Aufgabenblatt 3

Das sind die Aufgaben der Woche 3. Es handelt sich um Verzweigungen in Programmen, also if und else.

# Aufgabe 11 – Eingaben und Konditionale

Wir wollen lernen, wie man im Programm verzweigen kann, also eine sogenannte Selektion durchführen kann. Dies geschieht mit if-else-Ausdrücken.

Schreiben Sie ein Programm, das zwei Fliesskommazahlen einliest, beide miteinander vergleicht, die kleinere in eine Variable **minimum** schreibt und die grössere in eine Variable **maximum**. Finden Sie eine gute Code-Lösung für den Fall, dass die beiden Zahlen gleich gross sind.

Lösung:

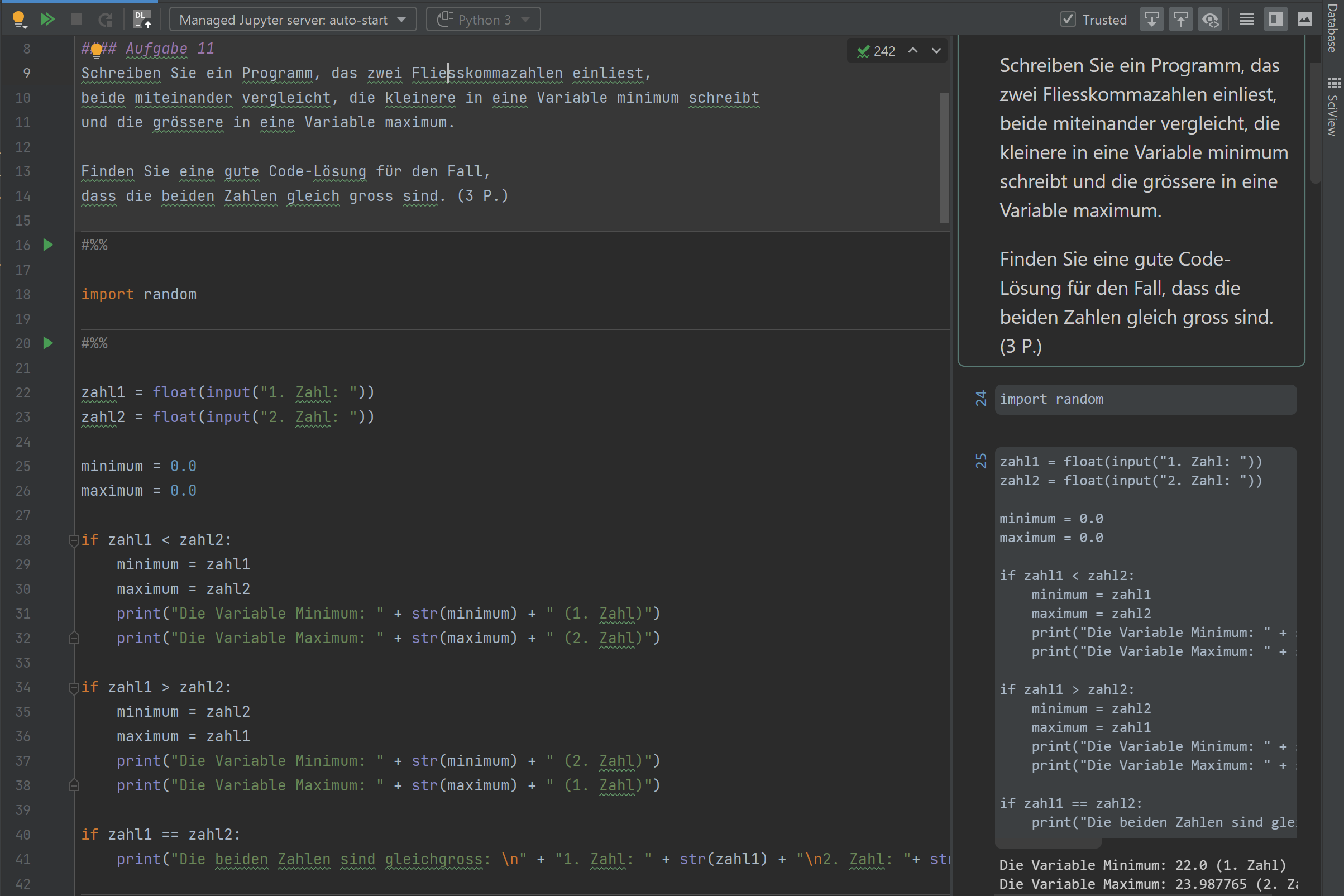


Abbildung - Aufgabe 11

Die Lösung ist auch im python Skript im .zip zu finden.

