Python-Grundlagen

1. Wertzuweisung, Typen, Typumwandlung

```
 1   a = 2          #  Die Variable a ist ein Speicherplatz mit dem Namen a (Adresse)
 2                  #  und dem Inhalt 2. Stelle dir eine Schachtel mit der Aufschrift a vor.
 3                  #  a ist hier vom Typ int (integer).
 4   b = 5.21       #  Kommentare beginnen mit #.
 5   c = a + b
 6   e = f = 6
 7   a = a + b      #  abgekürzt:  a += b,   zu a wird b addiert und in a gespeichert, entsprechend  /=, *=
 8   a = 2**3       #  2 hoch 3
 9   d = 8  % 3     #  ergibt 2, Rest bei Division

10   Ergebnis = Zahl1 + Zahl2        #  rechts vom Gleichheitszeichen wird gerechnet, links gespeichert
11          #  auf der linken Seite steht nur eine Variable, Ausnahme: Mehrfachzuweisung
12          #  Variablennamen sinnvoll wählen, keine Umlaute, mit einem Buchstaben beginnen
13          #  oder einem Unterstrich, Groß- und Kleinschreibung ist zu beachten


14  print("Informatik")
15  print (" Ergebnis = ", a)                    #  Ausgabe

16  Eingabe = input( )                           #  Eingabe ist vom Typ string
17  a = input("Bitte eine Zahl eingeben: ")      #  a ist vom Typ string



18   Variablen-Typen      Es gibt große und kleine Schachteln, die Unterschiedliches aufnehmen.
                          In Python gibt es keine explizite Variablendeklaration.


19  int            #   ganze Zahl
20  float          #   Kommazahl
21  string         #   Zeichenkette
22                 #  Typ wird bei der ersten Zuweisung automatisch festgelegt
23  bool           #   True oder False, Verknüpfung mit  and (&)  oder  or (|)



24    Typumwandlung,        häufig bei der Eingabe erforderlich

25  i  = int(b)
26  k = float(a)
27  v = str(e)        #   e  wird in den Typ string umgewandelt.
28  u = eval(v)
```

2. Schleifen

```
1   for x in [4,5,-6]:
2           print(x**2)             # ohne Zeilenumbruch   print(x**2, end=" ")


3   b = 0                           #  for-Schleife
4   for a in range(1, 100):         #  a von 1 (einschließlich)  bis 100 (ausschließlich), : wichtig
5           b = b + a               #  Einrückung erforderlich
6   print (b)


7   b = 0
8   for a in range(1, 4):
9           b = b + a
10          print (b)               #  Alle gleich eingerückten Befehle gehören zu einem Block


11  for a in range(1, 20, 5):       #  Schrittweite hier 5,  auch möglich:  range(10,0,-1)
12          print (a)               #  Schrittweite muss ganzzahlig sein.




13  a = 1                           #  while-Schleife
14  while a < 5:                    #  Block wird solange ausgeführt, wie die Bedingung wahr ist
15          a = a + 1
16  print(a)                        #  5


17  a = 1
18  while a <= 5:
19          a = a + 1
20          print ("erhöhe a um eins")
21  print (a)


22  break       #   verlässt die Schleife sofort und macht  nach der Schleife weiter
23  continue    #   beendet einen Schleifendurchlauf und macht mit dem nächsten Durchlauf weiter


24  import time
25  for a in range(1, 100):
26     print (a)
27     time.sleep(0.1)              #     verzögerte Ausführung,  time.sleep(Anzahl der Sekunden)
```

3. Bedingte Ausführung

```
1   if  a < 5:
2      print ("a ist kleiner als 5")

                                  #   <=  kleiner gleich,  >=  größer gleich,  !=  oder <>  ungleich, == gleich

3   if  a < 5:
4      print ("a ist kleiner als 5")
5   else:
6      print("a ist größergleich 5")


7   if  a < 5:
8      print ("a ist kleiner als 5")
9   elif  a > 5:
10     print( "a ist groesser als 5")
11  else:
12     print("a ist 5")


13  if  a < 5 and b > 5:
14          print ("Das ist selten")
```

