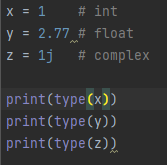
**Nummern in Python:**

In Python gibt es Drei verschiedene Numerische typen (int, float, complex).



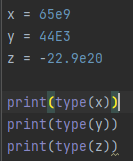
Mit Type() kannst du dir ausgegeben lassen welche um welchen datentype es sich handelt.

Ein Bild, das Text enthält.

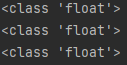
Automatisch generierte Beschreibung

Int/Integer sind Ganze zahlen, die positiv oder negativ sein können, aber ohne kommerziellen einer unlimitierten Länge.

Float/floting point number ist eine Nummer, die positiv oder negativ sein kann und mehrere nachkommerstellen haben kann.

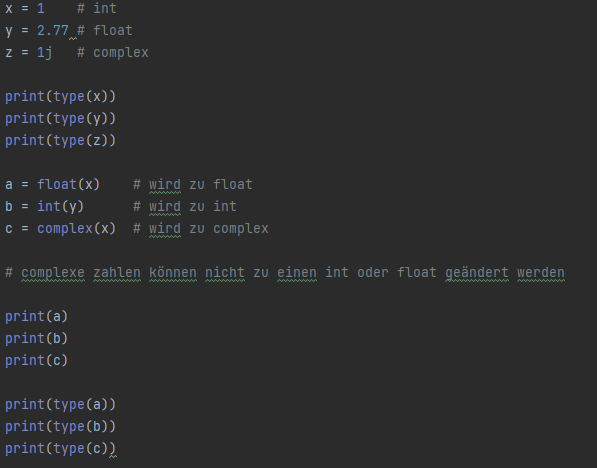


Float kann auch mit ‚e‘ ein wissenschaftliches Nummern zu bilden die einen Indikator auf die kraft von 10 weisen.



Komplexe Nummern werden mit einen ‚j‘ für den imaginären teil geschrieben.

Man kann auch die verschiedenen Typen mit den Methoden int(), float(), complex() konvertieren.

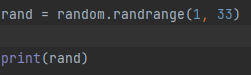


Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Python hat keine Random() Funktion um zufällige nummern erstellen zu können aber es hat ein eingebautes Modul random mit dem man zufällige zahlen erstellen kann.







**Operatoren in Python:**

Die Operatoren werden dazu genutzt, um mit variablen und werten Operationen durchführen zu können z.B.: (10 + 5) hier wird der + Operator genutzt um die zwei werte 10 und 5 zu adieren.

Python Operatoren werden in folgende Gruppen unterteilt:

Arithmetische Operatoren werden mit numerischen werten verwendet, um Mathematische Operatoren auszuführen.

* Arithmetische Operatoren:
  + + addieren
  + – subtrahieren
  + \* multiplizieren
  + / dividieren
  + % Modulo / Rest einer Division
  + \*\* Potenzierung
  + // floor division/ Grund Division

Zuweisungsoperatoren werden verwendet, um Variablen werte zuzuweisen.

* Zuweisungsoperatoren:
  + = x = 5 🡺 x = 5
  + += x += 5 🡺 x = x + 5
  + -= x -= 2 🡺 x = x - 2
  + \*= x \*= 2 🡺 x = y \* 2
  + /= x /= 2 🡺 x = y / 2
  + %= x %= 2 🡺 x = y % 2
  + //= x //= 2 🡺 x = y // 2
  + \*\*= x \*\*= 2 🡺 x = y \*\* 2
  + &= x &= 2 🡺 x = y & 2
  + |= x |= 2 🡺 x = y | 2
  + ^= x ^= 2 🡺 x = y ^ 2
  + >>= x >>= 2 🡺 x = y >> 2
  + <<= x <<= 2 🡺 x = y << 2

Vergleichsoperatoren werden verwendet, um zwei Werte zu vergleichen.

* Vergleichs Operatoren:
  + == ist gleich
  + != ist nicht gleich
  + > grösser als
  + < kleiner als
  + >= grösser als oder gleich
  + <= kleiner als oder gleich

Logische Operatoren werden verwendet, um Anweisungen zu kombinieren.

* Logik Operatoren:
  + and gibt (true) zurück, wenn alle der Aussagen richtig sind. X < 5 and x < 10
  + or gibt (true) zurück, wenn eine der Aussagen richtig ist. X < 5 or x< 4
  + not gibt (true) zurück, wenn das Original (false) ausgibt not(x < 5 and x < 10)

Identitätsoperatoren werden verwendet, um die Objekte zu vergleichen. (nicht wenn sie gleich sind, sondern wenn sie tatsächlich dasselbe Objekt sind.)

* Identitätsoperatoren:
  + is gibt (true) zurück, wenn beide variablen gleiche Objekte sind
  + is not gibt (true) zurück, wenn beide variablen nicht die gleichen Objekte sind

Zugehörigkeitsoperatoren werden verwendet, um zu testen, ob die Sequenz in dem Objekt dargestellt wird.

* Zugehörigkeitsoperatoren:
  + in gibt (true) zurück, wenn eine Sequenz mit dem spezifischen wert im Objekt gegeben ist
  + not in gibt (true) zurück, wenn der spezifische wert nicht im Objekt gegeben ist

Bitweise Operatoren werden verwendet, um (binäre) Zahlen zu vergleichen.

* Bitweise Operatoren
  + & setzt jedes Bit auf 1, wenn beide Bits sind, 1
  + | setzt jedes Bit auf 1, wenn ein Bit der beiden auf 1 ist
  + ^ setzt jedes Bit auf 1, wenn nur eines der beiden Bits auf 1 ist
  + ~ tauscht die Bits aus (1 = 0, 0 = 1)
  + << verschiebt nach links, indem 0 rechts hineinschiebt und die Bits ganz links fallen lassen
  + >> verschiebt nach rechts, indem 0 links hineingeschoben werden und rechts werden die Bits fallengelassen