# Aufgabe 12 – Zufallszahlen und Bibliotheken

Eine ganze Zahl (Integer) soll eingegeben werden. Zusätzlich soll eine weitere ganze Zahl zwischen 0 und 100 zufällig gezogen werden. Vergleichen Sie die Werte der eingegebenen Zahl und der gezogenen Zahl und geben Sie **Gewonnen** aus, wenn die beiden Zahlen übereinstimmen, ansonsten **Verloren**. **(4 P.)**

Lösung:

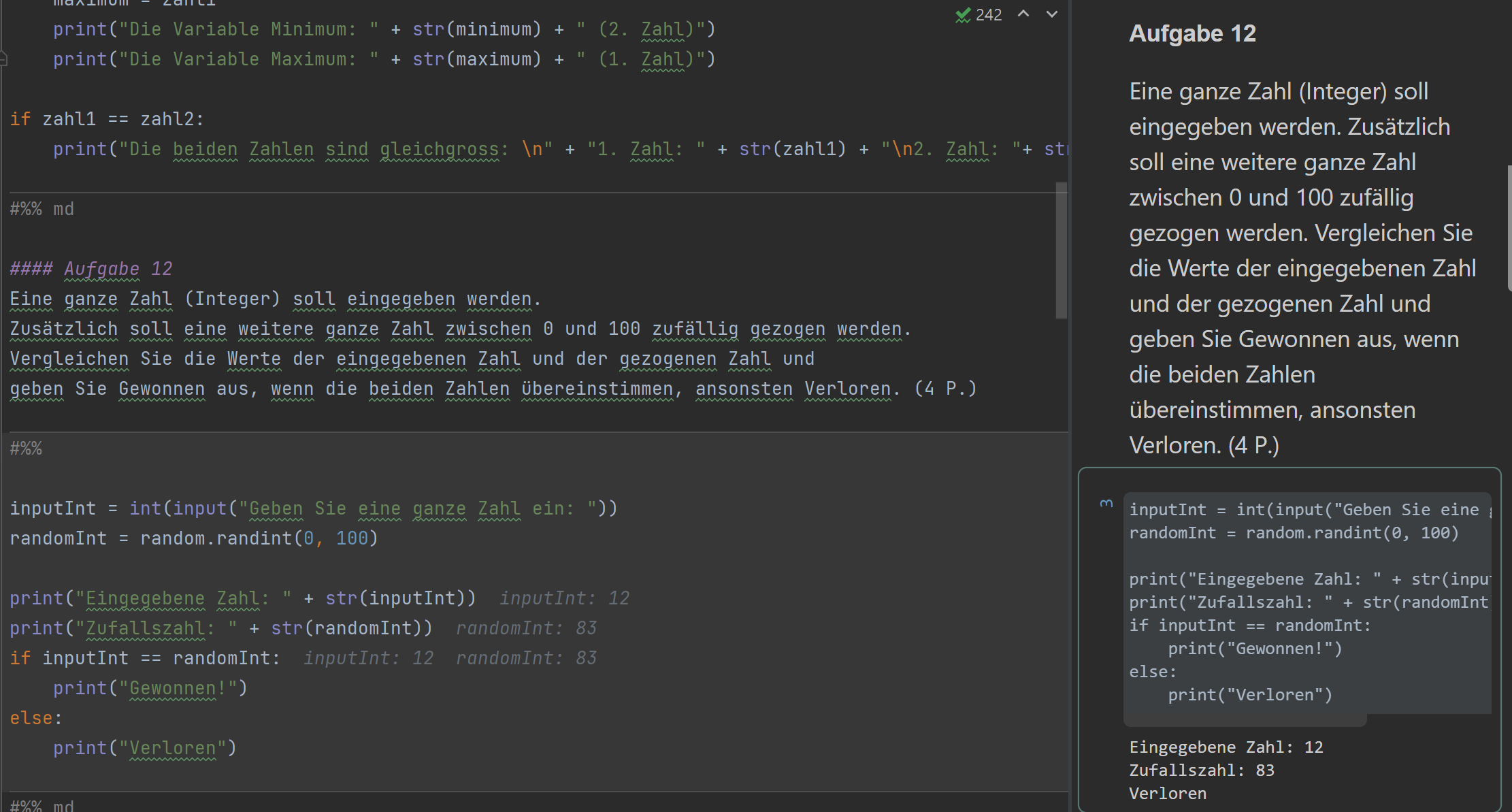


Abbildung - Aufgabe 12

Die Lösung ist auch als Python Datei (mit Kommentaren) im .zip.

# Aufgabe 13 – Kniffel, das Würfelspiel

Ihre folgende Aufgabe soll es ein, 5 Würfelvariablen zufällig Werte zwischen 1 und 6 zuzuweisen und diese Werte alle auszugeben. **(1 P.)** Schreiben Sie ein Programm, das überprüft, ob:

1. Alle 5 Werte gleich sind. **(1 P.)**
2. Alle 5 Werte unterschiedlich sind **(2 P.)**
3. Die Summe der 5 Werte grösser als 20 und kleiner als 25 ist. **(1 P.)**

Lösung:

Damit diese Aufgabe gelöst werden kann, müssen zuerst die Random Bibliothek und die 5 Zufallszahlen zugewiesen werden:

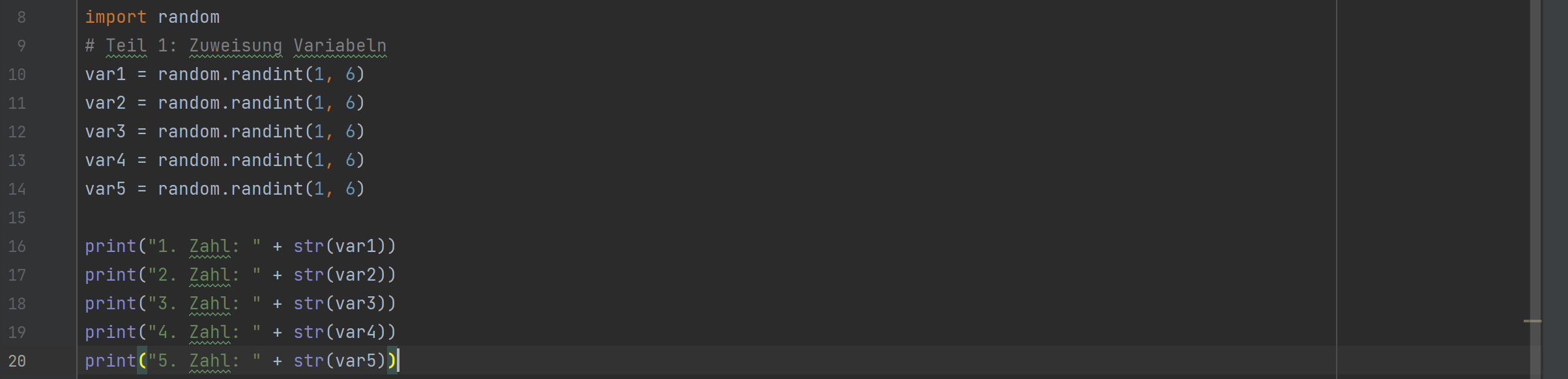


Abbildung - Aufgabe 13 Import&Variabelzusweisung

Die Zahlen werden ausgegeben, damit beim «Würfeln» auch etwas spass dabei ist.

1. Da alle Variabeln gleich sein müssen, können diese mit «=» aneinandergehängt werden:



Abbildung - Aufgabe 13 a)

1. Das Abprüfen der Variabeln auf nichtgleichheit, ist etwas komplizierter, da es 5+4+3+2+1-5 Möglichkeiten gibt, also 10.

