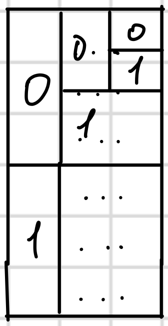
Комбинаторика

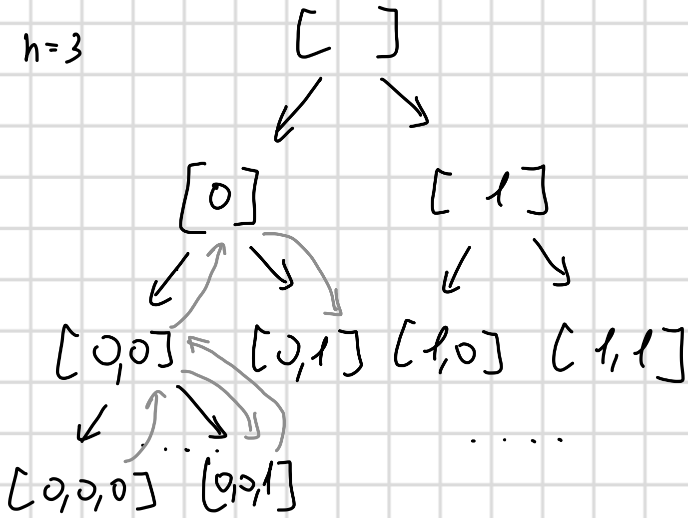
двоичные вектора фикс. длины

$\mathbb{B}^n$

000
001
...
...
111



n=3



псевдо псевдо код

gen

gen all, starting with 0

gen all, starting with 1

```
gen(p):
    if len(p) == n:
        print(p)
        return
    gen(p + [0])
    gen(p + [1])
```

```
main:
    gen([])
```

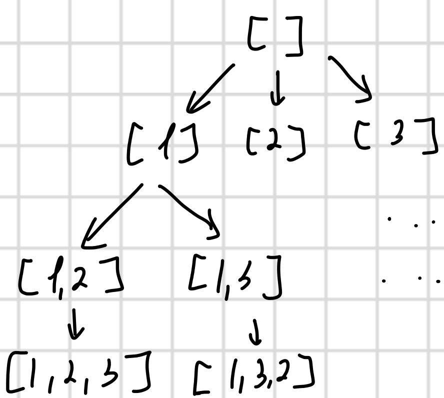
Перестановки

n=3

123
132

213
231

312
321



n эл-ов $\{1 \dots n\} = M$

$$P_n = \{ v \in M^n : v_i \neq v_j \text{ при } i \neq j \}$$

$n = 3$

123	1
132	2
213	3
231	4
312	5
321	6

$|P_n| = ?$

$|P_1| = 1$

$|P_2| = 2$

$$\overbrace{\begin{array}{l} 1 \\ 1 \\ 1 \\ \vdots \\ 1 \\ \hline 2 \\ \vdots \\ 2 \\ \hline \vdots \\ n \\ \hline \end{array}}^{n} \qquad \begin{array}{l} \updownarrow |P_{n-1}| \\ \\ \\ \updownarrow |P_{n-1}| \end{array}$$

при 1-ом известном элементе

Правило суммирования:

$$A = B \sqcup C$$

$$|A| = |B| + |C|$$

$$|P_n| = \sum_{k=1}^{n} |P_{n-1}| = n \cdot |P_{n-1}|$$

$$|P_n| = n!$$

```
gen(p):
    if len(p) == n:
        print(p)
        return
    for i = 1...n:
        if i not in p:
            gen(p+[i])
```

```
main:
    gen([])
```

∀ строки из 0 и 1 длины $\leq n$

$\bigcup_{k=0}^{n} \mathbb{B}^k$ $\qquad \varepsilon$ – пуст. стр.

0
00
000
001
01
010
011

```
gen(p):
    print(p)
    if len(p) == n: return
    gen(p+[0])
    gen[p+[1])
```

gen(p) $\quad$ p – префикс К.О. $\qquad$ К.О. – комбинаторный объект

$\exists C \subset \Sigma^*$

```
gen(p)
    if p - K.O : print(p)
    for c in Σ:      sorted
        if p+[c] – префикс K.O.:
            gen(p+[c])
```

(пример: 1) строки пост длины

$\Sigma = \mathbb{B}$

2) $\leq n$

3) перестановка

$\Sigma = \{1 \dots n\}$)

