Structuri de date și algoritmi

Tehnica Divide Et Impera

Lect. Dr. Ing. Cristian Chilipirea – cristian.chilipirea@mta.ro

Divide et Impera

1. Descompunerea problemei în subprobleme.
2. Rezolvarea independentă a fiecărei subprobleme.
3. Recompunerea rezultatelor pentru a afla soluția problemei inițiale.

Acești pași pot fi aplicați recursiv, până ce subproblemele sunt suficient de mici pentru a fi banale.

Metoda Master – calcul complexități soluții recursive

Problema de dimensiune n, divizată în $\boldsymbol{a}$ subprobleme de dimensiune $n/\boldsymbol{b}$. Combinarea subproblemelor şi divizarea problemei se realizează într-un timp $f(n)$. $\boldsymbol{a} \geq 1; \boldsymbol{b} > 1$

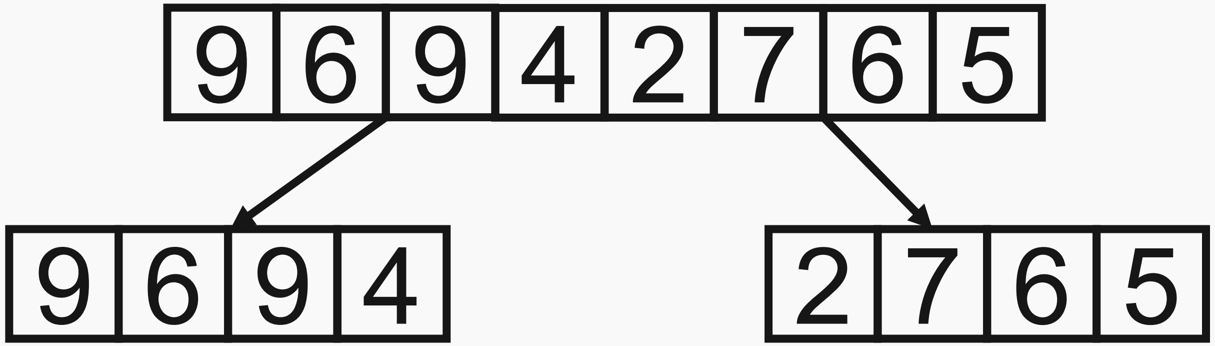
$$\boldsymbol{T(n) = aT\left(\frac{n}{b}\right) + f(n)}$$

1. $f(n) = O\left(n^{\log_b a-\epsilon}\right), \epsilon > 0 \Rightarrow \boldsymbol{T(n) = \Theta\left(n^{\log_b a}\right)}$
2. $f(n) = \Theta\left(n^{\log_b a}\right) \Rightarrow \boldsymbol{T(n) = \Theta\left(n^{\log_b a} lg n\right)}$
3. $f(n) = \Omega\left(n^{\log_b a+\epsilon}\right), \epsilon > 0,\ af\left(\frac{n}{b}\right) \leq cf(n), c < 1 \ \Rightarrow \boldsymbol{T(n) = \Theta(f(n))}$

