# Wykład 6

- Zastosowania rekurencji

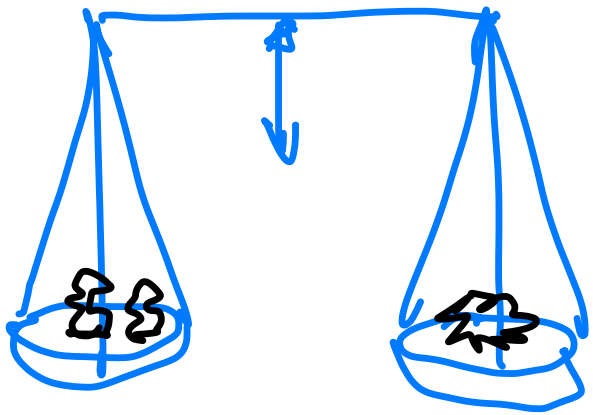
# Waga szalkowa

**Problem:**

**Dany jest zestaw odważników. Jakie ciężary można odważyć z użyciem tych odważników?**

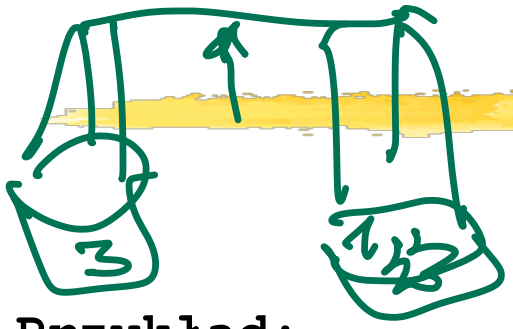
**Przykład:**

**odw = [1,3,5,7,10,24]**



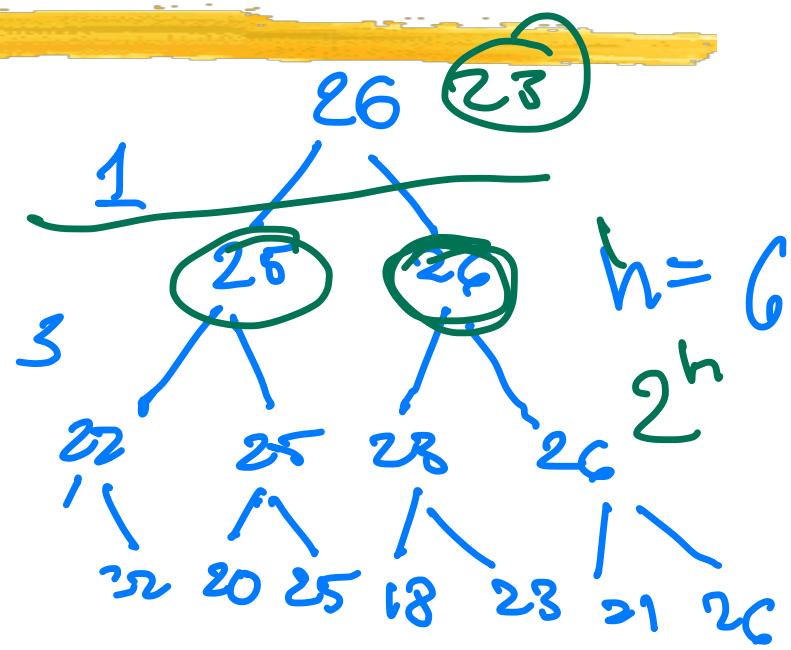
```
def waga1(tab,n):
  s = 0
  for i in range(len(tab)-1,-1,-1):
    if s+tab[i]<=n:
      s += tab[i]
    if s==n:
      return True
  return False
#end
```