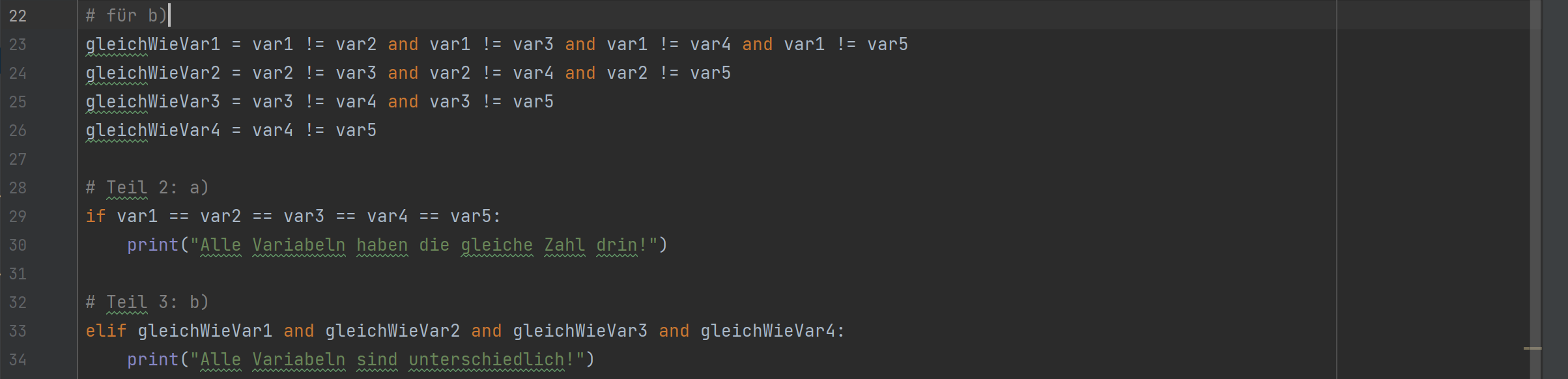
Alternativ zu meiner Lösung könnte auch der boolsche Wert in einer Variabel gespeichert werden, anstatt alles im «if» zu haben:  


Abbildung - Aufgabe 13 b) Alternative

Dieses Beispiel ist auch im Python Skript Aufgabe13\_02 zu finden. Generell wäre es eventuell sinnvoll das mit einer Schleife zu überprüfen, ich habe mich aber dagegen entschieden, aufgrund von Überkomplizierung des Codes.  
Zu beachten ist, dass der Code vor Zeile 29 stehen muss, da es sich um ein «elif» handelt und die Zeilen 31 und 32 nur im «if» stehen dürfen und nicht ausserhalb.

1. Im letzten Schritt wird die Summe der Variabeln in einer neuen Variabel gespeichert und danach im «if» überprüft, ob diese Summe zwischen 20 und 25 liegt:

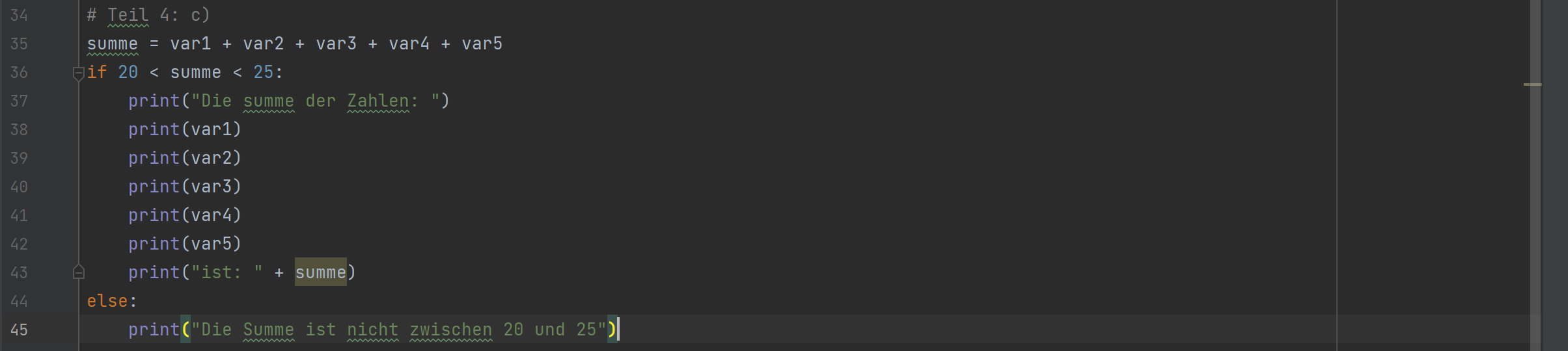


Abbildung - Aufgabe 13 c)

Im «else» wird nur noch ausgegeben, falls die Summe nicht zwischen 20 und 25 liegt.

# Aufgabe 14 – Boolsche Ausdrücke

Zu was werten die folgenden Ausdrücke aus (im Falle, dass Sie syntaktisch korrekt sind)? Begründen Sie Ihre Antwort kurz.

(a) True and False and True or False **(1 P.)**

(b) (3 < 3.0) or (not(not(True or False))) **(1 P.)**

(c) 3 != 4 != 5 != 6 != 3 **(1 P.)**

(d) a = 3

b = 4

c = 5

print(a\*\*2 + b\*\*2 == c\*\*2) **(1 P.)**

Lösung:

1. Das Resultat ist: False

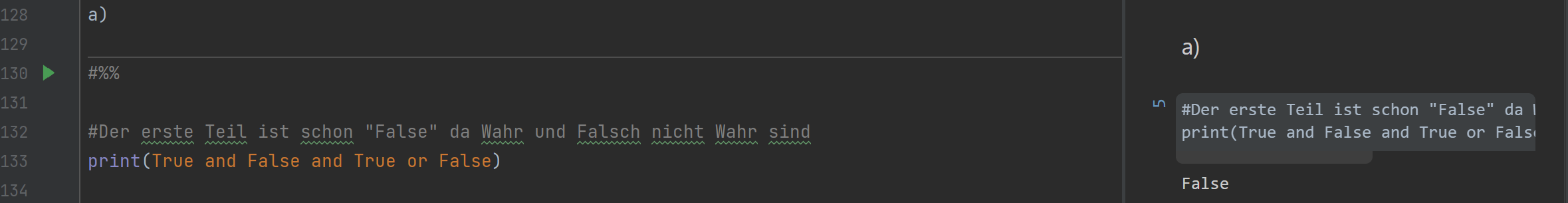
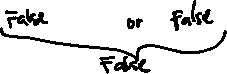
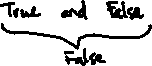


Abbildung - Aufgabe 14 a)

Der Ausdruck wird von links nach rechts ausgewertet.



1. Das Resultat ist: True

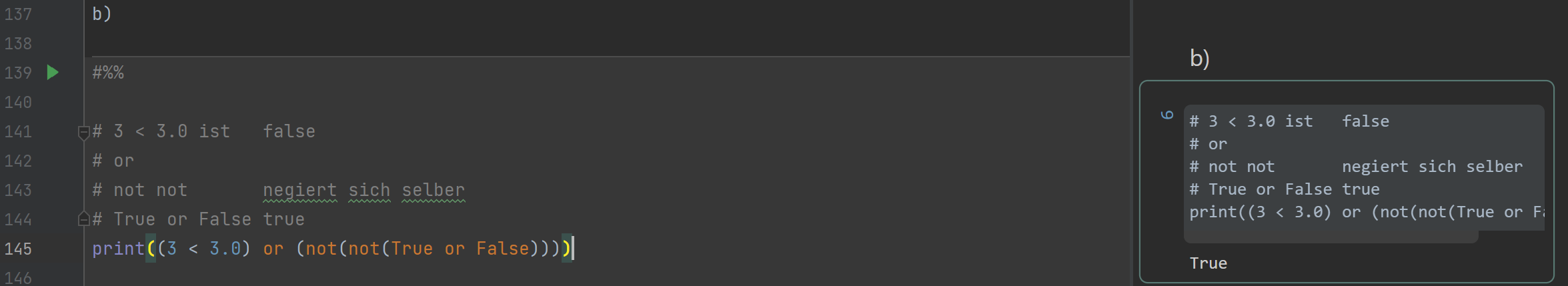


Abbildung - Aufgabe 14 b)