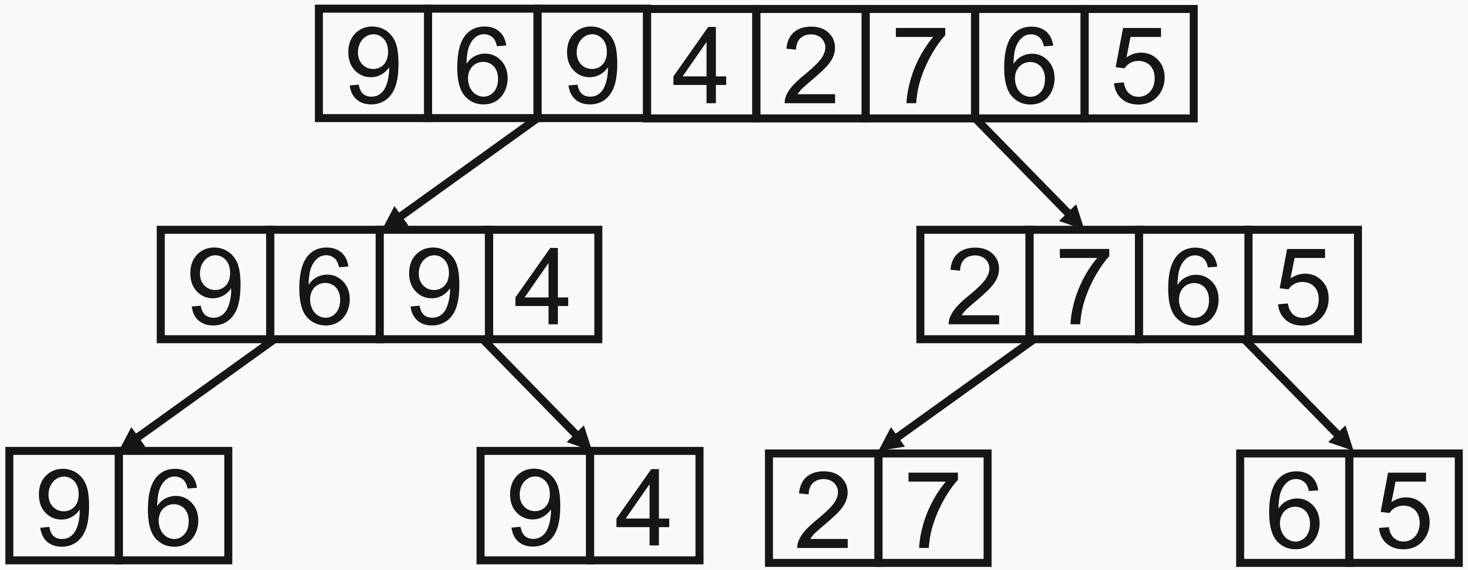
Problemă – min/max un vector

9	6	9	4	2	7	6	5

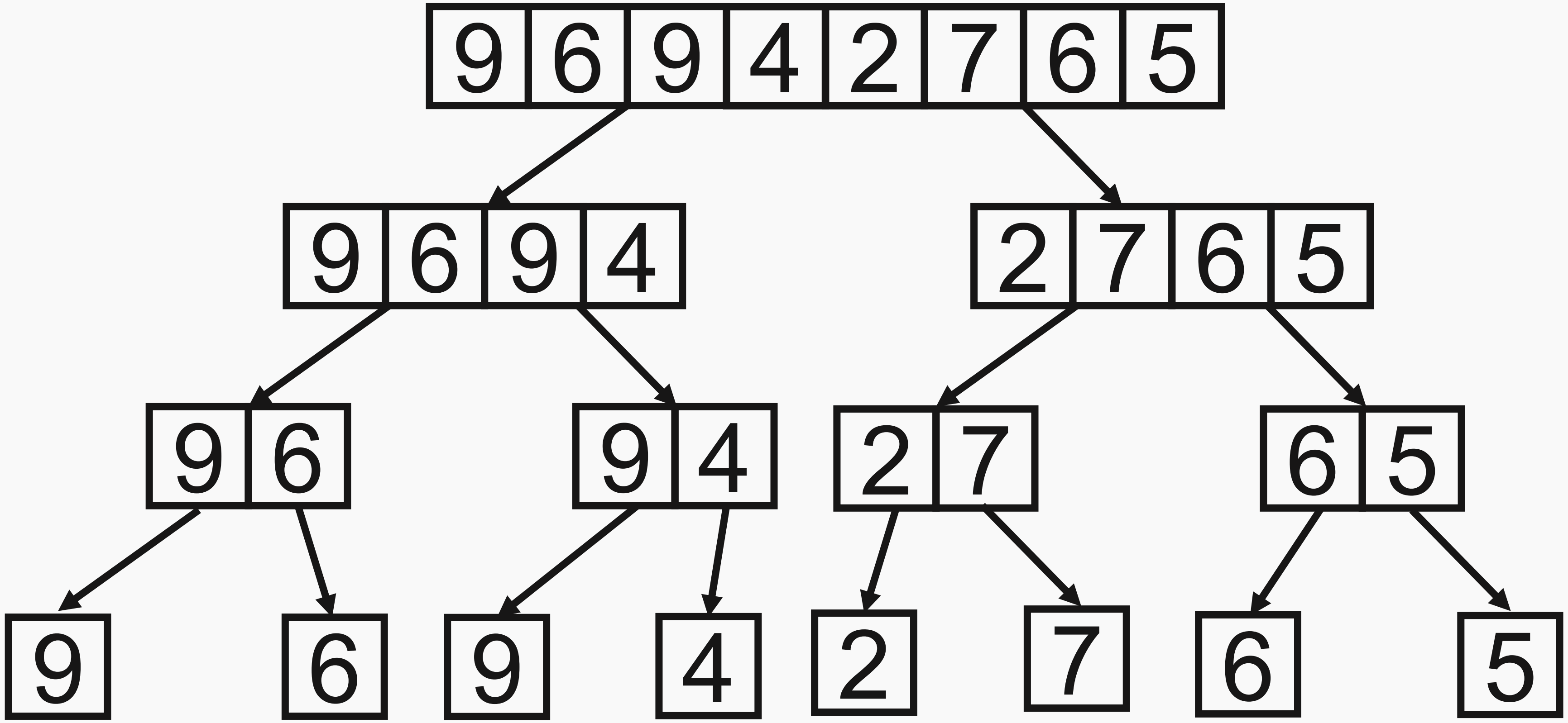
Problemă – min/max un vector



Problemă – min/max un vector



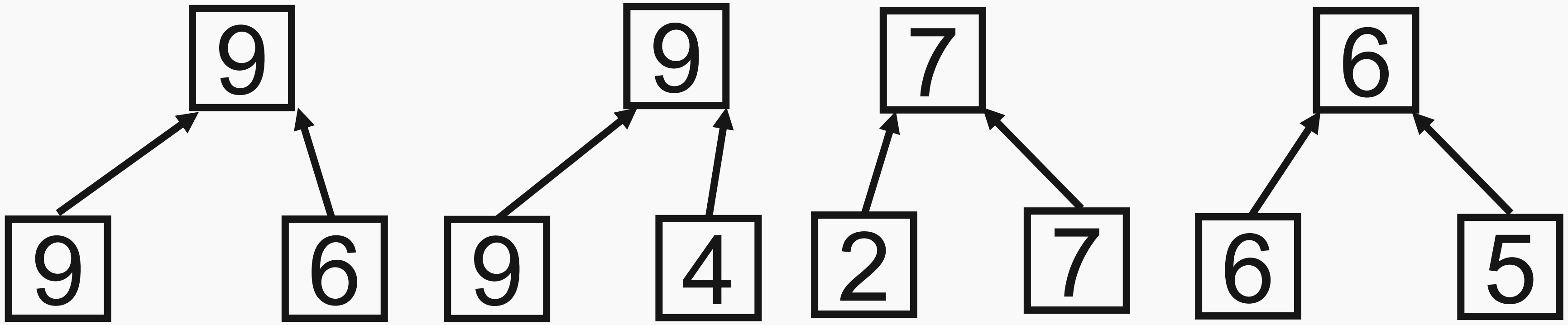
Problemă – min/max un vector



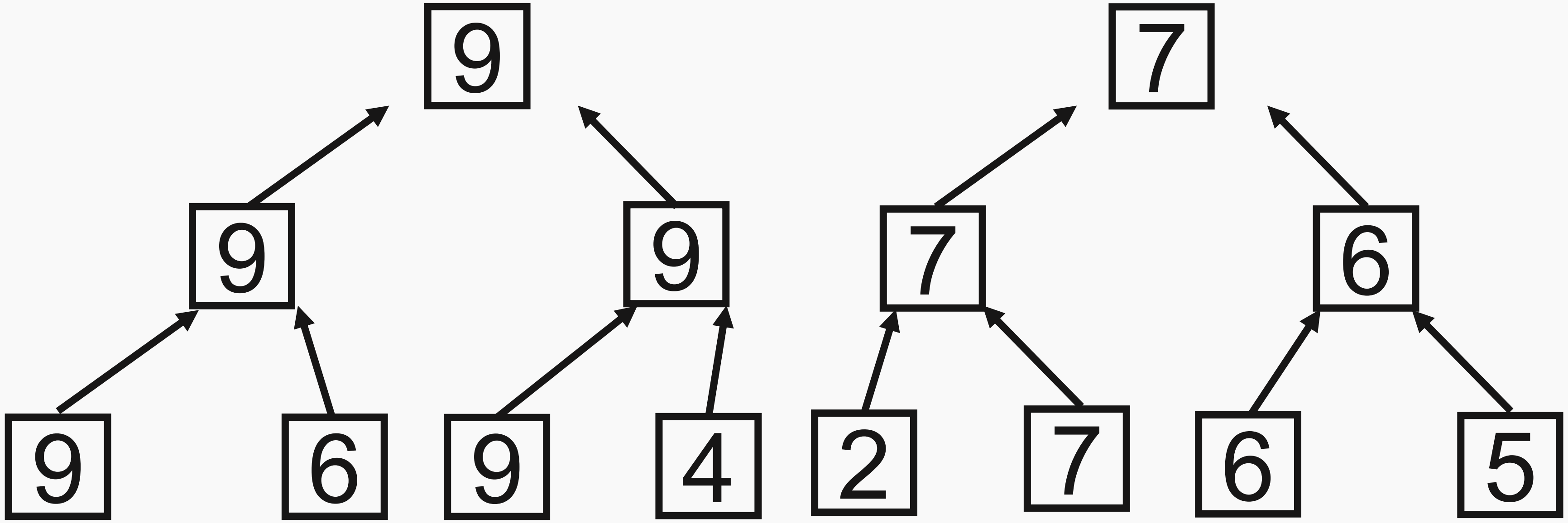
Problemă – min/max un vector

9 6 9 4 2 7 6 5

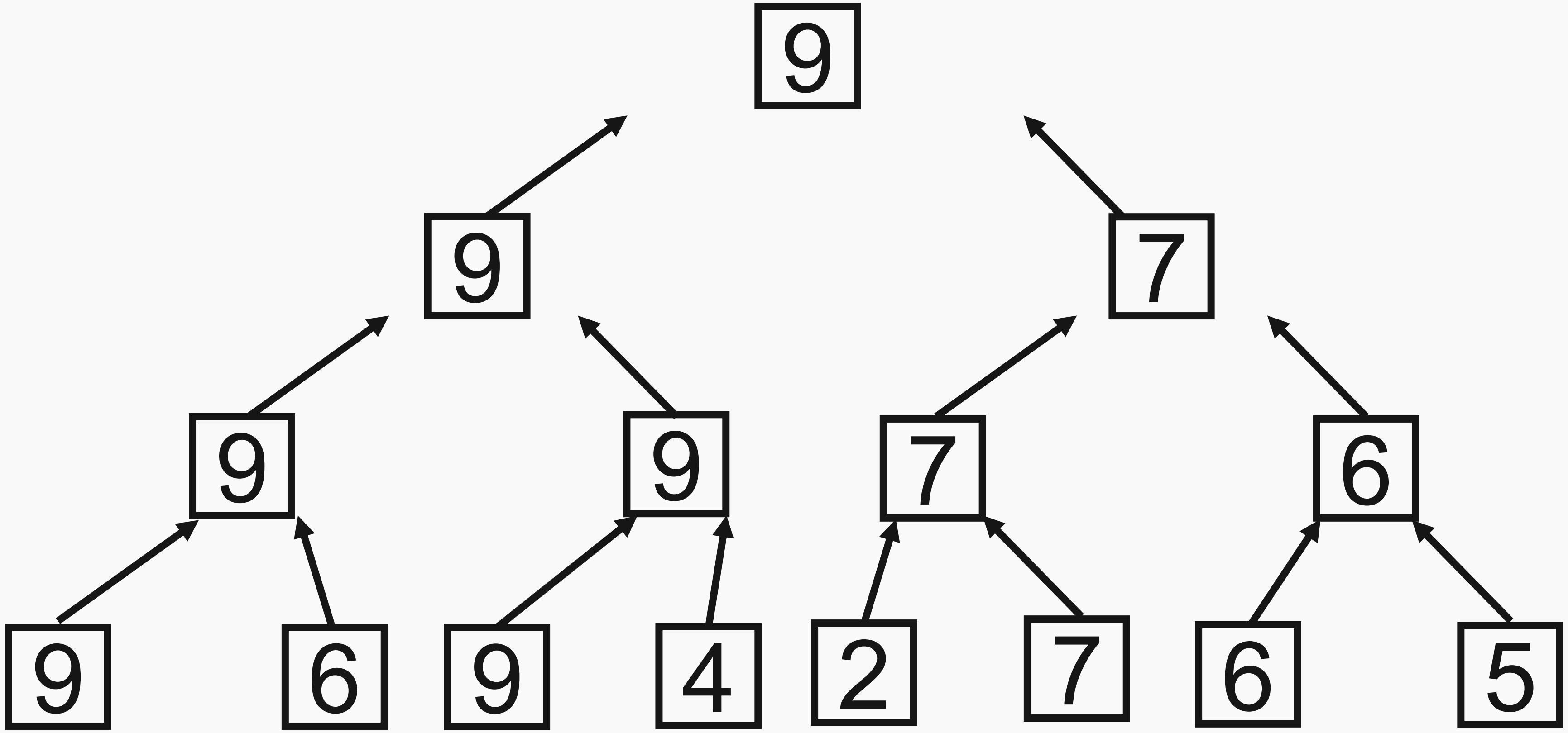
Problemă – min/max un vector



Problemă – min/max un vector



Problemă – min/max un vector



Problemă – min/max un vector - complexitate

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

Problemă – min/max un vector - complexitate

$$T(n) = 2T\left(\frac{n}{2}\right) + f(n)$$

Problemă – min/max un vector - complexitate

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

