

Cyber Security

Grundlagen der Programmierung

Grundlagen der Program- mierung



Einführung

- Es gibt eine Vielzahl an Programmiersprachen auf dem Markt
- Wichtig ist die Sprache zu wählen, welche zum Einsatzgebiet passt

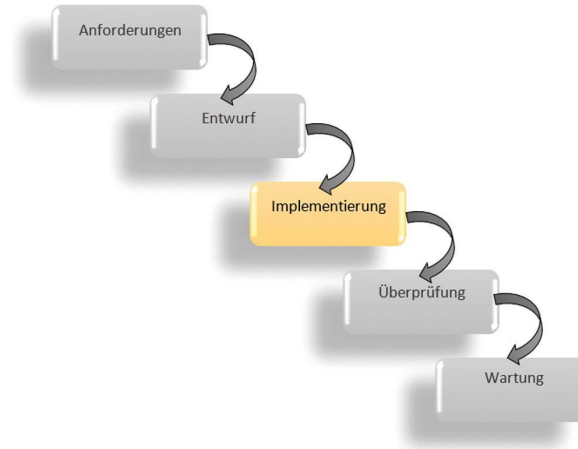
Beispiele:

- C# für Spieleentwicklung mit Unity
- Javascript für WebAnwendungen
- Kotlin für Anwendungsentwicklung auf Android Geräten
- etc. ...



Einführung

- Die Softwareentwicklung gliedert sich in mehrere Phasen.
- Die Programmierung im Step Implementierung steht dabei im Fokus.



Einführung

- Ziel der Softwareentwicklung ist das Lösen von Problemen mit Hilfe von Programmen
- Programme können auf unterschiedlichsten Geräten mit unterschiedlichsten Betriebssystemen laufen:
 - z.B. Smartphone, Computer, Fernseher, etc.
- Programme bestehen aus Algorithmen



Einführung

Erklärung Algorithmus:

- Ein Algorithmus kann als To Do Liste zur Erledigung einer Aufgabe verstanden werden. Die Ausführung des Algorithmus führt dabei immer wieder zum gleichen Ergebnis (siehe Turing Maschine).
- Algorithmen finden sich nicht nur in der Mathematik oder der IT-Welt, jeder Prozess und jede Handlung kann mit Hilfe von Algorithmen beschrieben werden.

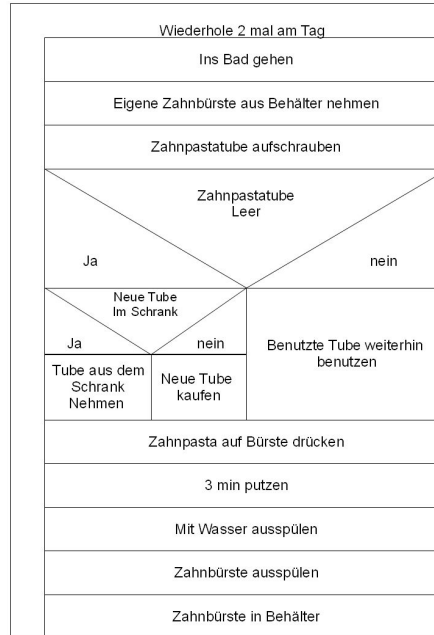


Einführung

- Es folgt eine beispielhafte Grafik für einen Algorithmus zum Zähne putzen.
- Eine solche Grafik nennt man „Struktogramm“.
- Ein Struktogramm hilft dabei den Algorithmus zu formulieren und zu visualisieren.



Einführung



Einführung

Das EVA Prinzip:

- Jedes (Computer-) Programm funktioniert nach diesem Prinzip, ansonsten könnten Mensch und Computer nicht miteinander interagieren.
- EVA ist eine Abkürzung und bedeutet:
 - **E**ingabe
 - **V**erarbeitung
 - **A**usgabe



Einführung

Das EVA Prinzip - die Eingabe:

- Damit ein Programm etwas berechnen bzw. ausführen kann, muss nicht nur ein Algorithmus vorhanden sein, sondern auch eine Eingabe erfolgen. Ohne Eingabe weiss der verarbeitende Teil, das „V“, sonst nicht was er zu erledigen hat.
- Eine Eingabe kann durch den Benutzer über die Tastatur oder die Maus erfolgen oder das Programm holt sich die Eingabe selbst, indem es z.B. Sensorwerte ausliest.