Размещения

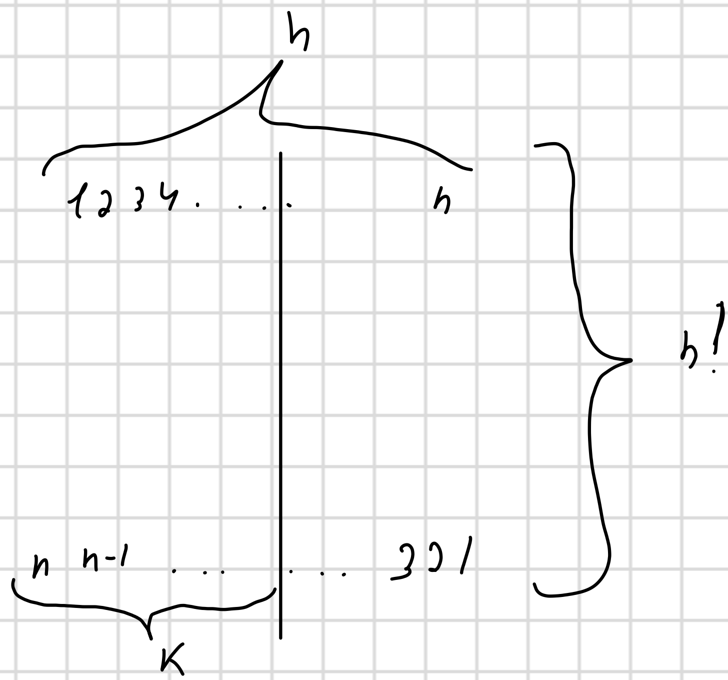
n эл. k мест $\quad k \le n$

$M = \{1 \dots n\}$

$\{ v \in M^k : v_i = v_j$ при $i \neq j \}$

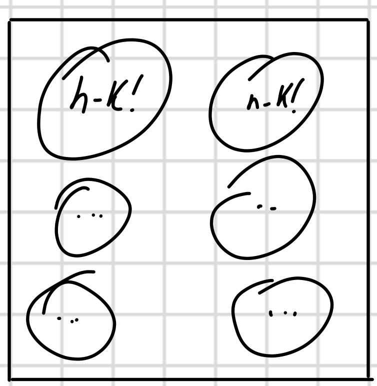
Д $n = 4, \ k = 2$

1	12
2	13
3	14
4	21
5	23
6	24
	⋮
12	43



хвостов: $(n-k)!$

$$\frac{n!}{(n-k)!}$$



$$A_n^k = \frac{n!}{(n-k)!}$$

Сочетания

$\begin{matrix} n \\ k \end{matrix} \Leftrightarrow C_n^k = \frac{A_n^k}{k!} = \frac{n!}{k!(n-k)!}$

$\Sigma = \{1, 2, \dots n\}$ $\{v \in M^k : v_i < v_j$ при $i < j\}$

$\{1, 2\}$ ← корректная запись т.к. возрастает

$\{2, 1\}$ ← нет

gen(p) p – префикс к.о.

```
gen(p)
    if len(p) == k:
        print(p)
        return
    for c = 1.....n:
        if p ≠ [] and c ≤ p[-1]:
            continue
        if n-c < k-len(p)-1:
            break
        gen(p+[c])
```

можно $c = p[-1]+1 \dots n-k+len(p)+1$ и 2 if не нужен

Правильные скобочные посл-ти

(( )) ( )

баланс $= \#(- \#) \geq 0$

в конце баланс $= 0$

$n=1$ ()

$n=2$ ()()

(())

$n=3$ ((()))

(()())

(())()

()(())

()()()

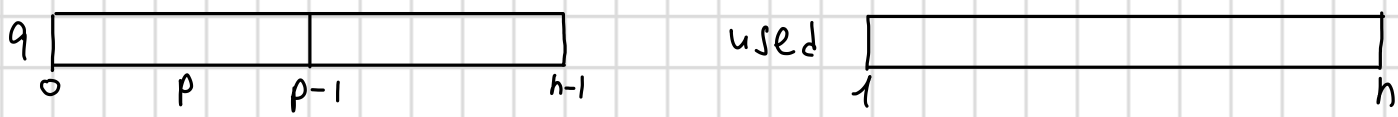
$$\Sigma = \{ (,) \}$$

(<)

```
gen (p, bal)
    if len(p) == 2n
        print(p)
        return
    if bal+1 ≤ 2n - len(p) - 1:
        gen(p + [ ( ], bal + 1)
    if bal ≠ 0:
        gen(p + [ ) ], bal - 1)
```

```
main():
    gen([], 0)
```