Die Klammern machen, dass zuerst 3 mit 3.0 verglichen wird, was False ist und erst danach die verschachtelten «not».   
Da eine Doppelverneinung vorhanden ist, können diese auch wieder gelöscht werden bzw. sind diese unnötig/überflüssig.  
True or False gibt True zurück  
Also schlussendlich heisst der Ausdruck: False or True 🡪 Resultat: True

1. Das Resultat ist: True

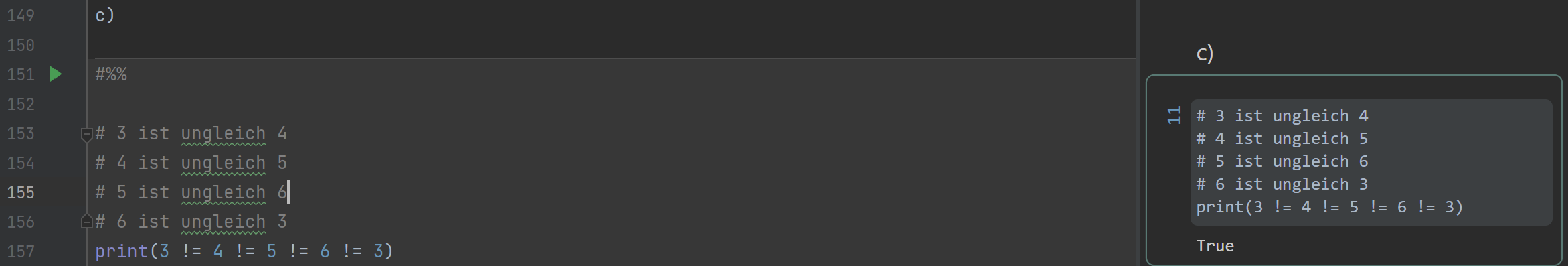


Abbildung - Aufgabe 14 c)

Da von links nach rechts überprüft wird, wird das erste «3» nicht mit dem letzten «3» verglichen.

1. Die Lösung ist: True

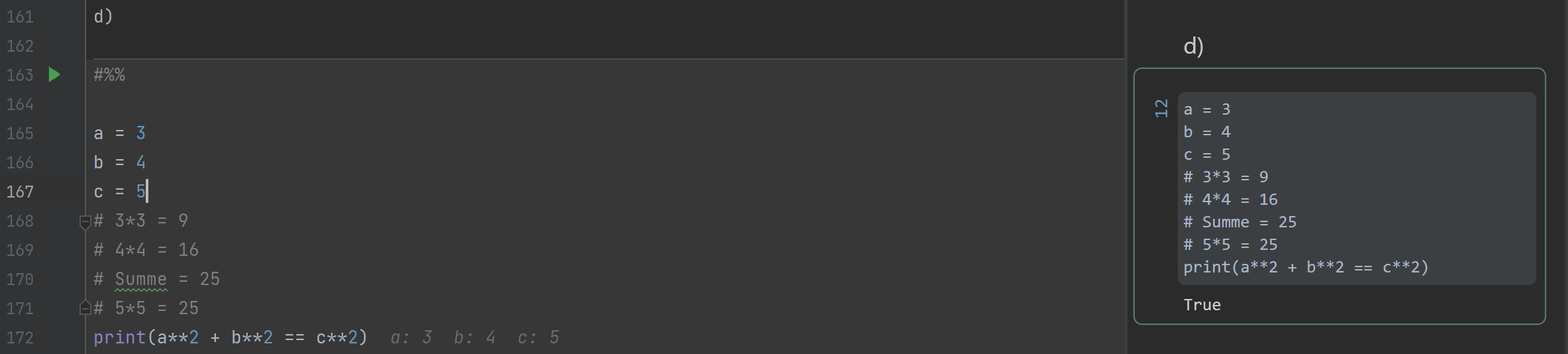


Abbildung - Aufgabe 14 d)

Die Antwort ist True, weil uns das Pythagoras bewiesen hat. (In einem Dreieck)  
 a2 + b2 = c2  
 32 + 42 = 52  
 9 + 16 = 25 🡪 Korrekt!

