

# Tutorium SS17

---



R  SALIND  
Learn(Science) with Programming

# Syntax

---

- Keine geschweiften Klammern {}
- Programmcode wird durch Einrücken getrennt – Ziel: gute Lesbarkeit
- Schleifen:
  - for
  - while
- Bedingungen:
  - If ... elif ... else
- Kommentare:
  - # Das ist ein Kommentar
  - `““““` Ein Block Kommentar `““““`
- Skript Sprache

# Bedingungen

---

```
if <condition>:  
    #do something  
elif <condition>:  
    #do something  
else:  
    #do something else
```

Code	Output
<pre>a = 6 b = 9 if a &lt; b:     print ("1") elif a == b:     print ("0") else:     print ("-1")</pre>	1

# Schleifen I

Code	Output
<pre>for x in range(1,10):     print(x)  languages = ["Java", "Python", "R", "Bash"] # a list for l in languages:     print ("Ich kann bald",l,"anwenden")</pre>	<pre>1 2 3 4 5 6 7 8 9 Ich kann bald Java anwenden Ich kann bald Python anwenden Ich kann bald R anwenden Ich kann bald Bash anwenden</pre>

# Schleifen II

Code	Output
<pre>x = 0 while x &lt; 10:     print (x)     x +=1  # kein x++ verfügbar else:     print ("Die Schleife wurde beendet")</pre>	<pre>0 1 2 3 4 5 6 7 8 9 Die Schleife wurde beendet</pre>

- `else` wird ausgeführt wenn:
  - `<Bedingung>` nicht mehr erfüllt ist
  - `break` aufgerufen wurde

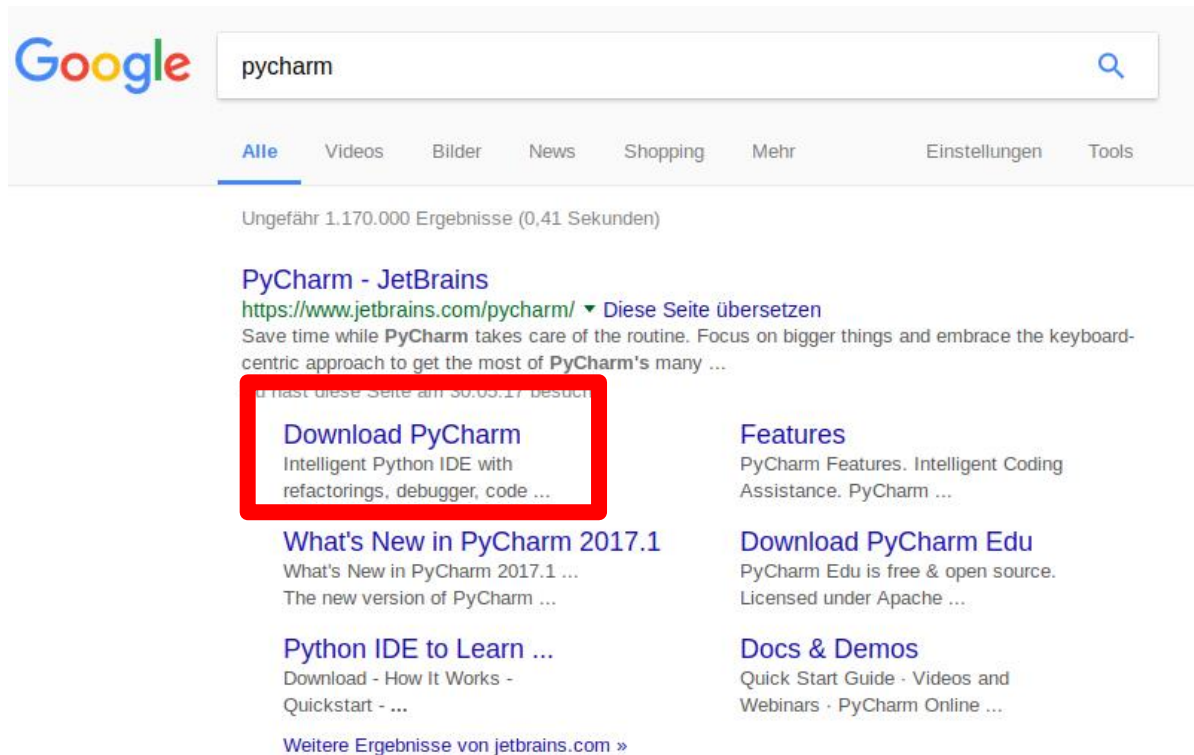
# Wo schreibt man den Code?

---

- Dem Interpreter (python3.3) ein Skript übergeben mit evtl. Argumenten
  - `python meinSkript.py <arg1>`
  - Skript schreiben mit einem Editor (Gedit, Kate, Notepad++, ...)
    - Oder einer IDE (Jupyter, Pycharm ...)
- Interaktive Modus (python in Konsole eintippen)
  - Schließen mit Strg + D oder dem Befehl `quit()`
  - Primäre Eingabeaufforderung: `>>>`
  - Sekundäre Eingabeaufforderung: `...`
    - Für mehrzeilige Eingaben
    - Beenden der mehrzeiligen Eingabe: Leerzeichens + Enter

# PyCharm

---



A screenshot of a Google search for 'pycharm'. The search bar contains 'pycharm' and the search button is visible. Below the search bar, there are tabs for 'Alle', 'Videos', 'Bilder', 'News', 'Shopping', 'Mehr', 'Einstellungen', and 'Tools'. The search results show approximately 1,170,000 results in 0.41 seconds. The first result is 'PyCharm - JetBrains' with a URL 'https://www.jetbrains.com/pycharm/'. Below the URL, there is a description: 'Save time while PyCharm takes care of the routine. Focus on bigger things and embrace the keyboard-centric approach to get the most of PyCharm's many ...'. A red box highlights the link 'Download PyCharm' with the subtext 'Intelligent Python IDE with refactorings, debugger, code ...'. Other search results include 'What's New in PyCharm 2017.1', 'Python IDE to Learn ...', 'Features', 'Download PyCharm Edu', and 'Docs & Demos'. A link 'Weitere Ergebnisse von jetbrains.com »' is at the bottom.



