Vectori - Acces in O(1) -> Random Acces

V[n], n>0

V[i], i<n

Sortare

① Selection Sort.

v = 5, 7, 1, 2, 10, 5, 7, 1

for i = 0 ... n-2 — n

p = find min(v) → n

swap v[i] ↔ v[p]

i = 0 → 1, (7, 5, 2, 10, 5, 7, 1)

i = 1 → 1, 1, (5, 2, 10, 5, 7, 7)

i = 2 → 1, 1, 2, (5, 10, 5, 7, 7)

i = 3 → - - - - -

i = 6 → 1, 1, 2, 5, 5, 7, 7, 10

Selection

(Time) Worst Case ≡ Best ≡ Average Case = $O(n^2)$

(Spatial) = $O(1)$

Swap

```
int a;
int b;
a = a + b;
b = a - b;
a = a - b;
```

Recurență → Definiția unei funcții prin ea însăși

a) $f(x) = \begin{cases} f(x-1)+1, & x \geq 1 \\ 0, & x = 0 \end{cases}$ → $f(x) = x$

$f : \mathbb{R}^+ \to \mathbb{R}$

b) $f(x) = \begin{cases} f(x-1) + f(x-2), & x > 2 \\ 1, & x = 1 \\ 1, & x = 0 \end{cases}$ → FIBONACCI

$f : \mathbb{N} \to \mathbb{N}^*$

e) $h(x) = \begin{cases} x \cdot h(x-1), & x > 0 \\ 1, & x = 0 \end{cases}$ → FACTORIAL

$h : \mathbb{N} \to \mathbb{N}^*$

C

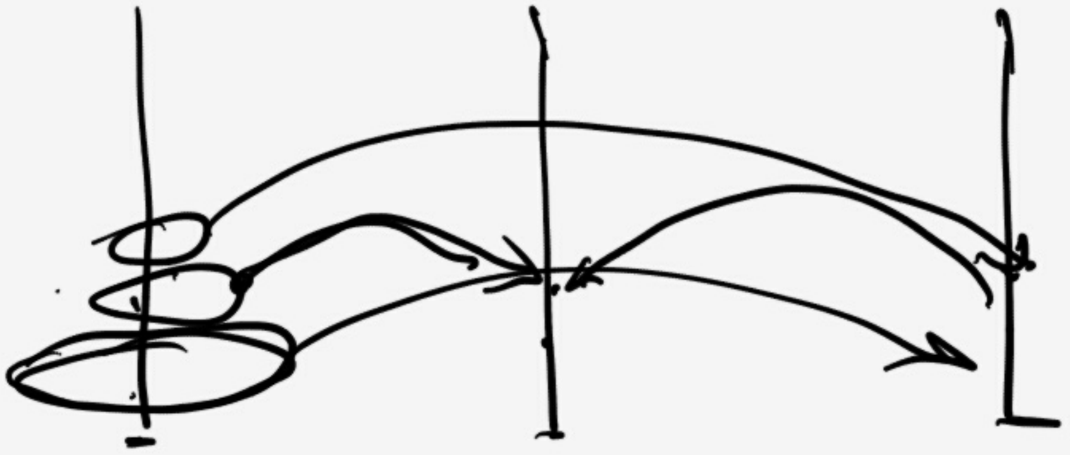
a)

```
float f(float x){
    if(x<0){
        fprintf(stderr, "%f is invalid", x); return -1;
    }
    if(x==0){
        return 0;
    }
    return f(x-1) + 1;
}
```

5) unsigned g(unsigned n){
    if(n==0 || n==1) return 1;
    return g(n-1) + g(n-2);
}

e) unsigned h(unsigned n){
    if(n==0) return 1;
    return n * h(n-1);
}

Hanoi




```
move (start, end, n)
        if (n==1)  start → end;
        else
            tmp ← 3-start-end
            move (start, tmp, n-1)
            move (start, end, 1)
            move (tmp, end, n-1)
```

move(0,2,3)

0→1(2)