4. Funktionen

```
15  def gruss( ):
16          print ("Hallo und Guten Tag")          #  Aufruf mit  gruss( )


17  def grussMitName(name):              #  Funktion mit Parameter, Aufruf  z.B. grussMitName("Martin")
18          print( "Hallo ", name, " und Guten Tag")


19  def fkt(x):
20       return  x**2

21  for i in range(11):
22      print(i," ",fkt(i))


23  def Max(a, b):
24          if a > b:  print(a)
25          else: print(b)
26  Max(3, 2)
```

```
1   def max(a, b):                  #  eine Funktion kann einen Wert zurückliefern
2           if a > b:  return a     #  der Wert wird mit return spezifiziert
3           else:   return b        #  return beendet die Funktion sofort
4   print( max(3, 2))
```

```
5   def ggT(a,b):
6      while b>0:
7        a, b = b, a % b            #  Mehrfachzuweisung
8      return a
9   print(ggT(24,16))               #  8
```

```
10  def Fakultaet (n):              #  Rekursion
11          if n <= 1:  return 1
12          return n * Fakultaet (n-1)
13
14  n = 5
15  print (n, "! =", Fakultaet(n) )
```

```
16  n = 5
17  Fakultaet = 1                   #  iterativ
18  for k in range(1, n+1):
19          Fakultaet = Fakultaet * k
20  print (n, "! =", Fakultaet )
```

```
21  def fib(n):
22     if n == 0:
23       return 0
24     elif n == 1:
25        return 1
26     else:
27        return fib(n-1) + fib(n-2)
```

```
28  def fibonacci(n):
29     a,b = 0,1
30     list=[1,1]                   # siehe Listen
31     for x in range(1,n-1):
32        a,b = b,a+b
33        list.append(a+b)
34     return list

35  print(fibonacci(10))
```

5. Listen

```
liste = [7,12,5,23,1,14,18,33,4]
print (len(liste))                          #  Anzahl der Elemente 9


a = [7, 12, 5, 23, 1]
a.append(6)                         #  [7, 12, 5, 23, 1, 6]
c=a.pop( )                          #  6
c=a.pop(1)                          #  12          pop(index),   index beginnt mit 0


liste = ["Eric", "John", "Michael"]
for i in liste:
        print(i)


s =[ ]
for a in range(10):
        s.append(a**2)
print(s)


liste = [4,1,7,55,10,7,2]
sorted(liste)                       #  [1, 2, 4, 7, 7, 10, 55]   liste wird nicht verändert
del liste [2]                       #  [4, 1, 55, 10, 7, 2]
del liste [1:4]                     #  [4, 10, 7, 2]      Elemente von Position 1 bis 3 werden gelöscht


liste = [7,12,5,23,1,14,18,33,4]            #  größtes Element einer Liste
max = liste[0]                              #  erstes Element:  liste[0]
for i in liste:
        if i > max:
                max = i
print(max)


liste = [7,12,5,23,1,14,18,33,4]            #  Position des größten Elements einer Liste
max = liste[0]
pos = 0
for i in range(1, len(liste)):
        if liste[i] > max:
                pos = i
print(pos)


liste = [x*x  for x in range(-5,13)]
print(liste)
```

Bereiche

```
liste = [4,1,7,55,10,7,2]
print(liste [3:])          #  [55, 10, 7, 2]
print(liste [:3])          #   [4, 1, 7]
print(liste [2:5])         #   [7, 55, 10]
```

Wie bei range() ist nur die Untergrenze enthalten.

Listen kopieren

```
L1 = [1,2,3]
L2 = L1
L1[1] = 7

print(L2)
print(L1)
```

Ausgabe
[1, 7, 3]
[1, 7, 3]

Beachte: Mit L2 = L1 wird L2 die Adresse (der Pointer) von L1 zugewiesen.
Damit beziehen sich die Veränderungen von L1 auch auf L2.

Das Gewünschte wird erreicht mit:

```
from copy import *

L1 = [1,2,3]
L2 = copy(L1)
L1[1] = 7

print(L2)
print(L1)
```

Ausgabe:
[1, 2, 3]
[1, 7, 3]

2dimensionaler Array (Listen in Liste)

```
1   n = 4            # Zeilen
2   m = 5            # Spalten

3   L= [[(i,j)  for j in range(1,n)]  for i in range(1,m)]
4   print(L)


5   for i in range(0, m-1):
6     print(L[i])


7   for i in range(0, m-1):
8     for j in range(0, n-1):
9         print(L[i][j], end =" ")
10    print(" ")
```

```
n=10
cities = [[randint(1,100),randint(1,100)]  for k  in range(n)];

nach einer Koordinate sortieren:
K=sorted(cities, key = lambda  a: a[1])        #    in  a: a[1]     ist der Name a beliebig
print(K)                                       #    absteigend mit    sorted (...  ,reverse=True)
```

