Frekvence vodilne števke

```
#!/usr/bin/env python3

def zaporedje(n):
    res = [n]
    while n >  1:
        yield n
        if n % 2:
            n = 3*n + 1
        else:
            n = n // 2
    yield 1

def frekvence(n):
    res = {}
    for z in zaporedje(n):
        ch = str(z)[0]
        res[ch] = res.get(ch,0) + 1
    return res

if __name__ == '__main__':
    n = int(input('n = '))
    res = frekvence(n)
    res = dict(sorted(res.items()))
    print(res)
```

Pascalov trikotnik

```
#!/usr/bin/env python3

def pascal(n):
    mem = [1]
    print(mem)
    for _ in range(n):
        a = [0] + mem
        b = mem + [0]
        mem = list([x + y for x, y in zip(a,b)])
        print(mem)

if __name__ == '__main__':
    n = int(input('n = '))
    pascal(n)
```

Pristanišče

```
#!/usr/bin/env python3
import math
import matplotlib.pyplot as plt
import numpy as np

x1 = -3
x2 =  4
y1 =  0
y2 =  0


def tocka(x):
    return np.sqrt((x1 - x)**2 + y1**2) + np.sqrt((x2 - x)**2 + y2**2)


def minam(f, a, b):
    while abs(b - a) > 0.0001:
        d = (b - a)/3
        t1, t2 = a + d, b - d
        y1, y2 = f(t1), f(t2)
        if y1 < y2:
            b = t2
        else:
            a = t1
    return (t1 + t2)/2


if __name__ == '__main__':
    y1, y2 = map(int, input('y1 y2 -> ').split())
    x = minam(tocka, x1, x2)
    plt.axis('equal')
    plt.scatter([x1, x, x2],[y1, 0, y2])
    plt.plot([x1, x, x2],[y1, 0, y2])
    plt.plot([x1 - 1, x2 + 1], [0, 0])
    X = np.linspace(x1 - 1, x2 + 1)
    Y = tocka(X)
    plt.plot(X, Y/4)
    print(x)
    plt.scatter([x], [tocka(x)/4])
    plt.show()
```

```
         Pari
```

```
#!/usr/bin/env python3

def getsums(nums, target):
    lookup=set(nums)
    res=[]
    for n in nums:
        if target - n in lookup:
            res.append((n,target-n))
            lookup.discard(n)
            lookup.discard(target-n)
    return(res)

if __name__ == '__main__':
    l = input('seznam, vsota: ')
    l=l.split(',')
    nums = [int(x.strip()) for x in l[0].split()]
    target = int(l[1].strip())
    print(getsums(nums,target))
```

```
          Vsota elementov gnezdenega seznama
```

```
#!/usr/bin/env python3

def nest_(s):
    for k in s:
        if type(k) == list:
            yield from nest_(k)
        else:
            yield k

def sum_(s):
    ss = 0
    for x in nest_(s):
        ss += x
    return ss


def sum__(s):
    ss = 0
    for k in s:
        if type(k) == list:
            ss += sum__(k)
        else:
            ss += k
    return ss



if __name__ == '__main__':
    s = [[1,4,2,[4,5,[3,1]],3,1,3],4,9,2,[8,9,3,[1,2]]]
    print(sum_(s))
    print(sum__(s))
```

```
          Faktorizacija
```

```
#!/usr/bin/env python3
from math import sqrt


def factorize(n):
    sieve = [True] * (n + 1)

    for x in range(2, int(len(sieve) ** 0.5) + 1):
        if sieve[x]:
            for i in range(x + x, len(sieve), x):
                sieve[i] = False

    lowerPrimes = [i for i in range(2, len(sieve)) if sieve[i] and (n % i == 0)]
    return lowerPrimes

def factors(n):    # (cf. https://stackoverflow.com/a/15703327/849891)
    j = 2
    while n > 1:
        for i in range(j, int(sqrt(n + 0.05)) + 1):
            if n % i == 0:
                n //= i ; j = i
                yield i; break
        else:
            if n > 1:
                yield n; break

if __name__ == '__main__':
    n = int(input("n = "))
    print(factorize(n))
    print(list(factors(n)))
```

```
          Tarča
```

```
#!/usr/bin/env python3
import math
import numpy as np
import matplotlib.pyplot as plt

def f(x, a):
    return a * x - np.exp(-x/10) * x**2


def bisek(a1, a2, t):
    h1, h2 = f(t, a1), f(t, a2)
    if h1 * h2 > 0:
        return None
    while abs(a1 - a2) > 0.0001:
        a3 = (a1 + a2) / 2
        h3 = f(t, a3)
        if h1 * h3 < 0:
            a2 = a3
        else:
            a1 = a3
    return (a1 + a2) / 2


if __name__ == '__main__':
    a1, a2 = 1, 1.1
    x1, x2 = 0, 1.3
    t = 1.2
    x = np.linspace(x1, x2)
    y1 = f(x, a1)
    y2 = f(x, a2)

    a3 = bisek(a1, a2, t)
    print(a3)

    y3 = f(x, a3)

    plt.plot(x, y1)
    plt.plot(x, y2)
    plt.plot(x, y3)
    plt.plot([x1,x2],[0,0])
    plt.scatter([t],[0])
    plt.show()
```

Uravnotežen trojiški sestav

```
#!/usr/bin/env python3


digits = '0123456789ABCDEF'
digits = dict(enumerate(digits))
nums = {v: k for k, v in digits.items()}


#print(digits, nums)


def dec_to_base(base, num):
```

```
    res = []
    while num:
        res.append(num % base)
        num //= base
    return ''.join([digits[x] for x in list(res[::-1])])


def base_to_dec(base, num):
    num = [nums[x] for x in num]
    res = num[0]
    for i in num[1:]:
        res = res * base + i
    return res


tri_digit = {'+': 1, 'o': 0, '-': -1}
digit_tri = {1: '+', 0: 'o', -1: '-'}

def to_tri(n):
    res = ''
    while True:
        a = n // 3
        b = n % 3
        if b == 2:
            res += digit_tri[-1]
            a += 1
        else:
            res += digit_tri[b]
        if a == 0:
            return res[::-1]
        n = a

def to_dec(s):
    print(s)
    res = 0
    res = tri_digit[s[0]]
    for x in s[1:]:
        res = res * 3 + tri_digit[x]
    return res

if __name__ == '__main__':
    base, num = [int(x) for x in input('base num -> ').split()]
    num1 = dec_to_base(base, num)
    print(num1)
    print(base_to_dec(base, num1))
    num2 = to_tri(num)
    print(num2)
    print(to_dec(num2))
```