Перестановки




```
gen(p)   длина префикса
    if p == n:
        print(a); return
```

```
for i = 1...n
    if !used[i]
        used[i] = true
        a[p] = i
        gen(p+1)
        used[i] = false
```

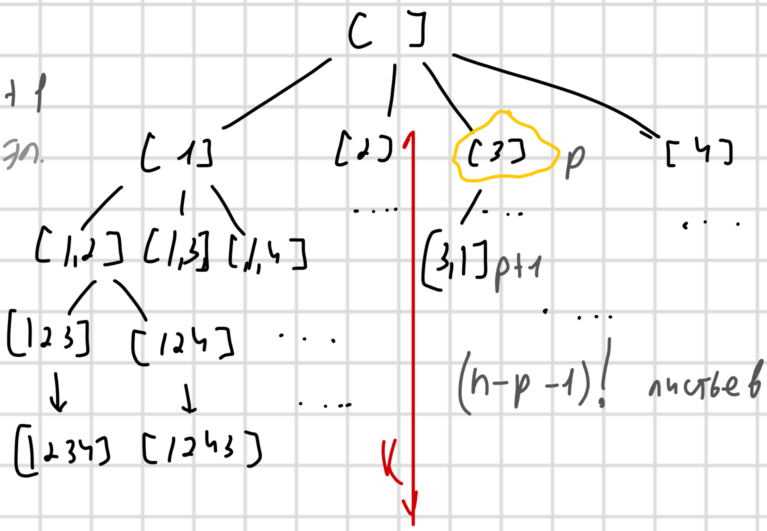
0	1 2 3
1	1 3 2
2	2 1 3
3	2 3 1
4	3 1 2
5	3 2 1

gen(p) $O(n^2) \longrightarrow O(n \log n)$

```
if p == n
    ~~if K == 0:~~   не нужно из-за след. проверок
        print(a)
    K--
    return
for i = 1...n
    if !used[i]
        if K ≥ (n-p-1)! : K -= (n-p-1)!
        else:                    K = K % (n-p-1)!
            used[i] = true
            a[p] = i
            gen(p+1)  ← return
```

$c = \lfloor K / (n-p-1)! \rfloor + 1$

взять c-е не исп. эл.



gen(p)   p – префикс к.о.

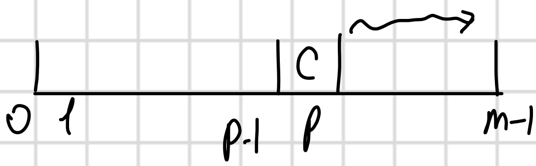
```
gen(p)
    if p - к.о.:
        if k==0: print(p); return
        k--
    for c in Σ (sorted):
        if p+[c] – префикс к.о.:
            t = кол-во к.о. с преф p+[c]
            if k>=t: k-=t
            else:
                gen(p+[c])
                return
```

Сложность: $len \cdot |\Sigma|$

Сочетания

gen(p): $O(nm+nm)$ [обознач $\binom{n}{m} = C_n^m$]

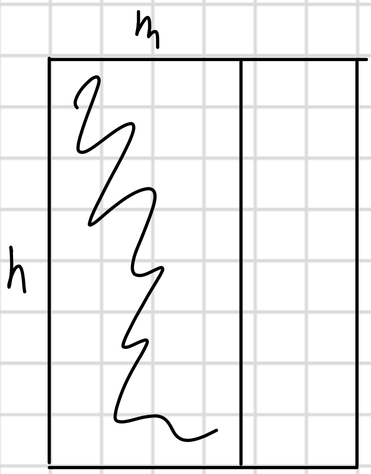
```
gen(p)
    if p == n: print(a); return
    for c = a[p-1]+1 (или 1) ... n-m+p+1
        t = C(n-c, m-p-1)
        if k>=t: k-=t
        else: gen(p+1)
```