$$a = 2 \qquad b = 2 \qquad f(n) = c$$

1. $f(n) = O\left(n^{\log_b a-\epsilon}\right), \epsilon > 0 \Rightarrow T(n) = \Theta\left(n^{\log_b a}\right)$
2. $f(n) = \Theta\left(n^{\log_b a}\right) \Rightarrow T(n) = \Theta\left(n^{\log_b a} \lg n\right)$
3. $f(n) = \Omega\left(n^{\log_b a+\epsilon}\right), \epsilon > 0,\ af\left(\frac{n}{b}\right) \leq cf(n), c < 1 \Rightarrow T(n) = \Theta(f(n))$

Problemă – min/max un vector - complexitate

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

$$a = 2 \qquad b = 2 \qquad f(n) = 1$$

1. $c = O\left(n^{log_2 2-\epsilon}\right), \epsilon > 0 \Rightarrow T(n) = \Theta\left(n^{log_b a}\right)$
2. $c = \Theta\left(n^{log_2 2}\right) \Rightarrow T(n) = \Theta\left(n^{log_b a} lgn\right)$
3. $c = \Omega\left(n^{log_2 2+\epsilon}\right), \epsilon > 0,\ af\left(\frac{n}{b}\right) \leq cf(n), c < 1 \ \Rightarrow T(n) = \Theta(f(n))$

Problemă – min/max un vector - complexitate

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

$$a = 2 \quad b = 2 \quad f(n) = 1$$

1. $c = O(n^{1-\epsilon}), \epsilon > 0 \Rightarrow T(n) = \Theta\left(n^{\log_b a}\right)$
2. $c = \Theta(n^1) \Rightarrow T(n) = \Theta\left(n^{\log_b a} lgn\right)$
3. $c = \Omega(n^{1+\epsilon}), \epsilon > 0,\ af\left(\frac{n}{b}\right) \leq cf(n), c < 1 \Rightarrow T(n) = \Theta(f(n))$

Problemă – min/max un vector - complexitate

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

$$a = 2 \qquad b = 2 \qquad f(n) = 1$$

$$c = O(n^{1-\epsilon}), \epsilon > 0 \Rightarrow T(n) = \Theta\left(n^{\log_2 2}\right)$$