# Waga szalkowa

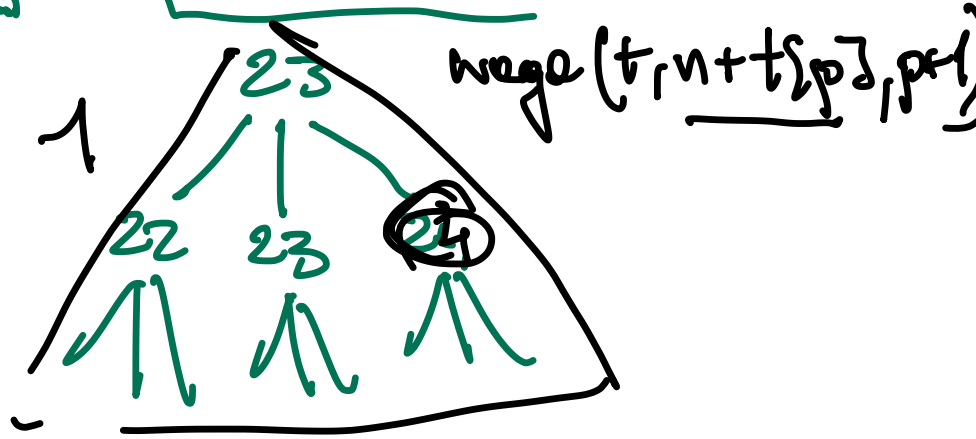


Przykład:

```
odw = [1,3,5,7,10,24]
```



```
def waga(t,n,p=0):
    if n==0: return True
    if p==len(t): return False
    return waga(t,n-t[p],p+1) or waga(t,n,p+1)
# end def

for w in range(1,50):
    print(w,waga(odw,w))
```



odw = [1,3,5,7,10,24]

13 [3,10]

[-1,-10,24]

Jak otrzymać rozwiązanie?

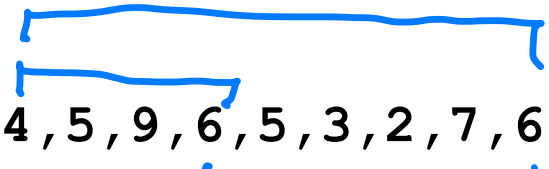
```
def wage(t, n, p=0, ves=[]):
    if n==0:
        print(ves)
        return
    if p==len(t):
        return
    wage(t, n-t[p], p+1, ves+[t[p]])
    wage(t, n, p+1, ves)
    wage(t, n+t[p], p+1, ves+[-t[p]])
```

# Przykład - pary

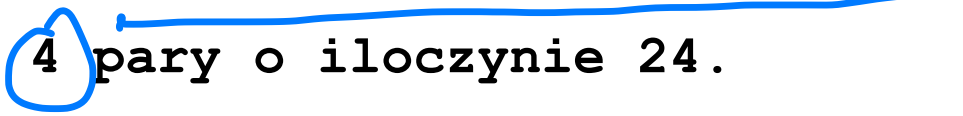
**Problem:**

**Dana jest tablica/lista z liczbami naturalnymi. Należy policzyć ile jest par elementów o określonym iloczynie.**

**Przykład:**



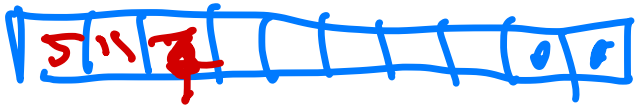
**t = [4,1,5,7,9,4,5,9,6,5,3,2,7,6,1,1,7,9,9,1]**