$m-p-1$

$c+1, \dots, n$

$n-c$

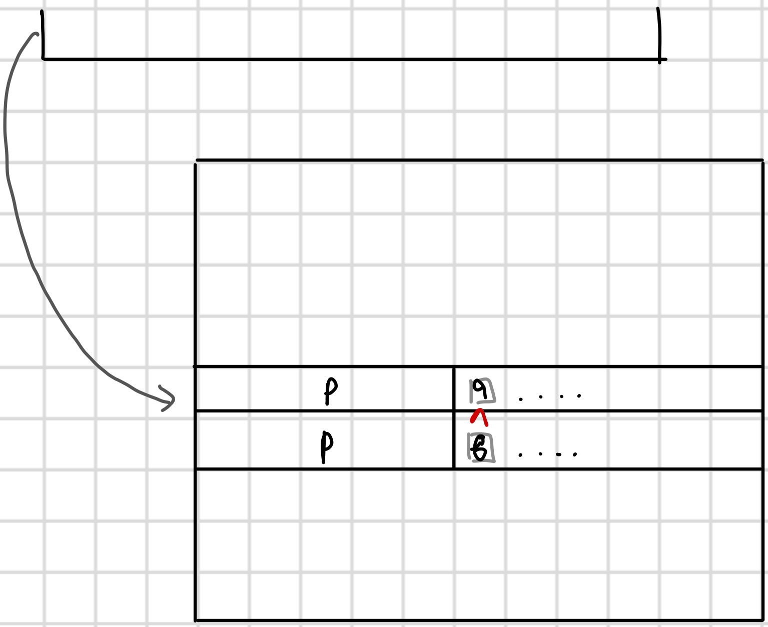


можно сгенерировать все сочетания с пом. треуг. Паск и обратиться

```
gen(p)                          z — к.о.    хотим найти номер z
    if p - к.о.:
        if p == z               плохой код
            res = k
        else: k++
    for c in Σ
        if (p + [c] - преф к.о.):
            gen(p + [c])
```

```
gen(p)   p — префикс к.о.
    if p - к.о:
        if k == z: print(p); return
        k++                                 хороший код
    for c in Σ (sorted):
        if p+[c] — префикс к.о.:
            t = кол-во к.о. с преф p+[c]
            if p+[c] — префикс z:
                gen(p+[c])
                return
            else:
                k += t
```

Следующий К.О.



1) макс префикс, который можно сохр увеличив след.
2) увел-ть след. эл. мин-м образом
превратим a в b
3) приписать p + b
минимальный „хвост"

Пр.: ∇ 010110111011

010110111100

∇ 3 5 7 6 4 2 1

3 6 1 2 4 5 7

Код Грея

- любые 2 соседние строки отл в 1 месте

Факт: $\forall n$ $\exists$ код Грея для $\mathbb{B}$

$\triangleright$ $n=1$

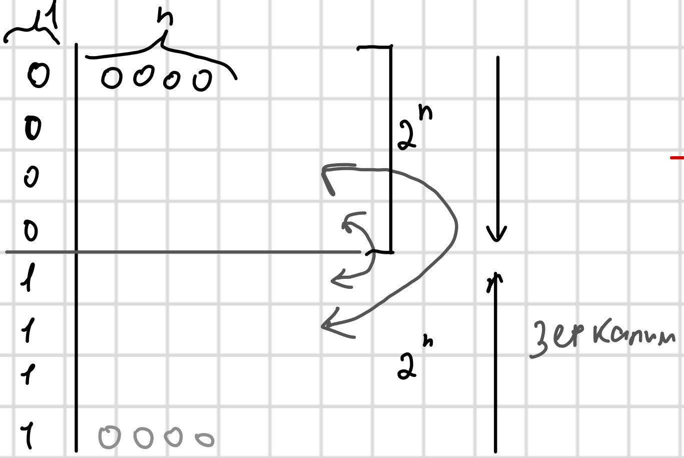
0
1

$n=2$

00
01
11
10

если 1 и посл. К.О. тоже отл в 1 месте — <u>циклический код Грея</u>

Зеркальный код Грея



$n = 3$

0	0	0
0	0	1
0	1	1
0	1	0
1	1	0
1	1	1
1	0	1
1	0	0

0
1
1
0

Если не получается разбиение

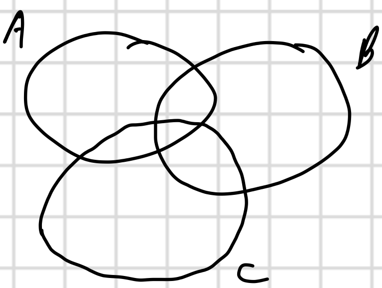
$A = B \sqcup C$ если все хорошо

Иначе

$\exists\ A = B \cup B$

$$|A + B| = |A| + |B| - |A \cap B|$$

$$|A+B+C| = |A| + |B| + |C| - |A \cap B| - |A \cap C| - |B \cap C| + |A \cap B \cap C|$$



Формула включение исключений

$$|A_1 \cup A_2 \cup \ldots \cup A_n| = \sum_{i} |A_i| - \sum_{i<j} A_i A_j + \sum_{i<j<k} |A_i A_j A_k| + $$

$$+ \ldots + |A_1 A_2 \ldots A_n|$$