Problemă – min/max un vector - complexitate

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

$$a = 2 \qquad b = 2 \qquad f(n) = 1$$

$$c = O(n^{1-\epsilon}), \epsilon > 0 \Rightarrow T(n) = \Theta(n)$$

$$T(n) = \Theta(n)$$

Problemă – căutare binară

De căutat: $\boxed{3}$

1	2	3	5	6	7	8	9

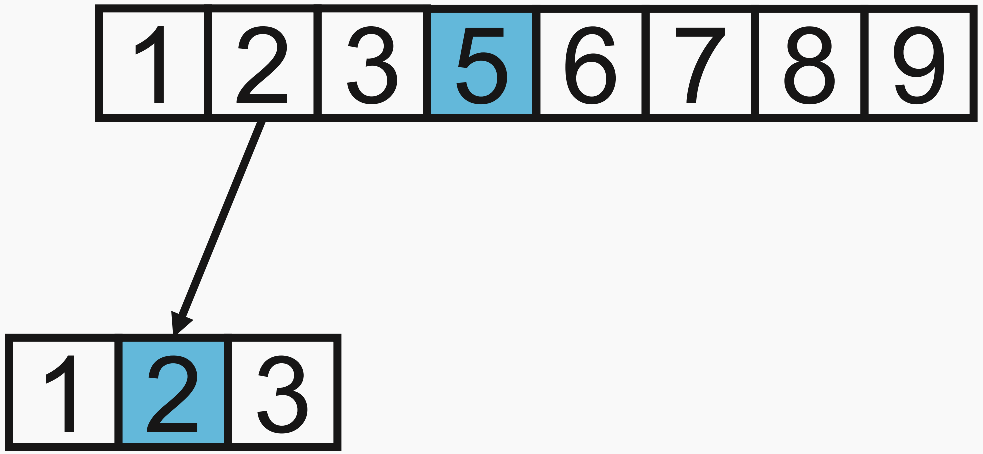
Problemă – căutare binară

De căutat: $\boxed{3}$

1	2	3	5	6	7	8	9

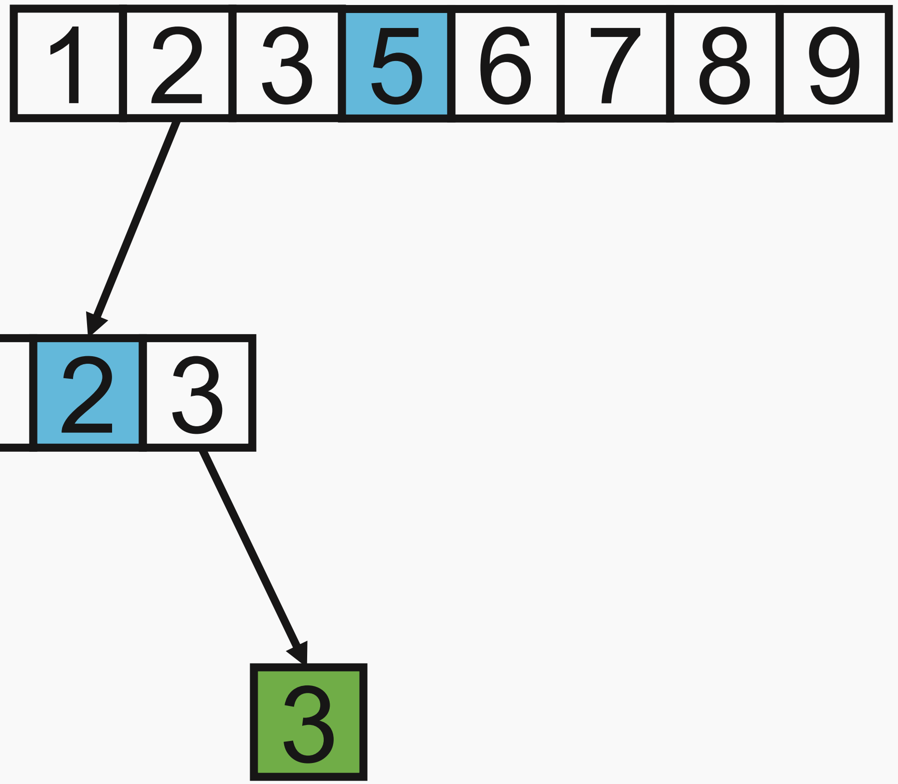
Problemă – căutare binară

De căutat: 3



Problemă – căutare binară

De căutat: 3



Problemă – căutare binară - Complexitate

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

Problemă – căutare binară - Complexitate

$$T(n) = 1T\left(\frac{n}{2}\right) + c$$

Problemă – căutare binară - Complexitate

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

$$a = 1 \quad b = 2 \quad f(n) = c$$

1. $f(n) = O\left(n^{\log_b a - \epsilon}\right), \epsilon > 0 \Rightarrow T(n) = \Theta\left(n^{\log_b a}\right)$
2. $f(n) = \Theta\left(n^{\log_b a}\right) \Rightarrow T(n) = \Theta\left(n^{\log_b a} \lg n\right)$
3. $f(n) = \Omega\left(n^{\log_b a + \epsilon}\right), \epsilon > 0,\ af\left(\frac{n}{b}\right) \le cf(n), c < 1 \Rightarrow T(n) = \Theta(f(n))$

Problemă – căutare binară - Complexitate

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

$$a = 1 \quad b = 2 \quad f(n) = c$$

$$c = \Theta\left(n^{log_2 1}\right) \Rightarrow T(n) = \Theta\left(n^{log_b a} lgn\right)$$

Problemă – căutare binară - Complexitate

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

$$a = 1 \quad b = 2 \quad f(n) = c$$

$$c = \Theta(n^0) \Rightarrow T(n) = \Theta\left(n^{\log_b a} \lg n\right)$$

Problemă – căutare binară - Complexitate

$$\boldsymbol{T(n) = aT\left(\frac{n}{b}\right) + f(n)}$$

$$\boldsymbol{a = 1 \quad b = 2 \quad f(n) = c}$$

$$c = \Theta(n^0) \Rightarrow \boldsymbol{T(n) = \Theta\left(n^0 lgn\right)}$$

Problemă – căutare binară - Complexitate

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

$$a = 1 \qquad b = 2 \qquad f(n) = c$$

$$c = \Theta(n^0) \Rightarrow T(n) = \Theta(lgn)$$

$$T(n) = \Theta(lgn)$$

Problemă – Numărul de inversiuni

Un șir de numere întregi. O inversiune este o pereche de indecşi $i < j$ astfel încât $S[i] > S[j]$. Să se detemine câte inversiuni sunt într-un şir de numere dat.

Exemplu: în şirul {0 1 9 4 5 7 6 8 2} sunt 12 inversiuni.

1. Divizarea vectorului în două jumătăţi.
2. Calculul numărului de inversiuni pe fiecare jumătate.
3. Calculul numărului de inversiuni cu un index în prima jumătate şi celălalt index în a doua jumătate.

Problemă – Numărul de inversiuni - Complexitate

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

Problemă – Numărul de inversiuni - Complexitate

$$T(n) = 2T\left(\frac{n}{2}\right) + \Theta\left(\left(\frac{n}{2}\right)^2\right)$$

Problemă – Numărul de inversiuni - Complexitate

$$T(n) = 2T\left(\frac{n}{2}\right) + \Theta\left(n^2\right)$$

Problemă – Numărul de inversiuni - Complexitate

$$T(n) = 2T\left(\frac{n}{2}\right) + \Theta\left(n^2\right)$$

1. $f(n) = O\left(n^{log_b a - \epsilon}\right), \epsilon > 0 \Rightarrow T(n) = \Theta\left(n^{log_b a}\right)$
2. $f(n) = \Theta\left(n^{log_b a}\right) \Rightarrow T(n) = \Theta\left(n^{log_b a} lgn\right)$
3. $f(n) = \Omega\left(n^{log_b a + \epsilon}\right), \epsilon > 0,\ af\left(\frac{n}{b}\right) \leq cf(n), c < 1 \Rightarrow T(n) = \Theta(f(n))$

Problemă – Numărul de inversiuni - Complexitate

$$\boldsymbol{T(n) = 2T\left(\frac{n}{2}\right) + \Theta\left(n^2\right)}$$

$$f(n) = \Omega\left(n^{\log_b a+\epsilon}\right), \epsilon > 0, af\left(\frac{n}{b}\right) \le cf(n), c < 1 \;\Rightarrow \boldsymbol{T(n) = \Theta(f(n))}$$

Problemă – Numărul de inversiuni - Complexitate

$$\boldsymbol{T(n) = 2T\left(\frac{n}{2}\right) + \Theta\left(n^2\right)}$$

$$f(n) = \Omega\left(n^{\log_b a+\epsilon}\right), \epsilon > 0, af\left(\frac{n}{b}\right) \le cf(n), c < 1 \Rightarrow \boldsymbol{T(n) = \Theta(f(n))}$$

$$\boldsymbol{T(n) = \Theta\left(n^2\right)}$$

Problemă – Numărul de inversiuni - MergeSort

```
int mergesort(int *v, int l, int r)
{
    if (l == r)
        return 0;
    int m = (l + r) / 2;
    int nbinv1 = mergesort(v, l, m);     // sortează prima jumătate
    int nbinv2 = mergesort(v, m + 1, r); // a doua jumătate
    int nbinv;
    merge(v, l, m, r, nbinv); // interclasează datele sortate
    return nbinv1 + nbinv2 + nbinv;
}
```