**Są 4 pary o iloczynie 24.**

# Przykład – Licz pary

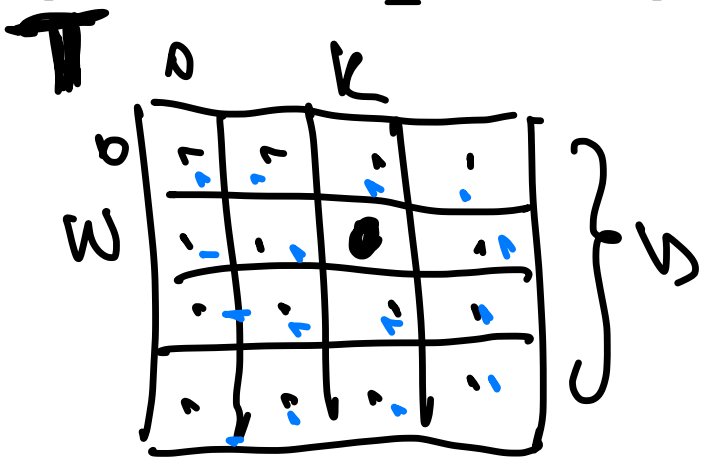
```
import random

def generuj(n):
  return [ random.randint(1,9)) for _ in range(n) ]
# end def

def licz_pary(t,s):
    n = len(t)
    licz = 0
    for i in range(n-1):
        for j in range(i+1,n):
            if t[i]*t[j]==s:
                licz = licz+1
    print("pary",licz)
# end def

t = generuj(20)
licz_pary(t,24)
```

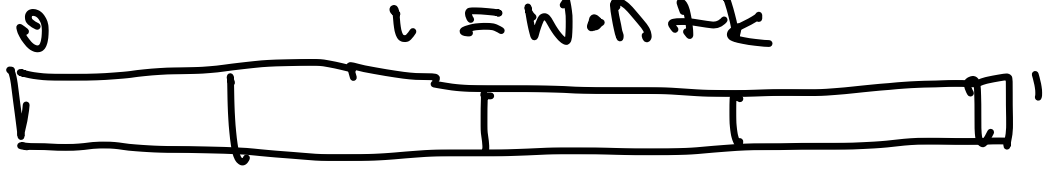


if n%t[i]==0:

# Licz pary w tablicy 2-wymiarowej

```
N = 100
t = [[randint(1,9) for _ in range(N)] for _ in range(N)]

for i in range(N):
    for j in range(N):
        for k in range(N):
            for l in range(N):
                if i!=k or j!=l:
                    if t[i][j]*t[k][l]==s:
                        licz += 1
print(licz//2)
```



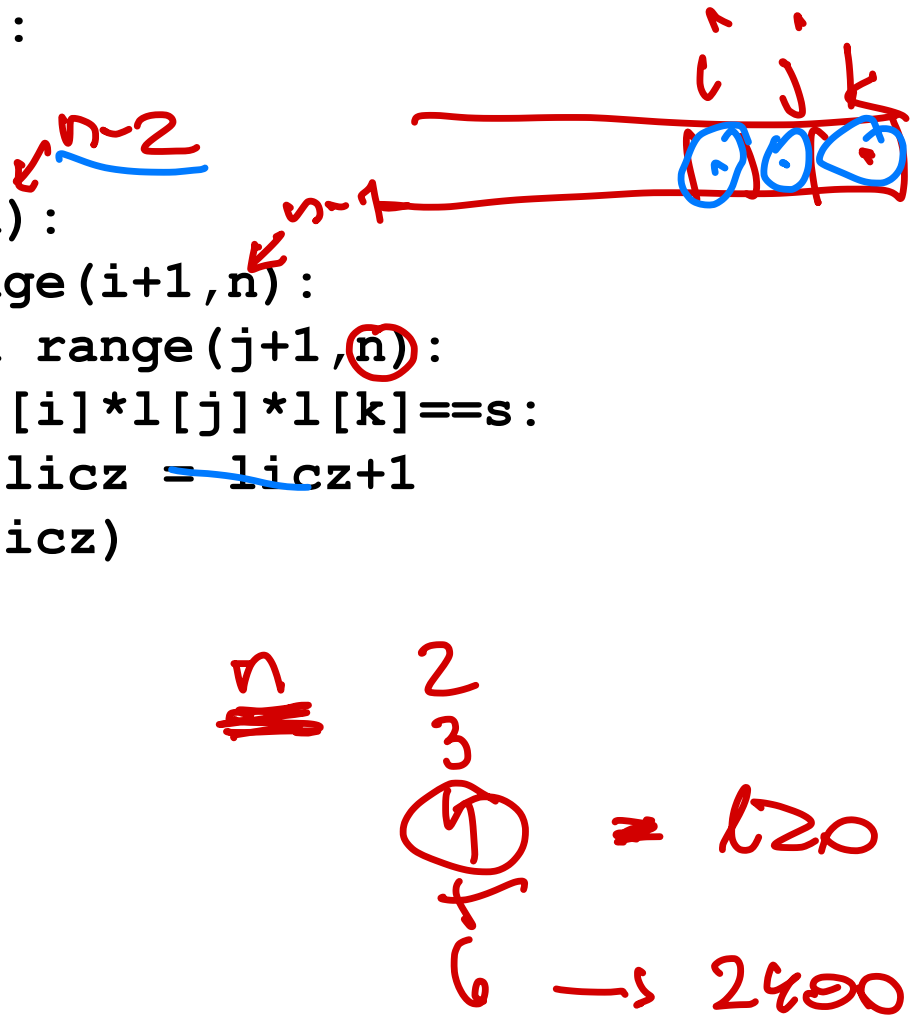




```
for i in range(0,n·n-1):
    for j in range(i+1,n·n):
        ...
```

```
w = p//n
k = p%n
```

