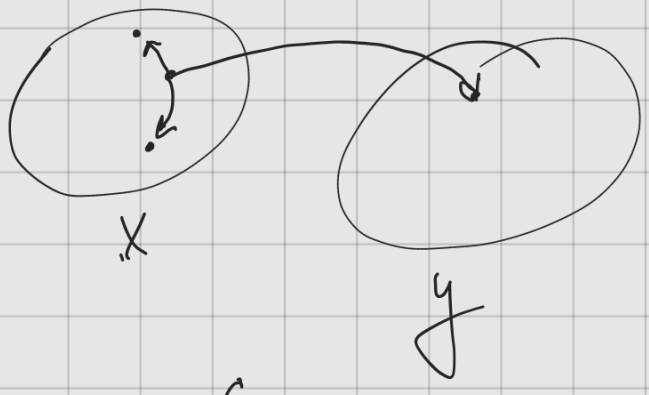
Рекурсия

Р. - способ решения задач через сведение её к подзадачам, которые аналогичны исходной задаче, но проще.

$F_n = F_{n-1} \cdot n \quad / F_{n-1} = \frac{F_n}{n}$



$$F_n = \begin{cases} F_{n-1} \cdot n, & n > 1 \\ 1, & n = 0, 1 \end{cases}$$

$$a^n = \begin{cases} 1, & n = 0 \\ a \cdot a^{n-1}, & n \% 2 == 1 \\ (a^2)^{\frac{n}{2}}, & n \% 2 == 0 \end{cases}$$

```
def pow(a, n):
    if n == 0:
        return 1
    else:
        return
            a * pow(a, n-1)
```

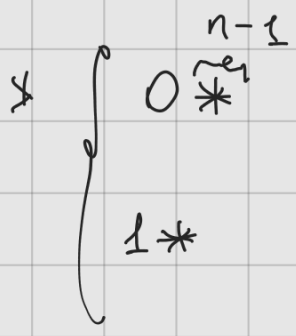
$$a^n = (a^{\frac{n}{2}})^2 = (a^2)^{\frac{n}{2}}$$

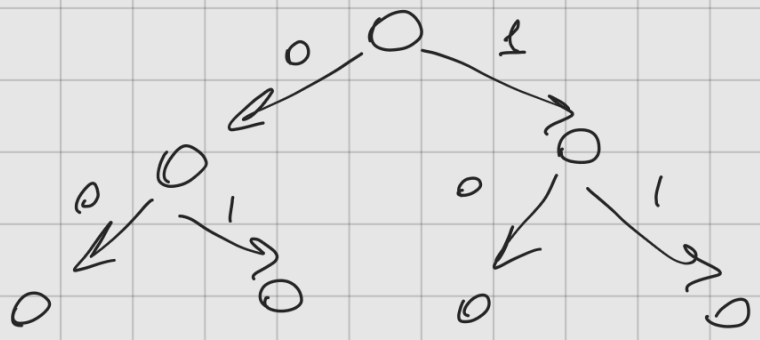
```
def pow(a, n):
    if n == 0: return 1
```

```
elif n%2==1: return a*pow(a, n-1)
else: return pow(a*a, n//2)
```

$O(\log_2 n)$

Генерация комбинаторных объектов.





gen_bin

```
def gen_bin(n, prefix)
    if n==0:
        print(prefix)
    else:
        gen_bin(n-1, prefix + '0')
        gen_bin(n-1, prefix+'1')

gen_bin(3, "число")
```

```
def gen_num(n, base, prefix)
    if n==0:
        print(prefix)
```

```
else:
    for digit in range(base):
        gen_num(n-1, prefix+str(digit)
                , base,
```

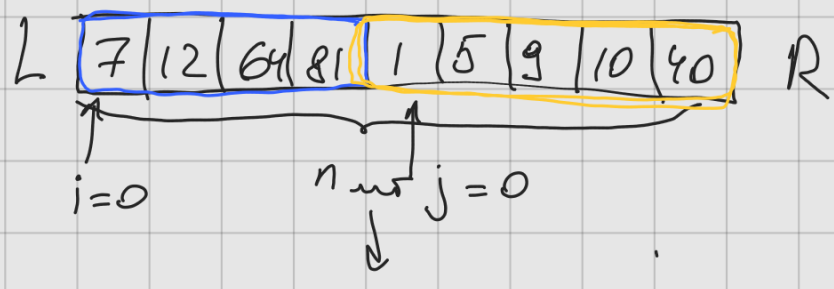
---

permutations

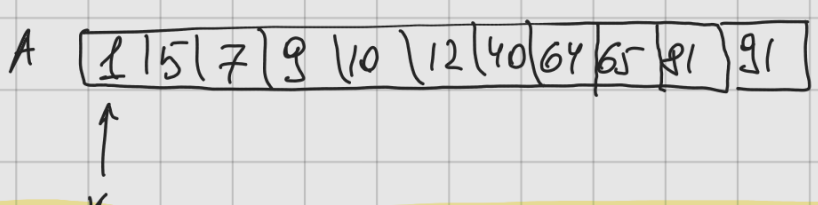
```
def per(prefix, original):
    if len(prefix) == len(original):
        print(prefix)
```

```
else:
    for letter in original:
        per(prefix+letter, original)
if letter not in prefix
```

Рекурентные сортировки



merge



```
while i < len(L) and j < len(R):
    if L[i] <= R[j]:
        A[k] = L[i]
        i += 1
    else:
        A[k] = R[j]
        j += 1
    k += 1
while i < len(L):
    A[k] = L[i]
    i += 1; k += 1
while k < len(R):
```