cap_sage_ex

Übungsaufgabe 1

```
def exS(n):
    s = 0
    for k in range(1,n+1):
        s = s + k**2
    return s
```

```
for k in range(1,11): print exS(k)
```

```
1
5
14
30
55
91
140
204
285
385
```

```
exS(100)
```

```
338350
```

```
exS(1000)
```

```
333833500
```

```
k = var('k')
sum(k**2,k,1,100)
```

```
338350
```

Übungsaufgabe 2

```
memo = {0:0, 1:1}
def fib3(n):
    if n not in memo:
        if n%2 == 0: return fib3(n/2)*(fib3(n/2) + 2*fib3(n/2-
1))
        else: return fib3((n+1)/2)**2 + fib3((n-1)/2)**2
    return memo[n]
```

```
fib3(1000)
```

```
43466557686937456435688527675040625802564660517371780402481725554179490518904038798400792551692959225930803226347752096896332247116164299644090653318793829896964992851600370447613779528875
```

```
fibonacci(1000)
```

```
43466557686937456435688527675040625802564660517371780402481725554179490518904038798400792551692959225930803226347752096896332247116164299644090653318793829896964992851600370447613779528875
```

```
timeit('fib3(10**5)')
```

```
5 loops, best of 3: 42.1 s per loop
```

```
timeit('fibonacci(10**5)')
```

```
625 loops, best of 3: 254 µs per loop
```

Übungsaufgabe 3

```
def mod1(a,b):
    #berechnen a mod b;
    res = a
    while res >= b:
        res -= b
    return res
```

```
10%3
```

```
1
```

```
mod1(10,3)
```

```
1
```

```
mod1(9,3)
```

```
0
```

```
mod1(1234567,1234)
```

```
567
```

```
1234567%1234
```

```
567
```

Übungsaufgabe 4

```
def ggT(a,b):
```

```
    if a<b: return ggT(b,a)
    elif b == 0: return a
    return ggT(b,a%b)
```

```
ggT(12345678,234)
```

```
18
```

```
gcd(12345678,234)
```

```
18
```

```
a = ZZ.random_element(10**1000)
b = ZZ.random_element(10**1000)
```

```
import sys
sys.setrecursionlimit(10000000)
print "erste Zahl: ", a
print "zweite Zahl: ", b
print "ggT ist gleich : ", ggT(a,b)
```

```
erste Zahl:
9817792315013134927644921702841989690302362586723142812440874
5476293036735441293822702782137607800756847298705908144598468
6055322255489622108130256573900218013529154128847014842383617
2542206661668813308472450596897863310904470756084203719426719
4720958209062797450505293733133334800246954171442253152928478
4178379014387120133146789292352730609042362966724598185977186
5361962023905670854055049805859387355733813923360808638981725
8802499867668632104935209362865546329539476901251942390576062
3074879168618973725095770633765929386172086311634106405539969
5883945912815331532307680335422773855980985687911923748028140
0459154080538516258536048608964258537428749791010801438026547
9488778986099599776857863679635675646275429620831179941219702
9131489686823423237258193142999682818513396696281664151254993
6582431811930976796896253106271064023777115917111796808526684
0670449633551146511171828573320444917788450322 6
zweite Zahl:
1041259036560256851073208981710332863197190900394017361580377
7457269577525232424885428185376493467921596454608216749151775
4003402031121938175594743575172929335174972259740375575603915
8884071937030785980482945269976260722237566287475277120309016
9386996631426806010628191166630899385195256298444907498899652
8710781207653679883708763368043784358488015624146716714197817
3943450261077805465229531328204153249402795213956865105136562
9972560171582579734872335292555693137120979052301228278171891
1555933557845923313144038540662030442513909718818672191124942
7123878701442396053432091426792650164986000591766194343447359
4458704344455285077917877129367749109585913618367946232566633
9190171631834027069852591642927984293916497844436287824714714
3265933134100318538479112420774843881091807485080562041600532
```

```
548089713528940683532906775569176435744817630370464953026047407731604293705995775861141957085172517145490669 4
ggT ist gleich :  2
```

```
gcd(a,b)
```

```
2
```

Hausaufgabe 1

```
def nextPrimeTwin(n):
    p = next_prime(n)
    np = next_prime(p)
    while not (p+2 == np):
        p = np
        np = next_prime(p)
    return [p,np]
```

```
p,q = nextPrimeTwin(10000000)
```

```
print p
print q
```

```
10000139
10000141
```