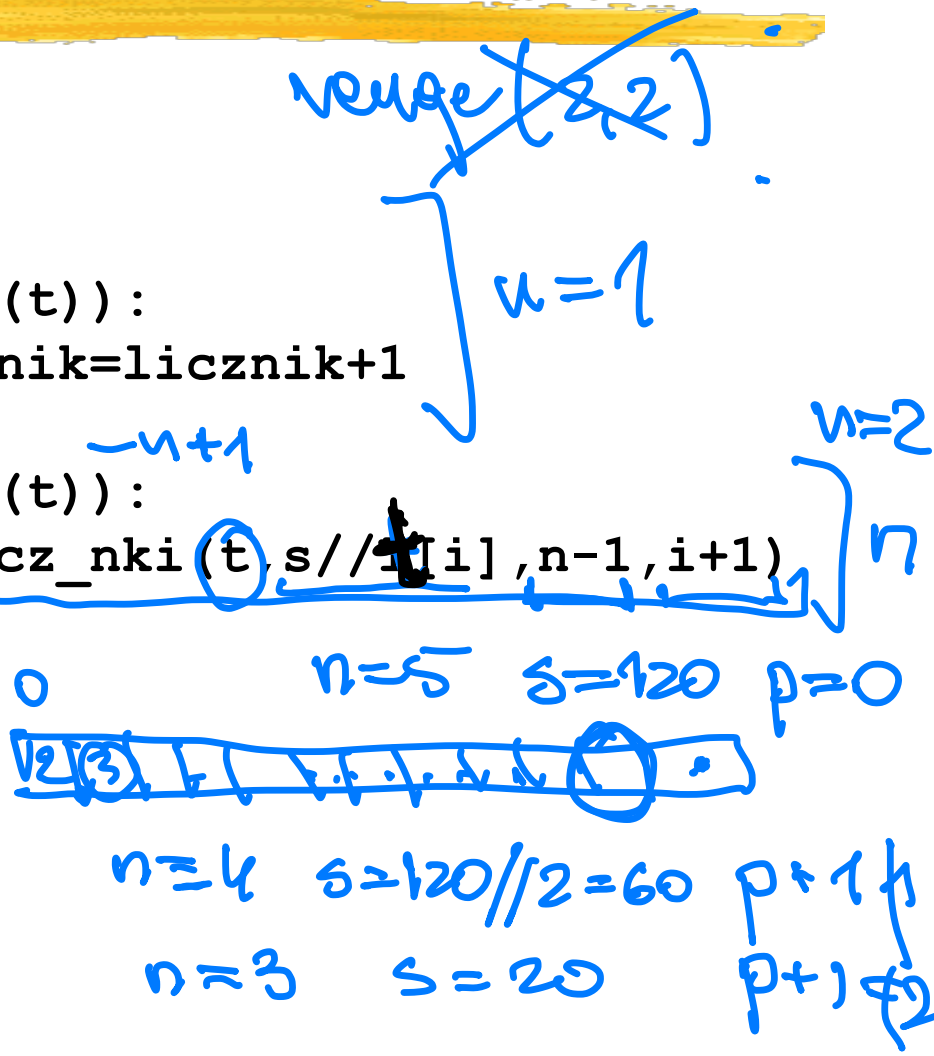
# Przykład – Licz trójki

```
def licz_trojki(t,s):
    n = len(t)
    licz = 0
    for i in range(n):
        for j in range(i+1,n):
            for k in range(j+1,n):
                if l[i]*l[j]*l[k]==s:
                    licz = licz+1
    print("trojki",licz)
# end def


t = generuj(20)
licz_trojki(t,24)
```