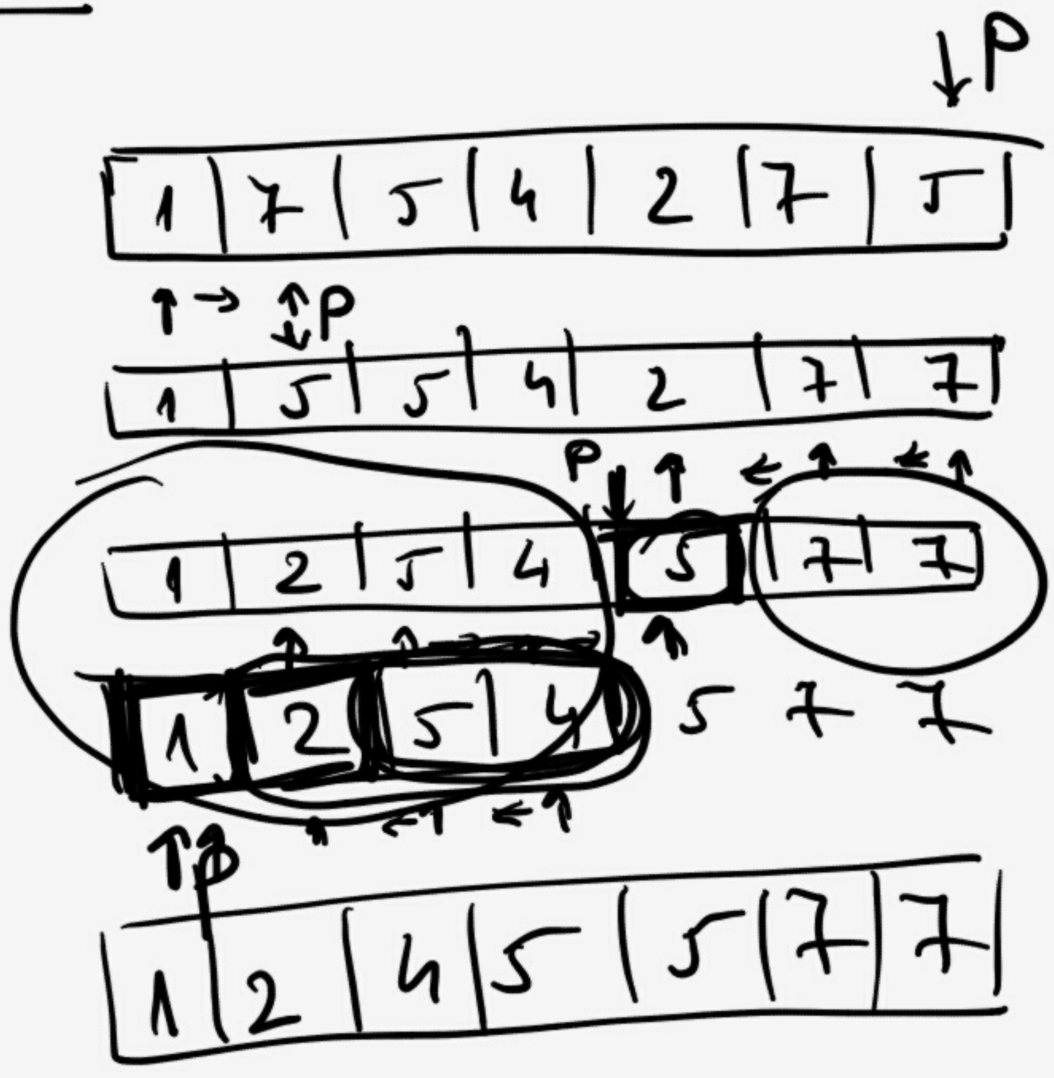
0→2(1)

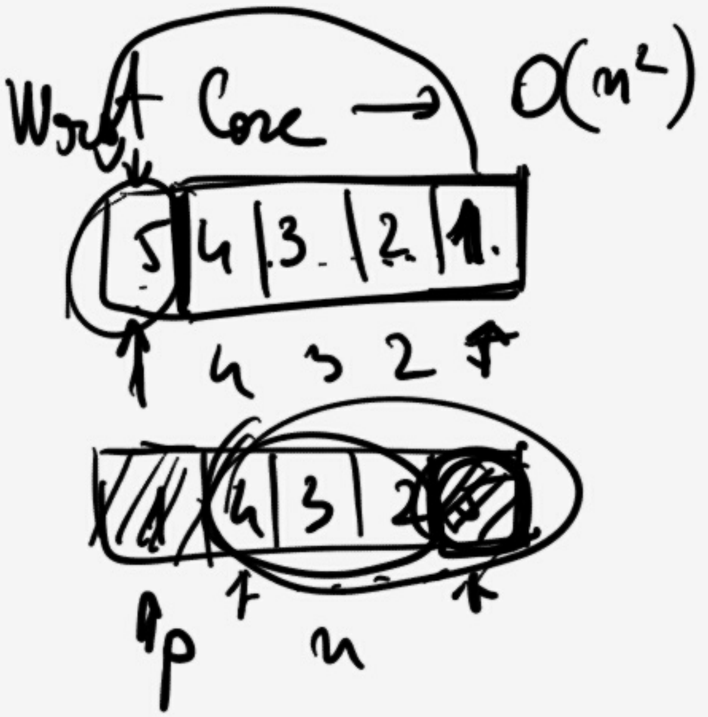
0→1(1)

2→1(1)

