Python-Einführung

Suche Python 3.4.1 download und installiere das Programm (in der Schule entfällt dieser Schritt).
Unter C: ist der Ordner Python34 zu finden und in ihm der Ordner Lib.
Rufe im Unterordner idlelib das Programm idle.py (Python Shell) auf. Eine Verknüpfung (rechte Maustaste) auf dem Desktop erspart zukünftiges Suchen. Mit dieser Shell können alle Python-Anweisungen getestet werde. Mit File / New File gelangt man zum Editor und mit Run / Run Module wird das Programm ausgeführt. Vorher muss es gespeichert werden.

Gib das Folgende in die Shell ein. Was wird ausgegeben?

```
>>> a = 5                    #  Eingabe mit Return beenden. Kommentare beginnen mit #.
                             #  SyntaxError: unexpected indent deutet auf eine Leerstelle hin.
                             #  Das Falsche kann kopiert (markieren, strg c) und neu eingegeben
                             #  werden (strg v).
>>> b = 100
>>> a**b

>>> a = a+b
>>> a = a+b
>>> a

>>> Eingabe = input( )       #  Gib dann nach Enter irgendetwas ein
>>> Eingabe

>>> print("Informatik")

>>> a < b
>>> a == 20*b                #  gleich  ==

for n in [1,2,3,4,5]:        #  Benutze nun den Editor.
    print(n)                 #  Eine  Einrückung muss sein.
                             #  Wie können die Quadratzahlen ausgegeben werden?

for n in range(1, 10):
    print(n**n)              #  Bis wohin wird n gezählt? Dies ist zukünftig zu beachten.


for n in range(1, 20, 5):
    print (n)                #  Was bedeutet die 5?


for k in range(20):
    print(k, end = " ")      #  oder   print(k, end = ' ' )


s = 0
for k in range(10):
    s = s+k
    print(s, end = ' ' )     #  Was ändert sich, falls print(s) ohne Einrückung ist?
```

1. Schreibe ein Programm, das für ein Anfangskapital K und einen Prozentsatz p das Kapital nach 1, 2, 3, ...,10 Jahren berechnet.

2. Erzeuge eine Wertetabelle für die Funktion $f(x) = x^3 - 2x^2$.
   Anfang und Schrittweite sind frei wählbar.

3. Untersuche, ob die Reihen gegen einen Grenzwert streben, d.h. ob immer mehr Dezimalstellen stabil bleiben, je mehr Summanden berücksichtigt werden.

   a) $1 + \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{16} + ...$ Erzeuge hierzu die Folge: $1,\ 1 + \frac{1}{2},\ 1 + \frac{1}{2} + \frac{1}{4},\ ...$

   b) $4 \cdot (1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - ...)$

   c) $1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5} + ...$

4. Programmiere ein Ratespiel, bei dem eine Zufallszahl geraten werden soll.

```
from random import *
n = 100
zahl = randint(1,n)              #  Zufallszahl liegt im Bereich  [1, n]


a = input("geratene Zahl: ")     #  Eingelesenes (hier a) ist vom Typ string (Zeichenkette).
                                 #  Für die Verwendung als ganze Zahl (integer) ist eine
                                 #  Typumwandlung erforderlich, z.B. mit  zahl = int(a).


if  Bedingung:
      print (" geratene Zahl ist zu groß")

                                 #  Bedingung, z.B.  a < b,  a > b,  a == b gleich
                                 #  <= kleiner gleich,  >= größer gleich,  != oder <> ungleich

break                            #  bricht eine Schleife ab

while True:                      #  Endlosschleife
    ...
```

5. Wir betrachten die Folge 2, 6, 18, ...
   Ab welcher Stelle sind die Folgenglieder erstmals größer als eine Million?
   Verwende eine while-Schleife.

6. Sieh dir diese Anweisungen an:

```
from random import *
N = 10
L = [randint(1,6)  for n in range(0, N)]        # Es wird eine Liste erzeugt.
print(L)
print(L[0])                                     # Die Indizierung beginnt stets mit 0.
print(L[1])
print(len(L))
```

   Ermittle absolute und relative Häufigkeiten der 6 für N = 6000.

7. Erzeuge eine Liste mit 50 Zufallszahlen aus dem Bereich [1, 1000].
   Die größte Zahl soll gesucht werden, ohne die Liste zu verändern.

8. Erzeuge eine Liste mit 10 Zufallszahlen aus dem Bereich [1, 100].

   b) Es sollen die ersten beiden Zahlen vertauscht werden.

   c) Die ersten beiden Zahlen sollen nur vertauscht werden, wenn die erste Zahl größer als die zweite ist.

   d) Die Zahlen in der Liste sollen aufsteigend sortiert werden.

9. a) Teste die Anweisungen: 15 % 4 und 20 % 8

   b) Zwei Zahlen m und k werden eingelesen. Es soll überprüft werden, ob m durch k teilbar ist.

   c) Eine eingelesene Zahl soll überprüft werden, ob eine Primzahl vorliegt.

   d) Ermittle alle Primzahlen und deren Anzahl im Intervall [2, 100].

   e) Ermittle alle Primzahlen und deren Anzahl in einem vorgegebenen Intervall [a, b].

10. 3n+1-Problem von Collatz

Betrachte die Zahlenfolge mit folgendem Bildungsgesetz:
Beginne mit irgendeiner natürlichen Zahl n.
Ist n gerade, so nimm als Nächstes n/2.
Ist n ungerade, so nimm als Nächstes 3n+1.
Wiederhole die Vorgehensweise mit der erhaltenen Zahl, falls sie ungleich 1 ist.

Für n = 6 entsteht die Folge
6, 3, 10, 5, 16, 8, 4, 2, 1
Schreibe ein Programm, mit dem du ermitteln kannst, wie viele Elemente die Folge hat, die mit deinem Geburtsjahr beginnt.

11. Fibonacci-Folge

Die ersten Elemente dieser Folge lauten:
1, 1, 2, 3, 5, 8, 13, ...
Untersuche die Quotientenfolge a_{n+1}/a_n, die aus jeweils aufeinanderfolgenden Elementen gebildet wird.

Für mathematisch Interessierte:
Ein Briefträger steigt täglich eine lange Treppe nach folgendem Muster empor:
Die erste Stufe betritt er in jedem Fall. Von da an nimmt er jeweils nur eine Stufe oder aber zwei Stufen auf einmal.
Auf wieviel verschiedene Arten kann der Briefträger die k-te Stufe erreichen?

Funktionen, zunächst ohne Rückgabewert (Prozeduren)

```
def Aussage( ):
  print("Informatik macht Spaß")


Aussage( )
Aussage( )
Aussage( )
```

```
def Teiler(n):
  for k in range(2, n):
    if n % k == 0:
      print(k)

Teiler(20)
Teiler(50)
```

```
def Fakultaet(n):
  produkt=1
  for k in range(2, n+1):
    produkt=produkt*k
  print(produkt)


Fakultaet(10)
Fakultaet(20)

for k in range(5, 15):
  print(k), Fakultaet(k)
```

Wertetabelle

```
def fkt(x):
    return x**2

links = float(input("von: "))
rechts = float(input("bis: "))
d = float(input("Schrittweite: "))

for n in range( int( (rechts - links)/d + 1) ):
    print ("P(","%.2f "%links,"|", "%.3f "%fkt(links),")")     #  print (links, fkt(links))
    links = links+d
```