Die Lösung ist auskommentiert im Python-Skript Aufgabe14.py.

# Aufgabe 15 – Primzahlentest > 7

Schreiben Sie ein Programm, das von einer eingegebenen Integerzahl grösser 7 und kleiner gleich 100 testet, ob diese eine Primzahl ist oder nicht. Eine Zahl zwischen 8 und 100 ist eine Primzahl genau dann, wenn sie nicht durch 2, 3, 5 und 7 teilbar ist. **(4 P.)**

Lösung:

Ich habe 2 Lösungen gefunden.

1. Testet in einzelnen «if» ob die eingegebene Zahl durch 2, 3, 5, 7 teilbar ist.

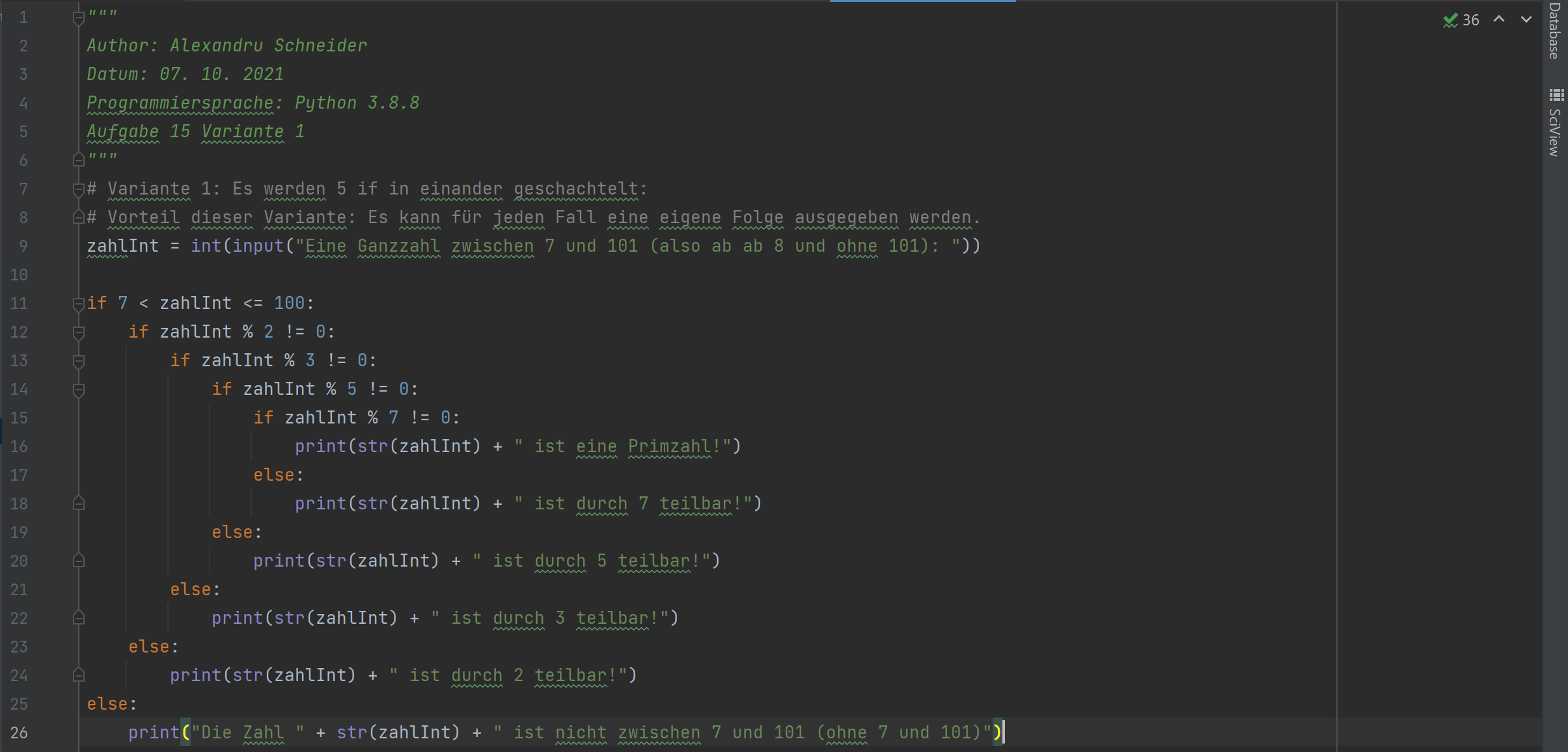


Abbildung - Aufgabe 15 Variante 1

Zuerst wird überprüft, ob der eingegebene Interger zwischen 7 und 100 ist. Dabei zu beachten ist, dass der eingegebene Integer auch 100 sein darf. (aber erst bei 8 beginnen)  