Verschiedenes

```
1   from math import *                  #  Import aus der Mathematik-Bibliothek
2   r = 10                              #  alternativ   import math
3   umf  = 2 * pi * r                   #               umf  = 2 *math.pi * r
4   print(umf)

5   print ("%.2f "%(umf))               #  2 Nachkommastellen, gerundet


6   from random import *
7   r = random( )                       #  Zufallszahl aus [0;1]
8                                       #  alternativ   import random
9                                                        r = random.random ( )
10  r =randint(1,6)                     #  randint(a, b): gleichverteilt, ganze Zahlen in [a; b]



11  sqrt(5)                             #  2.23606797749979
12  sum(range(1, 101))                  # 5050

13  a=1                                 #  Eine Variable, die in einer Funktion definiert wird (hier a),
14  def test():                         #  ist nur innerhalb der Funktion gültig (bekannt), es sei denn,
15      global a                        #  dass mit  global a  auf das a auch außerhalb der Funktion
16      a=2                             #  zugegriffen werden kann.
17  test()
18  print(a)


19  print(ord('A'))                     #  65   ASCII-Codierung
20  print(ord('B'))                     #  66   American Standard Code for Information Interchange

21  print(chr(65))                      #  A
22  print(chr(66))                      #  B


23  print(bin(48))                      #  0b110000  Dual
24  print(hex(255))                     #  0xff   hexadezimal


25  def xggT(a,b):                      #   erweiterter euklidischer Algorithmus,  ggT(a,b) = u*a+v*b
26      x1,x2,y1,y2=1,0,0,1
27      while b:
28          q=a //  b                  #  Restlose Division
29          x1,x2,y1,y2=x2,x1-q*x2,y2,y1-q*y2
30          a,b=b,a%b
31      return a,x1,y1
32
33  print(xggT(24,16))
```

Grafik

```
1   from tkinter import *                                   #   toolkit interface
2   master = Tk( )
3   w = Canvas(master, width=600, height=400, bg='gray')
4   w.pack( )


5   w.create_rectangle(65, 35, 135, 65, fill="blue", outline="yellow")
                                                            #  obere linke Ecke,  untere rechte Ecke
6   w.create_polygon(50,30,100,20,80,100, fill="yellow",outline="black")   #   hier Dreieck

7   w.create_line(0, 0, 500, 400, fill="black", width=1)    #   von nach

8   w.create_oval(30,30, 60,50, fill="yellow")              #  obere linke Ecke,  untere rechte Ecke

9   w.create_text(120, 300, font="Purisa", text="Informatik")

10  a=300
11  b=210

12  while a<600:
13     w.create_oval(a,a,b,b,fill="yellow")
14     a = a + 15
15     b = b + 15


16  def circle(w,x,y, r):
17     return  w.create_oval(x-r,y-r,x+r,y+r, outline="blue")

18  circle(w,250,50,20)



19  import time

20  for a in range(100,200,10):
21     …
22     time.sleep(0.5)                                      #   verzögerte Ausführung der Anweisungen  ...
23     w.update( )

24  …
25  r=w.create_rectangle(a, 35, a+35, 65, fill="blue", outline="yellow")
26     time.sleep(0.5)
27     w.update( )
28     w.delete(r)                                          #    möglich    w.delete(ALL)
```

Grays

Color Name	RGB CODE	HEX #	Sample
Black	0-0-0	000000	
Dark Slate Gray	49-79-79	2f4f4f	
Dim Gray	105-105-105	696969	
Slate Gray	112-138-144	708090	
Light Slate Gray	119-136-153	778899	
Gray	190-190-190	bebebe	
Light Gray	211-211-211	d3d3d3	

z.B. … fill="#2f4f4f "