Problemă – Numărul de inversiuni – MergeSort - Complexitate

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

Problemă – Numărul de inversiuni – MergeSort - Complexitate

$$T(n) = 2T\left(\frac{n}{2}\right) + \Theta(n)$$

Problemă – Numărul de inversiuni – MergeSort - Complexitate

$$\boldsymbol{T(n) = 2T\left(\frac{n}{2}\right) + \Theta(n)}$$

1. $f(n) = O\left(n^{log_b a-\epsilon}\right), \epsilon > 0 \Rightarrow \boldsymbol{T(n) = \Theta\left(n^{log_b a}\right)}$
2. $f(n) = \Theta\left(n^{log_b a}\right) \Rightarrow \boldsymbol{T(n) = \Theta\left(n^{log_b a} lgn\right)}$
3. $f(n) = \Omega\left(n^{log_b a+\epsilon}\right), \epsilon > 0,\ af\left(\frac{n}{b}\right) \leq cf(n), c < 1 \Rightarrow \boldsymbol{T(n) = \Theta(f(n))}$

Problemă – Numărul de inversiuni – MergeSort - Complexitate

$$\boldsymbol{T(n) = 2T\left(\frac{n}{2}\right) + \Theta(n)}$$

$$f(n) = \Theta\left(n^{\log_b a}\right) \Rightarrow \boldsymbol{T(n) = \Theta\left(n^{\log_b a} \lg n\right)}$$

Problemă – Numărul de inversiuni – MergeSort - Complexitate

$$T(n) = 2T\left(\frac{n}{2}\right) + \Theta(n)$$

$$\Theta(n) = \Theta\left(n^{\log_2 2}\right) \Rightarrow T(n) = \Theta\left(n^{\log_2 2} \lg n\right)$$

Problemă – Numărul de inversiuni – MergeSort - Complexitate

$$T(n) = 2T\left(\frac{n}{2}\right) + \Theta(n)$$

$$\Theta(n) = \Theta\left(n^{\log_2 2}\right) \Rightarrow T(n) = \Theta\left(n^{\log_2 2} lgn\right)$$

$$T(n) = \Theta(nlgn)$$

Problemă – Cele mai apropiate două puncte

N puncte în plan, situate pe o dreaptă. Să se determine perechea de puncte vecine care sunt cel mai apropiate.

Date de intrare: vector de coordonate sortat crescător.

Rezolvarea prin metoda Divide & Impera:

- se împarte mulţimea de puncte în două jumătăţi;
- se calculează cele 2 segmente minime din jumătăţi;
- se calculează segmentul “dintre” cele două jumătăţi;

Segmentul minim va fi minimul dintre cele 3 segmente!

Problemă – Cele mai apropiate două puncte - Complexitate

$$\boldsymbol{T(n) = 2T\left(\frac{n}{2}\right) + c}$$

1. $f(n) = O\left(n^{\log_b a - \epsilon}\right), \epsilon > 0 \Rightarrow \boldsymbol{T(n) = \Theta\left(n^{\log_b a}\right)}$
2. $f(n) = \Theta\left(n^{\log_b a}\right) \Rightarrow \boldsymbol{T(n) = \Theta\left(n^{\log_b a} \lg n\right)}$
3. $f(n) = \Omega\left(n^{\log_b a + \epsilon}\right), \epsilon > 0,\ af\left(\frac{n}{b}\right) \leq cf(n), c < 1 \Rightarrow \boldsymbol{T(n) = \Theta(f(n))}$

Problemă – Cele mai apropiate două puncte - Complexitate

$$T(n) = 2T\left(\frac{n}{2}\right) + c$$

$$f(n) = O\left(n^{\log_b a - \epsilon}\right), \epsilon > 0 \Rightarrow T(n) = \Theta\left(n^{\log_b a}\right)$$

Problemă – Cele mai apropiate două puncte - Complexitate

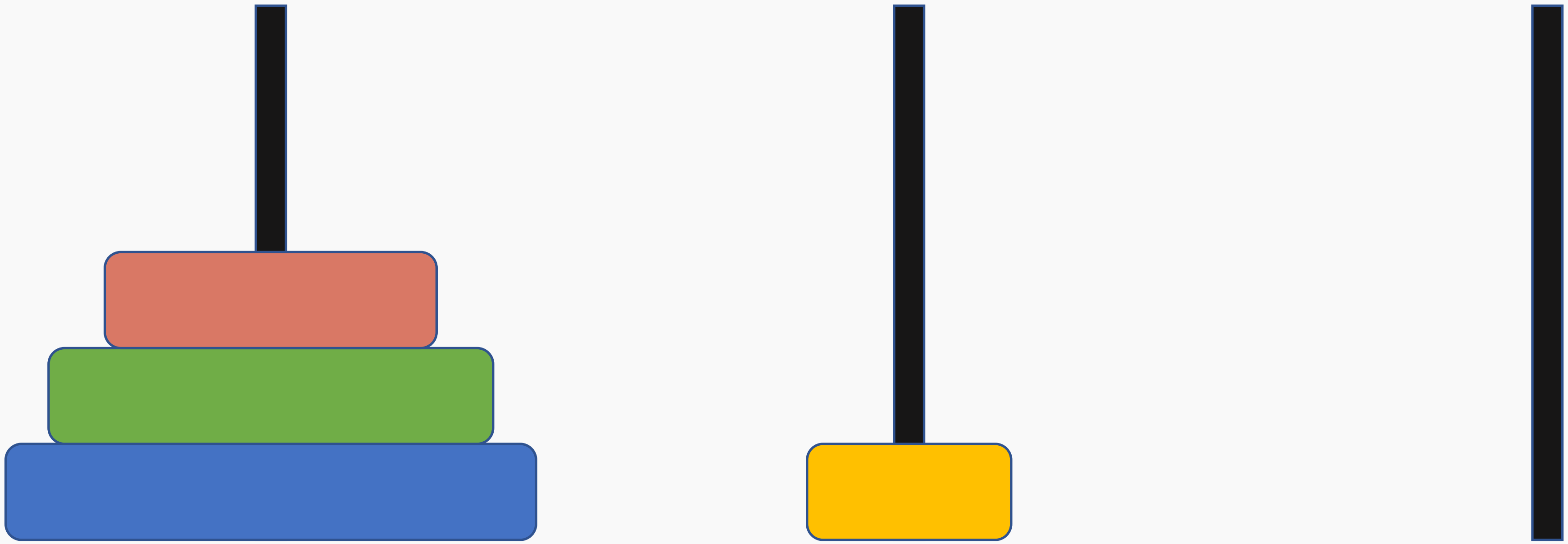
$$T(n) = 2T\left(\frac{n}{2}\right) + c$$