Einführung

Das EVA Prinzip - die Verarbeitung:

- Der verarbeitende Teil ist das eigentliche Programm.
- Daten werden in den Algorithmus hineingegeben und damit Berechnungen durchgeführt.
- Nach der Ausführung dieses Steps, wird das Resultat an den Ausgabeteil weiter gegeben.



Einführung

Das EVA Prinzip - die Ausgabe:

- Das Resultat der Verarbeitung wird in diesem Step dem Benutzer präsentiert.
- Die Art, wie dem Benutzer das Ergebnis präsentiert wird, kann sehr vielseitig sein:
 - Dies kann auf einem Monitor, auf einem Drucker, als Statusanzeige über eine optische Anzeige, etc. ... erfolgen.



Das Python Einmaleins

- In diesem Kurs gehen wir detaillierter auf die Programmiersprache Python ein
- Als Entwicklungsumgebung (IDE: Integrated Development Environment) wird PyCharm (Community Edition) von JetBrains verwendet
- Die aktuelle Version [hier herunterladen](#).



Das Python Einmaleins

Wofür wird Python eingesetzt?

- Web Entwicklung
- Software Entwicklung (Apps & Tools)
- Mathematische Problemstellungen (z.B. für Simulationen)
- Systemseitiges Scripting



Das Python Einmaleins

Vorteile von Python:

- Lauffähig auf vielen Plattformen (Windows, Mac, Linux, Raspberry Pi)
- Einfache Syntax
- Wenig Boilerplate Code (notweniger Code zum Definieren von Funktionalitäten)



Das Python Einmaleins

Vorteile von Python:

- Python kann prozedural, objektorientiert oder funktional eingesetzt werden
- Python verwendet einen Interpreter, der die Ausführung des Programms direkt nach dem Schreiben ermöglicht
 - Das Übersetzen des Codes erfolgt während der Ausführung



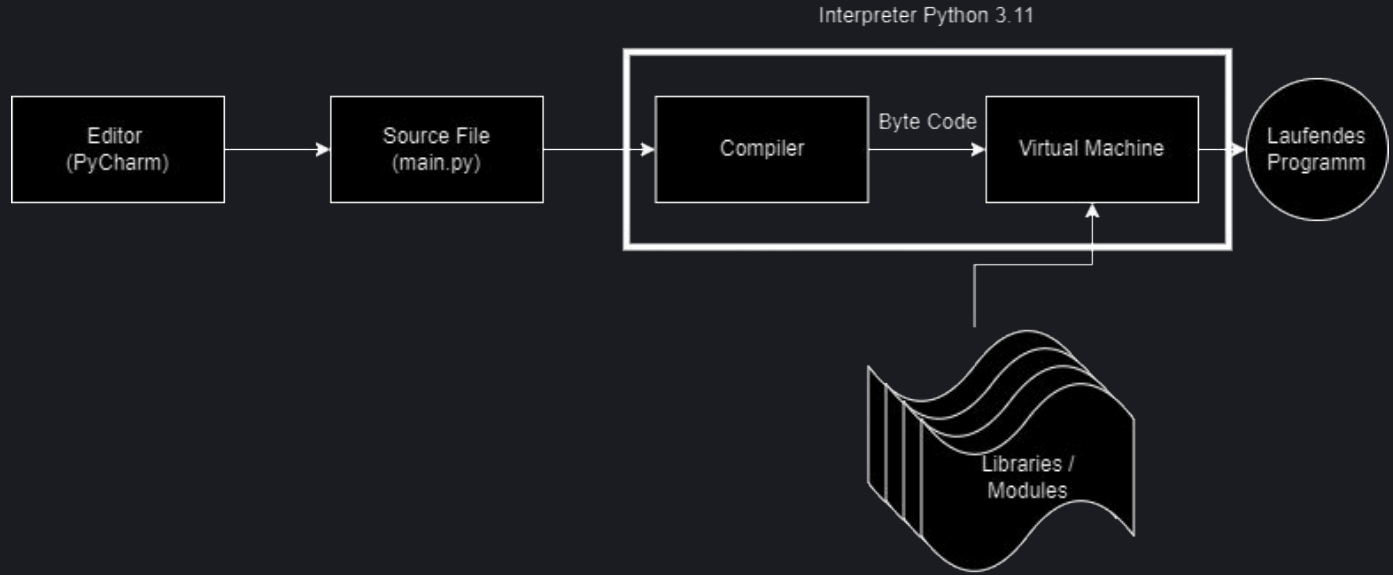
Das Python Einmaleins

Installation von Python:

- [Aktuelle Version herunterladen](#)
- Ein Programm in einem Editor der Wahl verfassen (z.B. 'main.py')
- Die Endung .py signalisiert, dass es sich um ein Python Programm handelt
- Das Programm kann über die Kommandozeile direkt ausgeführt werden
 - z.B.: C:\Users\User1>python main.py



Das Python Einmaleins



Datentypen, Variablen und mehr

Variablen:

- Der Name darf theoretisch unendlich lang sein
- Zusammensetzung aus Groß- und Kleinbuchstaben A-Z, a-z
- Ziffern von 0-9 und dem Unterstrich
- Es dürfen Unterstriche „_“ im Bezeichner enthalten sein
- Das erste Zeichen im Variablennamen darf keine Zahl sein



Datentypen, Variablen und mehr

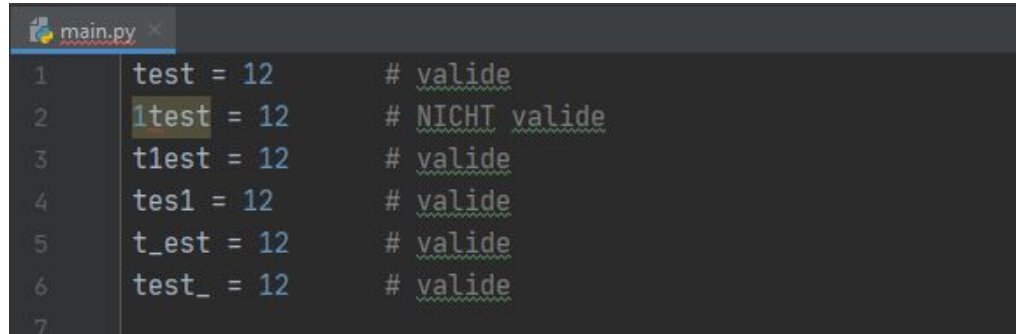
Variablen:

- Eine Zuweisung erfolgt über einen Variablennamen und ein anschließendes Gleichheitszeichen
- Hinter dem Gleichheitszeichen folgt der Wert der Variable
- Jede Variable in Python ist ein Objekt und kann **ohne Deklaration oder Definition**, sondern durch einfache Zuweisung benutzt werden
 - Unterschied zu vielen anderen Programmiersprachen



Datentypen, Variablen und mehr

Variablen:



```
main.py x
1 test = 12      # valide
2 1test = 12     # NICHT valide
3 t1test = 12    # valide
4 tes1 = 12      # valide
5 t_est = 12     # valide
6 test_ = 12     # valide
7
```



Datentypen, Variablen und mehr

Bestimmte Schlüsselwörter sind als Bezeichner für Variablen verboten:

False	def	if	raise
None	del	import	return
True	elif	in	try
and	else	is	while
as	except	lambda	with
assert	finally	nonlocal	yield
break	for	not	
class	from	or	
continue	global	pass	



Datentypen, Variablen und mehr

Kommentare:

- Beliebig viele Zeilen aus dem Quellcode können dem Compiler vorenthalten werden
- Kommentare dienen dazu den Code besser zu verstehen. (bspw. Für Leute, die ihn nicht selbst geschrieben haben.)
- Indem das Zeichen # verwendet wird, wird alles, was sich in derselben Zeile hinter dem #-Zeichen befindet, nicht als auszuführender Code interpretiert



Datentypen, Variablen und mehr

Kommentare:

- Entwickler können so Kommentare in ihrem Quellcode hinterlassen, um die Funktionalitäten zu erklären
- Es können temporär Stellen aus dem Code entfernt werden, ohne sie löschen zu müssen. (z.B. zum Debugging)



Datentypen, Variablen und mehr

Kommentare – schlechter Stil:

- Kommentare sollten am Anfang der Zeile starten
- Funktionalität sollte nur zu Testzwecken auskommentiert werden, nie beim fertigen Produkt



Datentypen, Variablen und mehr

Kommentare:

```
1  # Cloud Command - Grundlagen der Programmierung in Python
2
3  x = 3
4
5  usage
6
7  def myFunc():
8      y = 7+4
9      # h = a + c / d ** f
10     global x
11     z = y + x
12     h = z - y, #+ 3 * 4
13
14 myFunc()
```



Datentypen, Variablen und mehr

Variablen:

- Eine Ausgabe kann mit Hilfe der `print()` Funktion erfolgen
- Alle Werte innerhalb der Klammern werden in der Konsole ausgegeben
- Ein Templatestring mit einem `f` kann Variablen in den Output einbinden
 - Beispiel: `print(f" Meine Variable a ist {a}")`



Datentypen, Variablen und mehr

Variablen:

- Mehrere String Werte können mit Hilfe des + Operators verbunden werden
- Bei Zahlen wird der + Operator mathematisch angesehen
- Strings und Zahlen können nicht ohne Weiteres kombiniert werden



Datentypen, Variablen und mehr

Variablen:



```
main.py x
1 # Cloud Command - Grundlagen der Programmierung in Python
2
3 a = "Cloud Command "
4 b = "ist "
5 c = "toll!"
6 print(a+b+c)
7 #Output: Cloud Command ist toll!
8
9 d = 5
10 e = 6
11 print(d+e)
12 #Output 11
13
14 print(a+d)
15 #wird nicht ausgeführt, da ein Fehler entsteht
```



Datentypen, Variablen und mehr

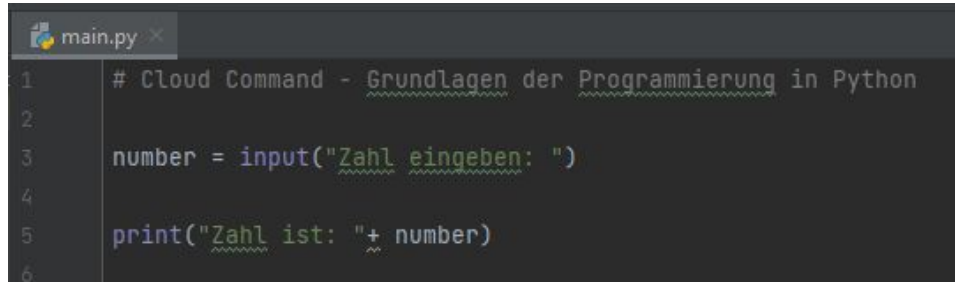
Variablen:

- Im Gegensatz zu `print()` kann mit `input()` der Eingabewert eines Benutzers eingelesen und in einer Variable gespeichert werden
- Beim Input wird die Eingabe immer als String interpretiert!
- Bei einer Weiterverarbeitung von anderen Datentypen muss ein sogenanntes ‚Type Casting‘ erfolgen



Datentypen, Variablen und mehr

Variablen:

A screenshot of a code editor window titled 'main.py'. The code is written in Python and demonstrates variable assignment and input/output. It includes a comment, an input statement, and a print statement. The code is as follows:

```
1 # Cloud Command - Grundlagen der Programmierung in Python
2
3 number = input("Zahl eingeben: ")
4
5 print("Zahl ist: " + number)
6
```



Datentypen, Variablen und mehr

Variablen:

- Bei der Zuweisung eines Werts wird dem Objekt auch gleichzeitig der entsprechende Typ zugewiesen
- Bei einer erneuten Zuweisung mit einem Wert eines anderen Typen wird der Typ des Objekts angepasst



Datentypen, Variablen und mehr

Häufig verwendete Typen sind:

- Für Zahlen die Typen int (Integer) und float (Float)
- Für Text der Typ str (String)
- Der Wahrheitswert bool (Boolean)
- Der Listentyp list



Datentypen, Variablen und mehr

Häufig verwendete Typen sind:

```
main.py x
1 test0 = 12                # class 'int'
2 test1 = 12.2              # class 'float'
3 test2 = "12"              # class 'str'
4 test3 = '12'              # class 'str'
5 test4 = True              # class 'bool'
6 test5 = [1, "on", False]  # class 'list'
7
```



Datentypen, Variablen und mehr

Type Casting:

- Beim Type Casting kann eine Variable mit einem bestimmten Datentyp als ein anderer Datentyp interpretiert werden

Beispiel: `x = float("2")`

- über die spezielle Angabe des Typen in der Klammer, wird der Stringwert "2" als float Wert 2.0 interpretiert



Datentypen, Variablen und mehr

Datentypen - Zahlen:

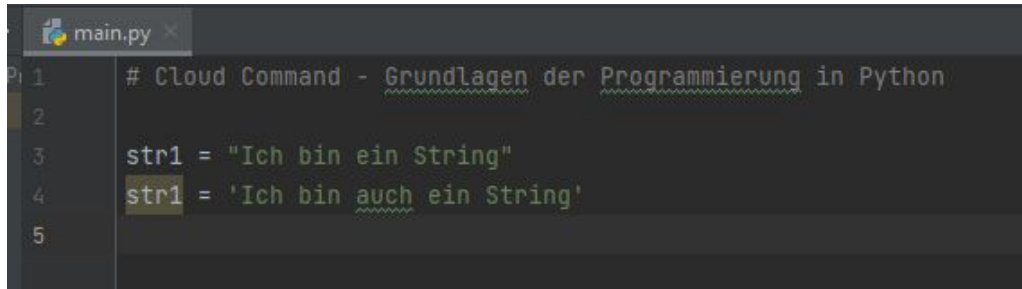
- Es gibt zwei Zahlentypen
- Integer beschreibt alle ganzen Zahlen (0,1,2,etc)
- Float beschreibt die Fließkommazahlen (1.234; 7.26; 3.1415926)



Datentypen, Variablen und mehr

Datentypen - Zeichenketten:

- Zeichenketten (Texte) werden im Englischen als Strings bezeichnet (str)
- Strings können sowohl mit einfachen als auch doppelten Anführungszeichen zugewiesen werden



```
main.py x
1 # Cloud Command - Grundlagen der Programmierung in Python
2
3 str1 = "Ich bin ein String"
4 str1 = 'Ich bin auch ein String'
5
```



Datentypen, Variablen und mehr

Datentypen - Zeichenketten:

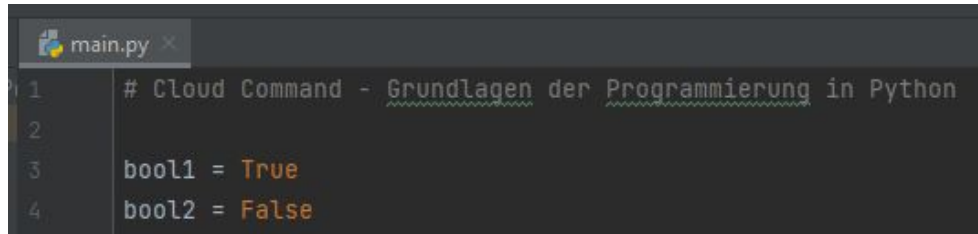
- Strings können auf verschiedene Arten und Weisen verändert werden
- Strings können verbunden und aufgeteilt werden
- Strings können automatisch GROß und klein geschrieben werden
- Python hat viele Methoden, die Strings modifizieren oder verändern können
- Strings können formatiert werden
 - z.B: strip(), replace, find(), join(), ...



Datentypen, Variablen und mehr

Datentypen - Wahrheitswerte:

- Wahrheitswerte werden im Englischen als Booleans bezeichnet (bool)
- Booleans bilden die beiden Werte „Wahr“ und „Falsch“ (True , False) ab
- Sie werden bei vielen bedingten Anweisungen benötigt



```
1 # Cloud Command - Grundlagen der Programmierung in Python
2
3 bool1 = True
4 bool2 = False
```



Datentypen, Variablen und mehr

Datentypen - Listentyp:

- Listentypen werden im Englischen als list bezeichnet
- Der list Typ in Python wird in vielen anderen Programmiersprachen Array genannt
- Eine Liste kann leer sein []
- Eine Liste ist ein Art Gruppe



Datentypen, Variablen und mehr

Datentypen - Listentyp:

- Die Elemente können beliebig viel sein, dabei können sie sich gleichen
- Listen sind von links nach recht geordnet und über ihre Stelle in der Liste zugreifbar
- Listen können mit Listen verschachtelt werden
- Listen können während der Ausführung des Programms größer und kleiner werden



Datentypen, Variablen und mehr

Datentypen - Listentyp:

```
liste1 = ['Cloud', 'Command', 'Python', 'Programmierung']  
liste2 = [1, 2, 3, 4, 5]  
liste3 = [1, 2, 'Cloud', 'Command']  
liste4 = []  
liste5 = [liste1, liste2]
```



Datentypen, Variablen und mehr

Datentypen - Listentyp:

- Mit dem Index Operator [] kann auf ein Element der Liste zugegriffen werden
- Der Index einer Liste beginnt immer bei 0!