Blues

Color Name	RGB CODE	HEX #	Sample
Midnight Blue	25-25-112	191970	
Navy	0-0-128	000080	
Cornflower Blue	100-149-237	6495ed	
Dark Slate Blue	72-61-139	483d8b	
Slate Blue	106-90-205	6a5acd	
Medium Slate Blue	123-104-238	7b68ee	
Light Slate Blue	132-112-255	8470ff	
Medium Blue	0-0-205	0000cd	
Royal Blue	65-105-225	4169e1	
Blue	0-0-255	0000ff	
Dodger Blue	30-144-255	1e90ff	
Deep Sky Blue	0-191-255	00bfff	
Sky Blue	135-206-250	87ceeb	
Light Sky Blue	135-206-250	87cefa	
Steel Blue	70-130-180	4682b4	
Light Steel Blue	176-196-222	b0c4de	
Light Blue	173-216-230	add8e6	
Powder Blue	176-224-230	b0e0e6	
Pale Turquoise	175-238-238	afeeee	
Dark Turquoise	0-206-209	00ced1	
Medium Turquoise	72-209-204	48d1cc	
Turquoise	64-224-208	40e0d0	
Cyan	0-255-255	00ffff	
Light Cyan	224-255-255	e0ffff	
Cadet Blue	95-158-160	5f9ea0	

Greens

Color Name	RGB CODE	HEX #	Sample
Medium Aquamarine	102-205-170	66cdaa	
Aquamarine	127-255-212	7fffd4	
Dark Green	0-100-0	006400	
Dark Olive Green	85-107-47	556b2f	
Dark Sea Green	143-188-143	8fbc8f	
Sea Green	46-139-87	2e8b57	
Medium Sea Green	60-179-113	3cb371	
Light Sea Green	32-178-170	20b2aa	
Pale Green	152-251-152	98fb98	
Spring Green	0-255-127	00ff7f	
Lawn Green	124-252-0	7cfc00	
Chartreuse	127-255-0	7fff00	
Medium Spring Green	0-250-154	00fa9a	
Green Yellow	173-255-47	adff2f	
Lime Green	50-205-50	32cd32	
Yellow Green	154-205-50	9acd32	
Forest Green	34-139-34	228b22	
Olive Drab	107-142-35	6b8e23	
Dark Khaki	189-183-107	bdb76b	
Khaki	240-230-140	f0e68c	

Yellow

Color Name	RGB CODE	HEX #	Sample
Pale Goldenrod	238-232-170	eee8aa	
Light Goldenrod Yellow	250-250-210	fafad2	
Light Yellow	255-255-224	ffffe0	
Yellow	255-255-0	ffff00	
Gold	255-215-0	ffd700	
Light Goldenrod	238-221-130	eedd82	
Goldenrod	218-165-32	daa520	
Dark Goldenrod	184-134-11	b8860b	

Oranges

Color Name	RGB CODE	HEX #	Sample
Dark Salmon	233-150-122	e9967a	
Salmon	250-128-114	fa8072	
Light Salmon	255-160-122	ffa07a	
Orange	255-165-0	ffa500	
Dark Orange	255-140-0	ff8c00	
Coral	255-127-80	ff7f50	
Light Coral	240-128-128	f08080	
Tomato	255-99-71	ff6347	
Orange Red	255-69-0	ff4500	
Red	255-0-0	ff0000	

Pinks/Violets

Color Name	RGB CODE	HEX #	Sample
Hot Pink	255-105-180	ff69b4	
Deep Pink	255-20-147	ff1493	
Pink	255-192-203	ffc0cb	
Light Pink	255-182-193	ffb6c1	
Pale Violet Red	219-112-147	db7093	
Maroon	176-48-96	b03060	
Medium Violet Red	199-21-133	c71585	
Violet Red	208-32-144	d02090	
Violet	238-130-238	ee82ee	
Plum	221-160-221	dda0dd	
Orchid	218-112-214	da70d6	
Medium Orchid	186-85-211	ba55d3	
Dark Orchid	153-50-204	9932cc	
Dark Violet	148-0-211	9400d3	
Blue Violet	138-43-226	8a2be2	
Purple	160-32-240	a020f0	
Medium Purple	147-112-219	9370db	
Thistle	216-191-216	d8bfd8	

Farben

```
from tkinter import *
from random import *


master = Tk( )
w = Canvas(master, width=800, height=400, bg='gray')
w.pack( )



farbe = "#%02x%02x%02x"%(250,5,30)              #  rot grün blau,  jeweils  0 ... 255

w.create_oval(150,200, 250,300, fill = farbe)




for i in range(1,700,20):                       # rotgelb
   p="#"+("%06x"% randint(16000000,16777215))
   w.create_oval(30+i,40, 50+i,60, fill = p)



for i in range(1,700,20):                       # blaugrün
   p="#"+("%06x"% randint(3900000,3920000))
   w.create_oval(30+i,60, 50+i,80, fill = p)



for i in range(1,700,20):                       # blauschwarz
   p="#"+("%06x"% randint(1000,10000))
   w.create_oval(30+i,80, 50+i,100, fill = p)
```