$$f(n) = O\left(n^{\log_b a - \epsilon}\right), \epsilon > 0 \Rightarrow T(n) = \Theta\left(n^{\log_b a}\right)$$

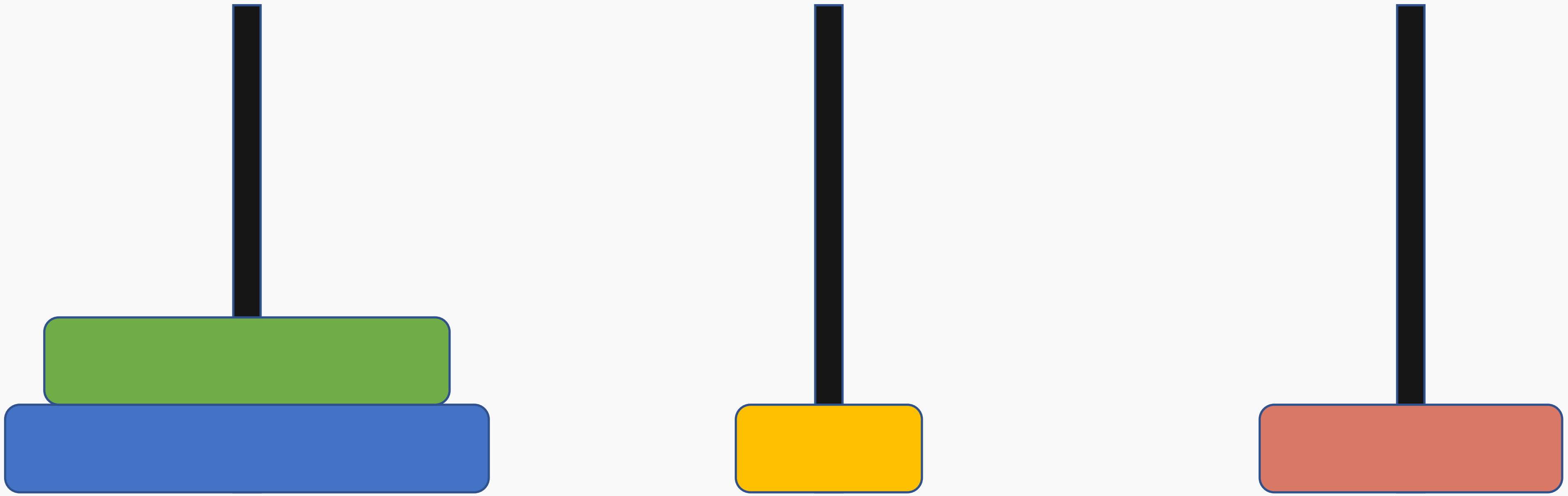
$$T(n) = \Theta(n)$$

Problemă – Turnurile din Hanoi

Problemă – Turnurile din Hanoi

