

1 Генерация

1.1 Все комбинаторные объекты размера n

Рекурсивная процедура генерации "*gen(prefix)*".

```
gen(prefix):  
    if (prefix - "0"..  
        print(prefix)  
    for (c - "0".."1")  
        newp = prefix + [c]  
        if (newp - "0".."1"..  
            gen(newp)
```

Для \mathbb{B}^n

```
gen(prefix)  
    if len(prefix) == n:  
        print(*prefix)  
    else:  
        for c in range(2):  
            newp = prefix+[c]  
            gen(newp)
```

```
gen(p):  
    if p == n:  
        print(*a)  
        return  
    for c=0..1:  
        a[p]=c  
        gen(p+1)
```

1.2 Перестановки размера n

```
gen(p):  
    if p == n:  
        print(a)  
        return  
    for c=1..n  
        a[p]=c  
        if !used[c]:  
            a[p] = c  
            used[c] = true  
            gen(p+1)  
            used[c] = false
```

```
gen(p):  
    if p == k:  
        print(a)  
        return
```

```
for c=1..n
  if (p == 0 or c > a[p - 1]) and (n - c) >= k - (p + 1)
    a[p] = c
    gen(p + 1)
```

1.3 Разбиения на слагаемые

```
gen(p, sum):
  if sum == 0:
    print(a[0:p])
  else:
    for c=min(p==0?n:a[p-1], sum)..1:
      a[p]=c
      gen(p+1, sum-c)
```

1.4 Правильная скобочная последовательность

“” — правильная скобочная последовательность

$A - \text{ПСП} \Rightarrow (A) - \text{ПСП}$

$A, B - \text{ПСП} \Rightarrow AB - \text{ПСП}$

```
gen(p, bal):
  if p == 2*n:
    print(a)
    return
  if 2*n-p-1 >= bal+1:
    a[p] = (
      gen(p + 1, bal + 1)
  if bal > 0
    a[p] = )
    gen(p + 1, bal - 1)
```

2 Нумерация

Номер объекта в нумерации с 0 равен количеству меньших объектов.