0, 2, 1

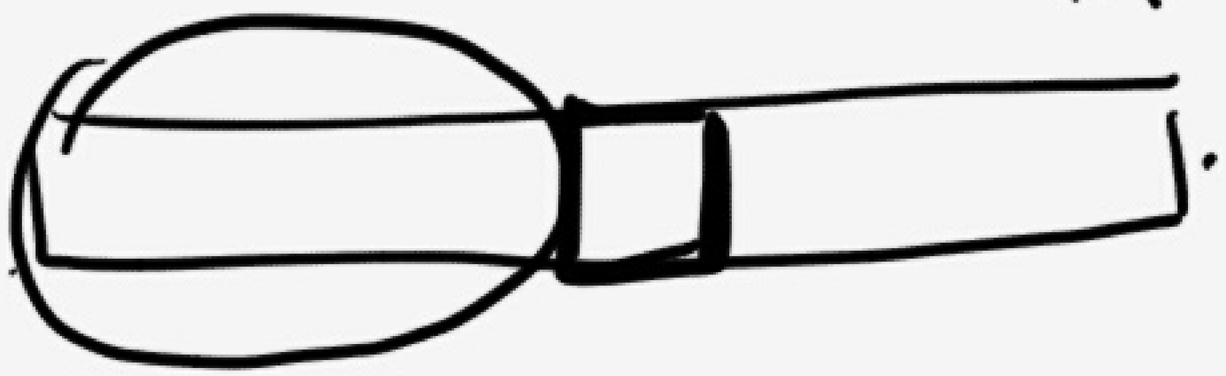
Quick Sort


p
1 7 5 4 2 7 5
p
1 5 5 4 2 7 7
p
1 2 5 4 5 7 7
1 2 5 4
5 7 7
1 2 4 5 5 7 7


Worst Case → O(n²)
5 4 3 2 1
4 3 2
1 4 3 2
p
n
n-1
1 2 3 4 5 6
(n-1)+
(n-2)+
n-> - n²

$n \rightarrow$ n pași pivot

$\frac{n}{2}$



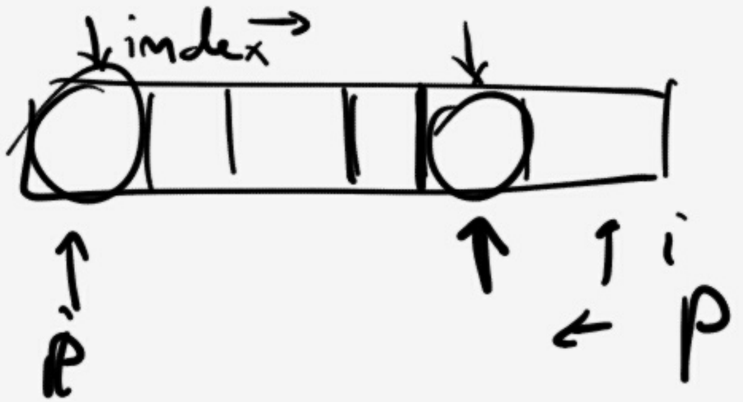
$$n + 2\left(\frac{n}{2} + 2\left(\frac{n}{4} + \ldots\right.\right.$$

$$O(n \cdot \log_2 n) \rightarrow \text{best Case}$$

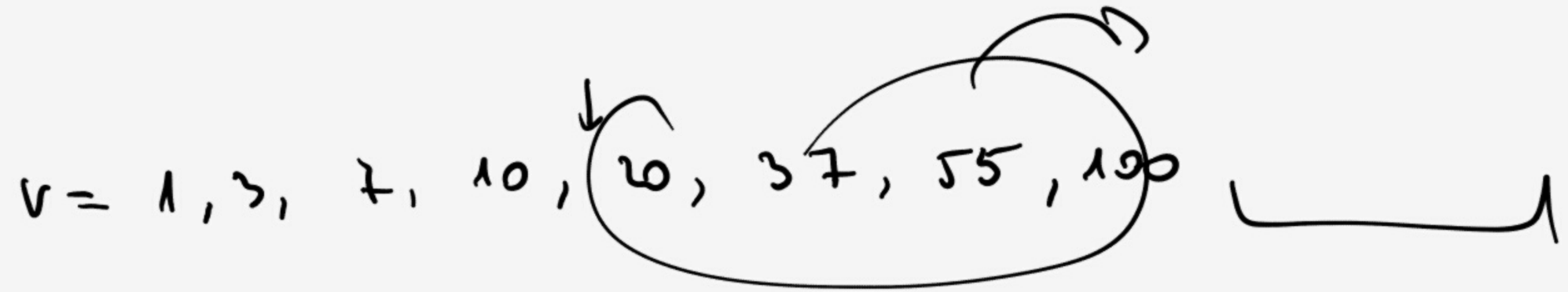
$$4\left(\frac{n}{8} + \ldots 2\left(\frac{n}{16}\right)\right)$$

---

qSort(v, 0, n-1)




```
qsort(v, start, end)
    if(start == end) return;
    pivot ← start
    direction ← -1
    index ← end
    while pivot != index
        if((pivot > index) != (v[pivot] > v[index]))
            swap(v[pivot], v[index])
            swap(pivot, index)
            direction = - direction
        index += direction
    qsort(v, start, pivot)
    qsort(v, pivot, end)
```



→ 17 → (caută binară în v) → k

$O(\log n + n) \rightarrow O(n)$