Musterlösung zu Übungsblatt 2

Thomas Graf
EF / WF Informatik 2018-2019
Programmieren in Python I

15. September 2018

1 Erstes Programm ($\star$)

keine Musterlösung nötig

2 Strings und Indizes ($\star$)

```
wort = 'Informatik'

print(wort[:4]) # gib die ersten 4 Buchstaben aus
# Output wird sein: Info
```

3 Input ($\star$)

```
a = int(input()) # Einlesen der Zahl a
b = int(input()) # Einlesen der Zahl b

# Computer wartet nun auf die Eingabe von a und b
# ueber die Tastatur

print(a+b) # gib die Summe a + b aus
```

4 Fahrenheit zu Celsius ($\star$)

```
F = 134 # setzte den Wert der Temperatur in Fahrenheit

C = (F - 32.)*(5./9.) # Umrechnung zu Celsius

print(F,'Grad Fahrenheit sind', round(C,3),'Grad Celsius')
```

5 Fakultät ($\star\star$)

```
def factorial(n):
    """ berechnet n! (n-Fakultaet) fuer gegebenes n
    """
    fak = 1
    for k in range(2,n+1):
        fak = fak * k
    return(fak)
```

6 Potenzieren ($\star\star$)

```
def power_8(a):
    """ berechnet a^8 fuer gegebenes a
    """
    # einfache Anwendung des Potenzgesetzes:
    # a^b * a^c = a^(b+c)
    b = a*a # b == a^2 (erste Multiplikation)
    c = b*b # c == a^4 (zweite Multiplikation)
    d = c*c # d == a^8 (dritte Multiplikation)
    return(d)
```

7 Algebraische Gleichung ($\star\star\star$)

```
from numpy import sqrt # importiere die Wurzel-Funktion

def solve_quadratic_equation(a, b, c):
    """ berechnet (falls sie exisiteren)
        die Loesungen / Loesung der quadratischen Gleichung
```

```
6           ax^2 + bx + c = 0
7           fuer beliebige reelle Zahlen a, b, c
8       """
9
10      # quadratischer Fall (quadratische Gleichung)
11      if a != 0:
12          # berechne die Diskriminante
13          D = b**2. - 4.*a*c
14
15          # keine reellen Loesungen fuer D < 0
16          # (Wurzel einer negativen Zahl)
17          if D < 0:
18              print('negative Diskriminantne D =',D,'\n')
19              print('keine reellen Loesungen')
20              return
21          # Mitternachtformel
22          # es gibt zwei verschiedene Loesungen fuer D != 0
23          # und genau eine (doppelte) Loesung fuer D == 0
24          else:
25              wd = sqrt(D) # wd = Wurzel der Diskriminante
26              x1 = (-b + wd)/(2.*a) # Mitternachtsformel
27              x2 = (-b - wd)/(2.*a) # Mitternachtsformel
28              return([x1, x2])
29
30      # linearer Fall (lineare Gleichung)
31      elif b != 0:
32          x = -c/float(b) # Loesung einer linearen Gleichung
33          return(x)
34
35      # konstanter Fall
36      elif c != 0:
37          print('Fall c = 0, mit c ==',c,':\n')
38          print('diese Gleichung ist nie erfuellt')
39          return
40      else:
41          print('Fall c = 0, mit c ==',c,':\n')
42          print('diese Gleichung ist trivialerweise erfuellt')
43          return
```

```
44
45 solutions = solve_quadratic_equation(3,-4,-15) # Beispiel
```

8 Perfekte Zahlen (⚛)

```
1  def is_perfect(n):
2      """ entscheidet, ob eine gegebene ganze Zahl n
3          eine perfekte Zahl ist oder nicht
4      """
5      # eine perfekte Zahlen muessen groesser sein als 0:
6      if n <= 0:
7          return(False)
8
9      # summiere alle echten Teiler der Zahl n
10     sum = 0
11     for i in range(1,n):
12         if n % i == 0:
13             sum += i
14     # pruefe, ob die Summe der Teiler von n gerade
15     # der Zahl n selbst entspricht (Definition einer
16     # perfekten Zahl)
17     return(sum == n)
18
19
20 def perfect_numbers(n1, n2):
21     """ printet alle perfekten Zahlen in
22         [n1,n2], mit n1 < n2, in aufsteigender
23         Ordnung
24     """
25     for k in range(n1,n2+1):
26         # pruefe fuer jede Zahl k in [n1,n2]
27         # ob sie perfekt ist oder nicht
28         # k wird ausgebeben <=> k ist perfekt
29         if is_perfect(k) == True:
30             print(k)
31
32 perfect_numbers(1,100000)
```