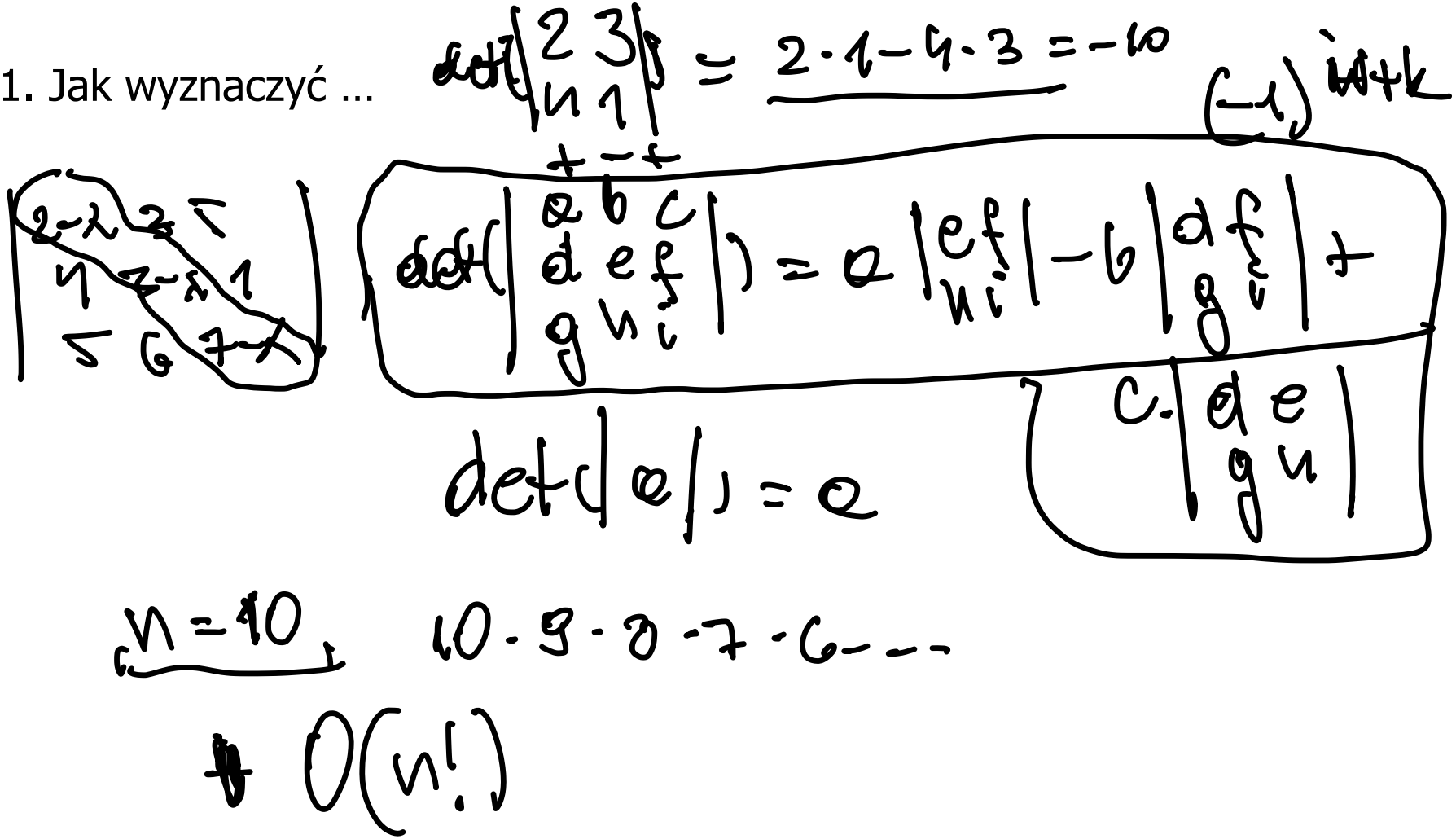
# Przykład – Licz n-ki

```
def licz_nki(t,s,n,p):
    global licznik
    if n==1:
        for i in range(p,len(t)):
            if t[i]==s: licznik=licznik+1
    else:
        for i in range(p,len(t)):
            if s%t[i]==0: licz_nki(t,s//t[i],n-1,i+1)
# end def

t = generuj(20)

licznik=0
licz_nki(t,24,4,0)
print("nki",licznik)
```