Funktionsgraph

```
from tkinter import *
from math import *


def zeichnen( ):
   s = eingabe.get( )

   for i in range(4000):
      x = i/200.0 - 10
      f = eval(s)
      flaeche.create_line(x*20+200,200-f*20,x*20+201,200-f*20)

fenster = Tk( )
flaeche = Canvas(fenster, width=400, height=400)

eingabe = Entry(fenster, width=40)
label = Label(fenster, text='f(x)=')

button = Button(fenster,text='Graph zeichnen',command=zeichnen)
flaeche.pack( )
label.pack(side=LEFT)
eingabe.pack(side=LEFT)
button.pack(side=RIGHT,pady=10,padx=30)

flaeche.create_line(0,200,400,200)
flaeche.create_line(200,0,200,400)
```

Turtle-Grafik

```
from turtle import *

fd(100)          forward
lt(50)           left
rt(40)           right
bk(100)          backward

ht( )            hideturtle
st( )            showturtle

setpos(0,-200)          setposition(x-Koordinate, y-Koordinate)
penup( )
pendown( )

x = xcor( )             x enthält die x-Koordinate
y = ycor( )             y enthält die y-Koordinate
winkel = heading( )     Winkel wird ermittelt

seth(winkel)            setheading   Winkel (zur x-Achse) wird festgelegt

color('blue')
bgcolor('orange')       background
speed(-1)

color(.2, 0.8, 0.2)     rot, grün, blau

begin_fill( )
  ….
end_fill( )
```

String-Anweisungen

```
1  s='vlv'
2  print(s)
3  d=s.replace('v','vlw')
4  print(d)
5  print(d[0])
6  print(d[len(d)-1])
7  print(d.count('v'))
```

Zeitmessung

```
1   from time import *
2
3   t1 = clock( )
4   a = 1
5   for a in range(1,10000):
6       a = a+1
7   t2 = clock( )
8
9   t = t2 - t1
10  print("%6.4f"% t)
```

Mengen

```
11  a = {1,3,7,9}
12  b = {1,2,6,7,10}

13  a & b              Durchschnitt  {1,7}
14  a | b              Vereinigung  {1, 2, 3, 6, 7, 9, 10}
15  a - b              Differenz {9, 3}

16  a.issubset(b)      a Teilmenge von b?      True / False

17  c = a.copy( )      c = a erzeugt nur einen weiteren Pointer auf a, d.h. nur einen weiteren
                           Namen für das gleiche Objekt.
18  a.clear( )
19  print(a)
20
```

Stack

```
stack = [3, 4, 5]

d = stack.pop( )

stack.append(8)
```

Vorrangwarteschlange (Prioritätenliste, engl. priority queue)
ist eine erweiterte Form einer Warteschlange.
Den Elementen, die in die Warteschlange gelegt werden, wird ein Schlüssel mitgegeben,
der die Reihenfolge der Abarbeitung der Elemente bestimmt.

Eingabe

```
from tkinter import*

def Versuch():
    p=float(P.get())
    n=int(Anzahl.get())
    N=int(Wiederholungen.get())
    print(n, N, p)

master=Tk()
w=Canvas(master,height=600,width=1000,bg='white')
w.pack()

W=Label(master,text=' Wahrscheinlichkeit ')
W.pack(side=LEFT)
P=Entry(master,width=10)
P.pack(side=LEFT)

L=Label(master,text=' Länge n ')
L.pack(side=LEFT)
Anzahl=Entry(master,width=10)
Anzahl.pack(side=LEFT)

V=Label(master,text=' Wiederholungen ')
V.pack(side=LEFT)
Wiederholungen=Entry(master,width=10)
Wiederholungen.pack(side=LEFT)

Enter=Button(master,text='Ausführen',command=Versuch)
Enter.pack(side=LEFT)
```

Dateien

```
datei=open("DateiZahlen","w")              #  w steht für write

for n in range(10):
    datei.write(str(n))        #  Typ string erforderlich
    datei.write("\n")          #  Zeilenumbruch hinzufügen

datei.close()



text=open("DateiZahlen").readlines()
print(text)
print(text[3])



datei=open("DateiZahlen")
for line in datei:
    print(line)



datei=open("DateiZahlen")
for line in datei:
    print(line.strip("\n"))      # to strip  entfernen



datei.close()
```