Die 4 verschachtelten «if» werden dann auf die oben angegebenen Zahlen überprüft. Dabei wird mit dem Modulo ermittelt, ob ein Rest vorhanden ist. Wenn der Rest nicht «0» ist, dann ist dieser Integer keine Primzahl.

Der Vorteil dieser Variante ist, dass zu jeder Zahl ein eigener «else»-Fall erstellt werden kann. In diesem Fall ist das nicht nötig, deshalb habe ich auch die Variante 2 erstellt:

1. Testet in einem «if» ob die Zahl zwischen 7 und 100 ist, sowohl auch ob die Zahl eine Primzahl ist. (Zur Hilfe der Lesbarkeit wurde das in Klammern gesetzt.

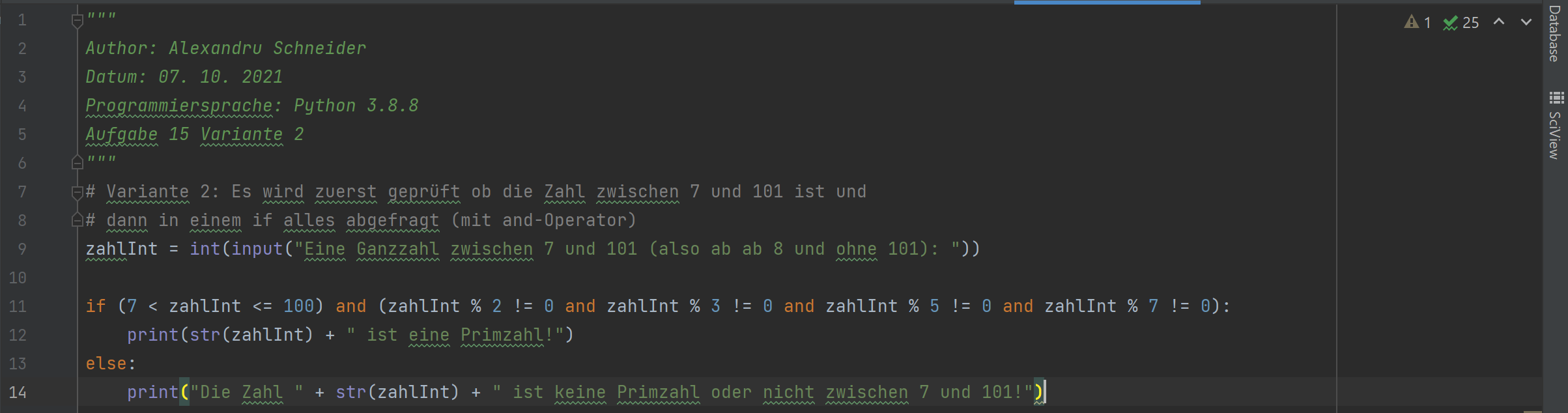


Abbildung - Aufgabe 15 Variante 2

Nun kann beim «else» auch nicht ermittelt werden, warum bzw. wann false (oder in anderen Fällen der Fehler) eingetroffen ist.