## Community

Lightweight IDE  
for Python & Scientific  
development

**DOWNLOAD**

Free, open-source

# PyCharm

---

1. `cd ~/Downloads`
2. `mv pycharm-community-2017.1.3.tar.gz ~/<ordner_eurer_wahl>`
3. `cd ~/<ordner_eurer_wahl>`
4. `tar -xvf pycharm-community-2017.1.3.tar.gz`
5. `cd pycharm-community-2017.1.3/`
6. `./bin/pycharm.sh &`
  - & Öffnet Prozess in der Shell
  - Zur Shell \$PATH Variablen hinzufügen oder als Alias ausführen (beides optional)



# Grundoperationen und Variablen

---

- + , - , \* , /
  - Klammern zum Gruppieren
- Weitere Operatoren
  - // : Abgerundete Division
  - = : Wertzuweisung
- Variablen müssen definiert sein
  - $y = 0$
- Systemvariablen im interaktiven Modus:
  - \_ : Letzter ausgegebener Wert

# Datentypen

---

- Man muss den Datentyp **nicht** explizit den Variablen zuordnen
- Integer
- Fließkommazahlen
- Zeichenketten (Strings)

Code	Output
<pre>print("'Hallo'") print('"Hallo"') print("Hallo \nWie geht es dir?\n")  s="Ich bin ein"+' String' print(s)  print("a"*6) s = "123456" i = s[0] # i is not a char</pre>	<pre>'Hallo' "Hallo" Hallo Wie geht es dir?  Ich bin ein String aaaaaa</pre>

# Datenstrukturen I

---

- Tupel
  - Eine Variable die mehrere Werte, unterschiedlicher Datentypen, hat, die durch ein Komma getrennt sind
  - Unveränderlich
  - „Packing“ und „Unpacking“

Code	Output
<pre>t = 123,456,"Hallo" # Packing x,y,z = t          # Unpacking print(x,y,z)</pre>	<pre>123 456 Hallo</pre>

# Datenstruktur II

---

- Sequenzdatentypen:
  - list, string
  - Unterstützten „slicing“:
  - Erster Index: 0

Code	Output
<pre>languages = ["Java", "Python", "R", "Bash"] # a list s = "Ich bin ein String" # a string  print(languages[1:2]) print(languages[2:]) print(languages[:3]) print(s[:-2]) print(s[-2:])</pre>	<pre>['Python'] ['R', 'Bash'] ['Java', 'Python', 'R'] Ich bin ein Stri ng</pre>

# Datenstrukturen III

---

- Mengen
  - Ungeordnet, keine doppelten Elemente
  - Operatoren: in, -, |, &, ^

Code	Output
<pre>obst = {"Apfel", "Orange", "Birne"} print("Apfel" in obst) gemuese = {"Tomate", "Gurke"} korb = obst   gemuese # union print(korb) print(korb - obst) # a minus b print(obst &amp; korb) # intersection print(obst ^ korb) # in a or b, but not in both print(obst in korb)</pre>	<pre>True {'Birne', 'Orange', 'Apfel', 'Gurke', 'Tomate'} {'Gurke', 'Tomate'} {'Birne', 'Orange', 'Apfel'} {'Gurke', 'Tomate'} False</pre>

# Datenstrukturen IV

---

- Dictionaries (Key:Value)
  - Schlüssel sind eindeutig und „unsortiert“

Code	Output
<pre>aa = {'N': 'Asn', 'R': 'Arg', 'A': 'Ala'} print(list(aa.keys())) print(sorted(aa.keys())) del aa["A"] print ("A" in aa) aa["A"] = "Asn" print("A" in aa)</pre>	<pre>['N', 'R', 'A'] ['A', 'N', 'R'] False True</pre>

# Beispiel

Code	Output
<pre>## a function def multiply(a,b):     result = a*b     return result print(multiply(3,4))  ## reading command line arguments import sys print("Value of sys.argv:\n", " \$ ".join(sys.argv)) print("Number of arguments: ", len(sys.argv)) print("Argument list: ", ", ".join(sys.argv))  ## reading a file file = open("/home/lukas/testfile.txt", "r") for line in file:     # end="" avoids an empty line after each print     print(line,end="") ## or file = open("/home/lukas/testfile.txt", "r") print(file.readlines())  file.close() # always close a file handler</pre>	<pre>12 Value of sys.argv:  /home/lukas/pycharm_workspace/Tutorium/tutorium.py \$ param1 \$ param2 Number of arguments: 3 Argument list: param1,param2 line1 line2 line3 ['line1\n', 'line2\n', 'line3\n']</pre>

# Pyhton Village – Rosalind Challenge

---

- Pyhton Village
  - Übungen für die Pyhton Basics
  
- Rosalind Challenge
  - <http://rosalind.info/classes/enroll/79b51edf5a>
  - 3 Wochen ab heute
  - Bioinformatische Aufgaben
  - Vorteil fürs ProPra



# Achtung!

---

- Befehle im Internet können bei euch ggf. nicht funktionieren da sie zu Python 2.x gehören und nicht zu Python 3.x !

**Python 3.0**

**Python 2.x**

