, auch wenn die `sixteen_is_dead` Funktion besser dokumentiert sein könnte. Das Userinterface ist zwar etwas umständlich aber trotzdem gut gelungen. Leider endet das Programm bei der initialen Abfragen mit einem Fehler, wenn man nichts oder einen String eingibt.

7.3

Veränderungen

- Die Funktionen `setup_number_of_dices`, `setup_number_of_faces` und `roll_cheating_dice` wurden entfernt, da diese nicht benötigt werden.
- Die `roll_dice` Funktion wurde so angepasst, dass sie die gewürfelten Zahlen als Liste und nicht als Summe zurückgibt.
- Die Funktion `user_interface_2` wurde so angepasst, dass eine Liste an Würfelindizes angegeben werden kann, die liegen bleiben sollen.
- Es wurde eine Funktion `get_points_from_dice` geschrieben, die die gewürfelten Zahlen zu einem Score umrechnet.
- Die Funktion `sixteen_is_dead` wurde zu `student_pasch` umgenannt.
- Die Funktion `student_pasch` wurde in großen Teilen an das neue Spiel angepasst.
- Der Quellcode des Programms wurde an den Header2017 angepasst.

Ich habe über dieses Veränderungen hinaus keine Verbesserungen am Quellcode vorgenommen, so dass es immer noch bei der Eingabe eines Strings zu Beginn des Programms zum Absturz kommen kann.

Beurteilung des Programms

Einige der Funktionen (z.B. die `user_interface` und `restart` Funktion) mussten nicht angepasst werden. Die Aufteilung in zwei `user_input` Funktionen war für die Veränderungen hilfreich, da nur die zweite angepasst werden musste. Wie schon in 7.2 angemerkt, war die `roll_dice` Funktion in ihrer alten Form nicht flexibel genug (die Aufsummierung der Zahlen war für das neue Programm unbrauchbar). Ebenso war die `sixteen_is_dead` zu komplex, um einfach auf das neue Spiel umgeschrieben zu werden und musste daher nahezu komplett neu geschrieben werden. Trotzdem war das Umschreiben relativ einfach und die Struktur des Programms nicht überaus statisch.