Datentypen, Variablen und mehr

Datentypen - Listentyp:

```
main.py x
1  # Cloud Command - Grundlagen der Programmierung in Python
2
3  leereListe = []
4
5  liste1 = [7, "Hund", 7, "Hase", 4, 89, ["Hahn", 5, 1, "Pferd"]]
6
7  print(liste1[1])
8  #Output ist Hund
9
10 # Wir ändern das Element an Stelle 2
11 liste1[1] = "Hündin"
12 print(liste1[1])
13 #Output ist Hündin
```



Datentypen, Variablen und mehr

Datentypen - Listentyp:

- Neben list gibt es drei weitere Typen, die eine Sammlung in Python darstellen: **Tuple, Set und Dictionary**
- **Tuple:** mytuple = ("Hund", "Katze", "Maus") //unveränderbar, Duplikate erlaubt, Index Zugriff
- **Set:** myset = {"Hund", "Katze", "Maus"} //unveränderbar, keine Duplikate, kein Index Zugriff
- **Dictionary:** mydictionary = {"Rasse": "Husky", "Geschlecht": "männlich"} // veränderbar, keine Duplikate, Zugriff über Schlüssel



Datentypen, Variablen und mehr

Operatoren:

- **Zuweisungsoperator:** $a = 4$, kann auch mit arithmetischen Operatoren kombiniert werden ($a += 3$ entspricht $a = a + 3$).
- **Vergleichsoperator:** $3 == 4$ ergibt den Wahrheitswert False
- **weitere Vergleichsoperatoren:** $<$, $>$, $<=$, $>=$, $!=$ (kleiner, größer, kleiner gleich, größer gleich, ungleich)



Datentypen, Variablen und mehr

Operatoren:

- **Arithmetische Operatoren:** + , - , * , / , % , ** , // (Addition, Subtraktion, Multiplikation, Division, Modulo, Potenz, ganze Division)
- **Logische Operatoren:** and, or, not ($1 < 3$ and $4 < 7$) Wahr, wenn Ausdrücke auf beiden Seiten wahr sind.



Datentypen, Variablen und mehr

Globale Variablen und Gültigkeitsbereiche:

- Variablen können global genutzt werden, wenn sie außerhalb von Funktionen erstellt werden. Sie sind von überall aus im Code nutzbar.
- Globale Variablen können auch innerhalb von Funktionen benutzt werden
- Variablen mit demselben Namen, die innerhalb von Funktionen erstellt werden sind lokal
 - Der Wert der globalen Variable wird dadurch nicht verändert
- Um eine globale Variable auch innerhalb einer Funktion verändern zu können wird das Schlüsselwort global verwendet



Datentypen, Variablen und mehr

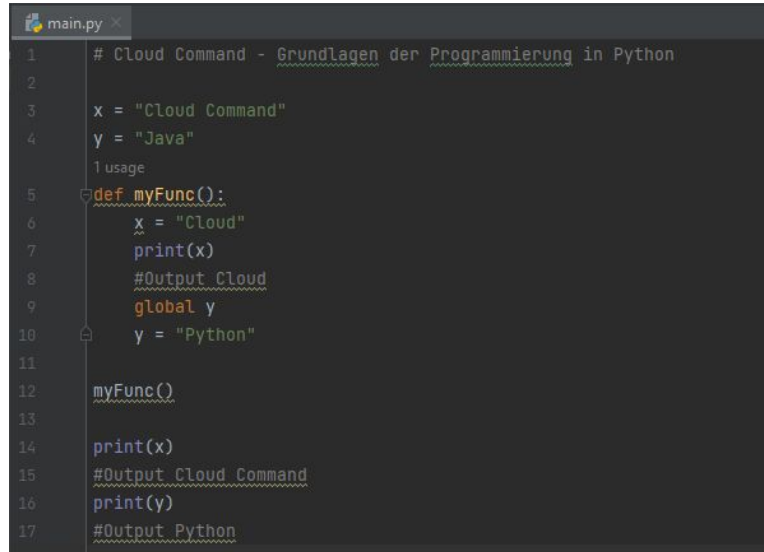
Globale Variablen und Gültigkeitsbereiche:

- Die Variable y konnte innerhalb der Funktion verändert werden, da sie als global deklariert wurde
- x kann nicht innerhalb der Funktion verändert werden, da x nur lokal ist



Datentypen, Variablen und mehr

Globale Variablen und Gültigkeitsbereiche:



```
1 # Cloud Command - Grundlagen der Programmierung in Python
2
3 x = "Cloud Command"
4 y = "Java"
5
6 usage
7
8 def myFunc():
9     x = "Cloud"
10    print(x)
11    #Output Cloud
12    global y
13    y = "Python"
14
15 myFunc()
16
17 print(x)
18 #Output Cloud Command
19 print(y)
20 #Output Python
```





CloudCommand