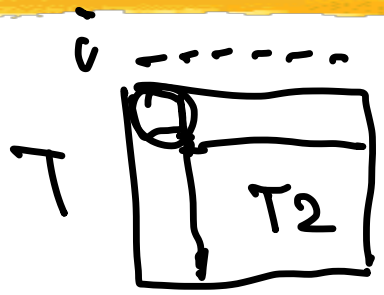
# Wyznacznik z macierzy

1. Jak wyznaczyć ...

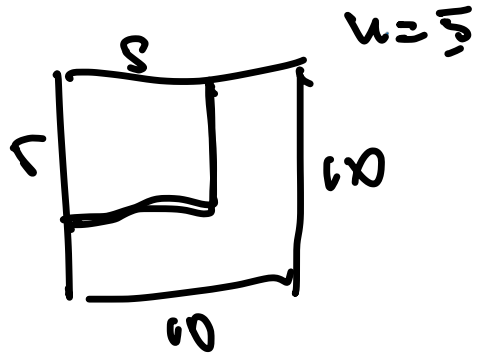


# Wyznacznik z macierzy

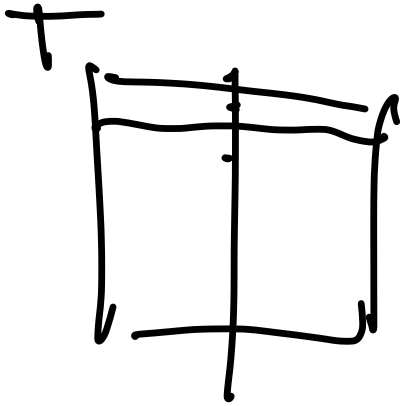
1. Jak wyznaczyć ...




```
def det(T):
    n = len(T)
    if n == 1: return T[0][0]
    S = 0; zn = 1
    for i in range(n):
        T2 = make(T, i)
        S = S + det(T2) * zn * T[0][i]
        zn = -zn
    return S
```




```
def make(T, i):
    n = len(T)
    T2 = [[0 for _ in range(n-1)]
          for _ in range(n-1)]
```




alternative

```
T2 = T.copy()
Te.del(ø)
```

Pyt

# Pytania i zadania

1. Rozwiązać dzisiejsze zadania bez użycia rekurencji.