Hausaufgabe 2

```
def reverse1(ll):
    if len(ll)<2: return ll
    first = ll[0]; rest = ll[1:]
    return reverse1(rest)+[first]
```

```
liste = [ZZ.random_element(100) for _ in range(100)]; liste
```

```
[85, 40, 89, 28, 10, 52, 0, 47, 94, 50, 35, 37, 75, 2, 40, 61
95, 62, 21, 66, 39, 83, 96, 47, 53, 75, 58, 17, 65, 87, 59, 6
14, 73, 53, 33, 25, 1, 31, 36, 0, 40, 84, 58, 26, 78, 58, 71,
39, 64, 43, 34, 38, 13, 89, 80, 5, 26, 93, 39, 97, 88, 36, 47
54, 98, 29, 70, 5, 6, 1, 64, 22, 86, 53, 66, 4, 99, 58, 96, 1
34, 57, 13, 79, 32, 11, 79, 16, 76, 36, 66, 29, 90, 99]
```

```
rev_liste = reverse1(liste);rev_liste
```

```
[99, 90, 29, 66, 36, 76, 16, 79, 11, 32, 79, 13, 57, 34, 18,
58, 99, 4, 66, 53, 86, 22, 64, 1, 6, 5, 70, 29, 98, 54, 92, 4
```

```
88, 97, 39, 93, 26, 5, 80, 89, 13, 38, 34, 43, 64, 39, 72, 71
78, 26, 58, 84, 40, 0, 36, 31, 1, 25, 33, 53, 73, 14, 20, 68,
87, 65, 17, 58, 75, 53, 47, 96, 83, 39, 66, 21, 62, 95, 60, 6
2, 75, 37, 35, 50, 94, 47, 0, 52, 10, 28, 89, 40, 85]
```

Übungsaufgabe 5

```
def PowerMod1(a,n,p):
    #(a**n)mod p
    if n == 0: return 1
    if n%2 == 0: return (PowerMod1(a,n/2,p)**2)%p
    else: return (PowerMod1(a,n-1,p)*a)%p
```

```
power_mod(3,2,5)
```

```
4
```

```
PowerMod1(3,2,5)
```

```
4
```

```
a = ZZ.random_element(10**100)
n = ZZ.random_element(10**100)
p = ZZ.random_element(10**100)
sys.setrecursionlimit(1000000)
print "we derive: ",a," ** ",n," mod ",p
print "our function: ", PowerMod1(a,n,p)
print "build-in function", power_mod(a,n,p)
```

```
we derive:
6864602542493006005471929994040904596662123239420849594036592
7682091305835719420899757376339  **
1789608950326904525682345521577819564027657894013490186423829
4299726395884134182025803674411 1  mod
9225235399524717856980927907941060429372587303357830632549378
6723346241452717218349530907478 3
our function:
2134013014027432107215331914773537472311591075205713216704726
2146869338324196537794663776837 3
build-in function
2134013014027432107215331914773537472311591075205713216704726
2146869338324196537794663776837 3
```

Übungsaufgabe 6

```
def EEA(x,y):
    if y<x: return EEA(y,x)
```

```
    if y%x == 0: return [x,1,0]
    g,ss,tt = EEA(y%x,x)
    s = tt-ss*y.quo_rem(x)[0]
    t = ss
    return [g,s,t]
```

```
EEA(1234,56789)
```

```
    [1, -24989, 543]
```

```
xgcd(1234,56789)
```

```
    (1, -24989, 543)
```

```
EEA(12347394837*56784394830489,56784394830489)
```

```
    [56784394830489, 1, 0]
```

Hausaufgabe 4

```
def chrem(llist,mlist):
    m = prod(mlist)
    slist = []
    mulist=[]
    for k in range(len(mlist)):
        mulist.append(m/mlist[k])
        slist.append(xgcd(mulist[k],mlist[k])[1])
    result = sum( [ llist[i]*slist[i]*mulist[i] for i in
range(len(mlist))] )%m
    return result
```

```
chrem([2,3,1],[3,4,7])
```

```
    71
```

```
crt([2,3,1],[3,4,7])
```

```
    71
```

Übungsaufgabe 7

```
def ISBN(liste):
    return sum( [ (k+1)*liste[k] for k in range(9) ] )%11
```

```
ISBN([3,5,4,0,2,9,8,9,4])
```

```
    0
```

```
ISBN([3,5,2,8,0,6,7,5,2])
```

```
    7
```

Übungsaufgabe 8

```
def EAN(liste):
    return (-sum( liste ) - 2*sum( [ liste[k] for k in
range(1,12,2) ] ))%10
```

```
EAN([9,7,8,3,5,4,0,2,9,8,9,4])
```

```
6
```

Übungsaufgabe 9

```
def modinv(e,p):
    g,s,t = xgcd(e,p)
    if g != 1: print "Es gibt kein Inverses"; return
    else: return s%p
```

```
modinv(1234,56789)
```

```
31800
```

```
power_mod(1234,-1,56789)
```

```
31800
```

```
modinv(12345,67890)
```

```
Es gibt kein Inverses
```

```
power_mod(12345,-1,67890)
```

```
Traceback (click to the left of this block for traceback)
...
ZeroDivisionError: Inverse does not exist.
```

```
```