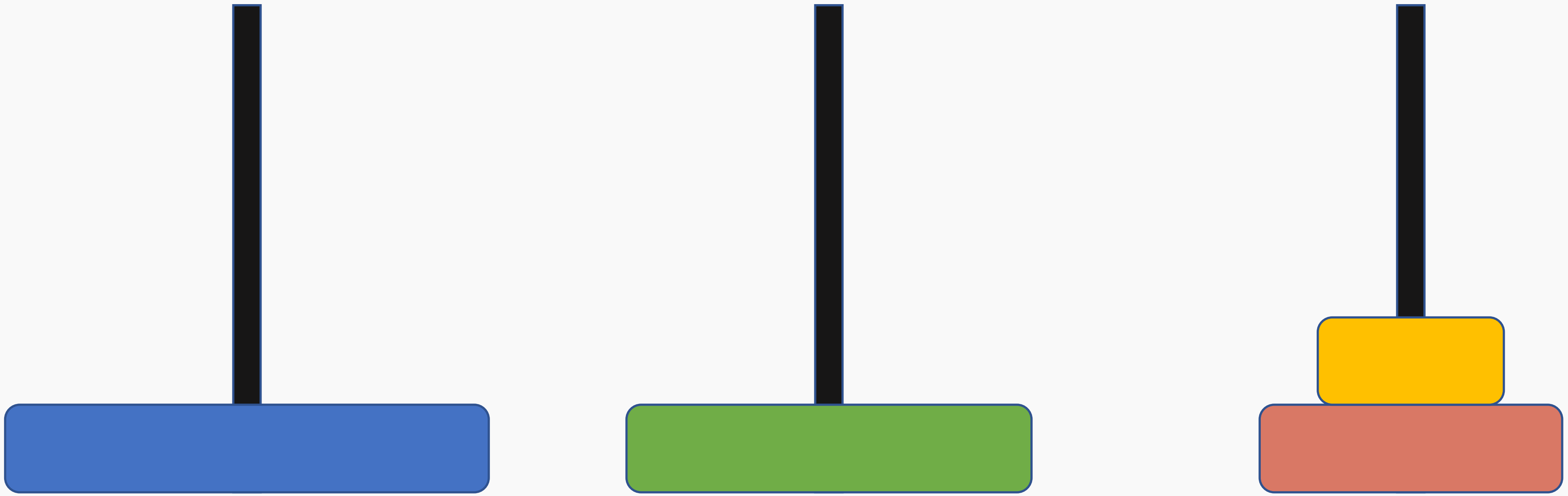


Problemă – Turnurile din Hanoi

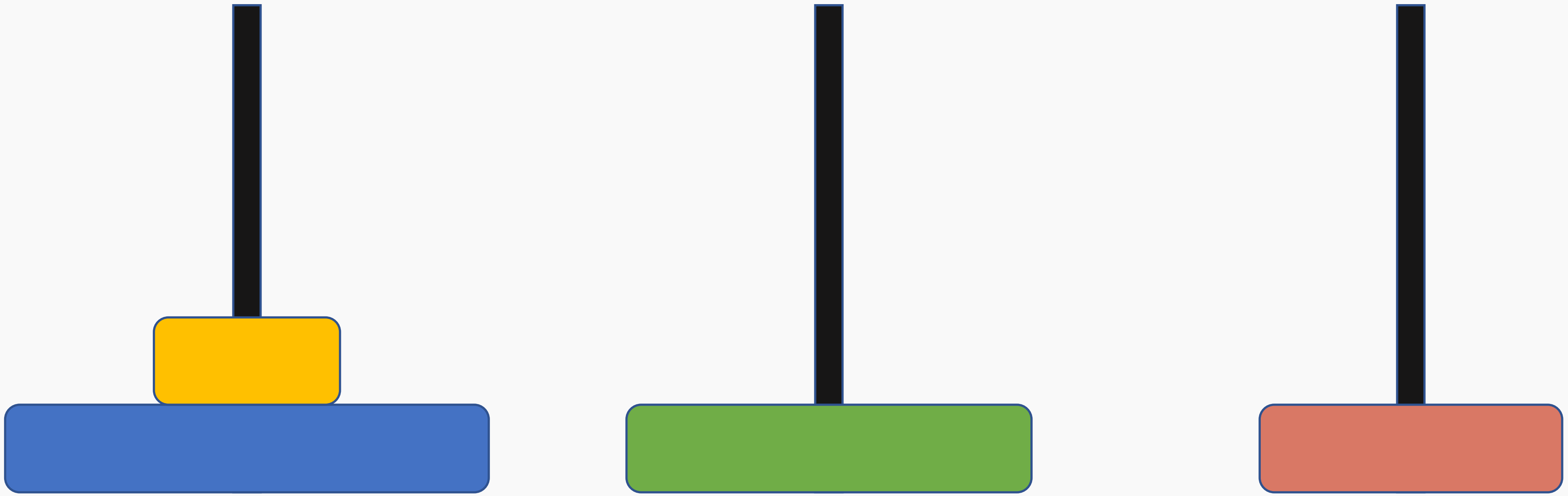


Problemă – Turnurile din Hanoi

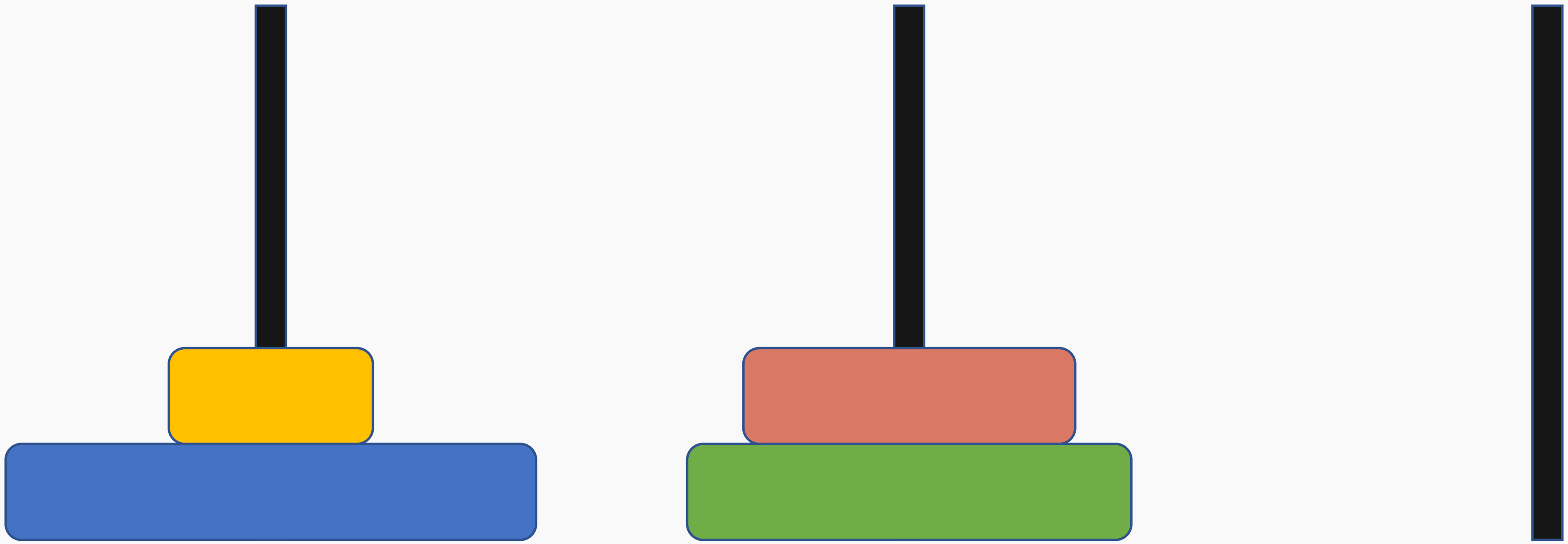
Problemă – Turnurile din Hanoi



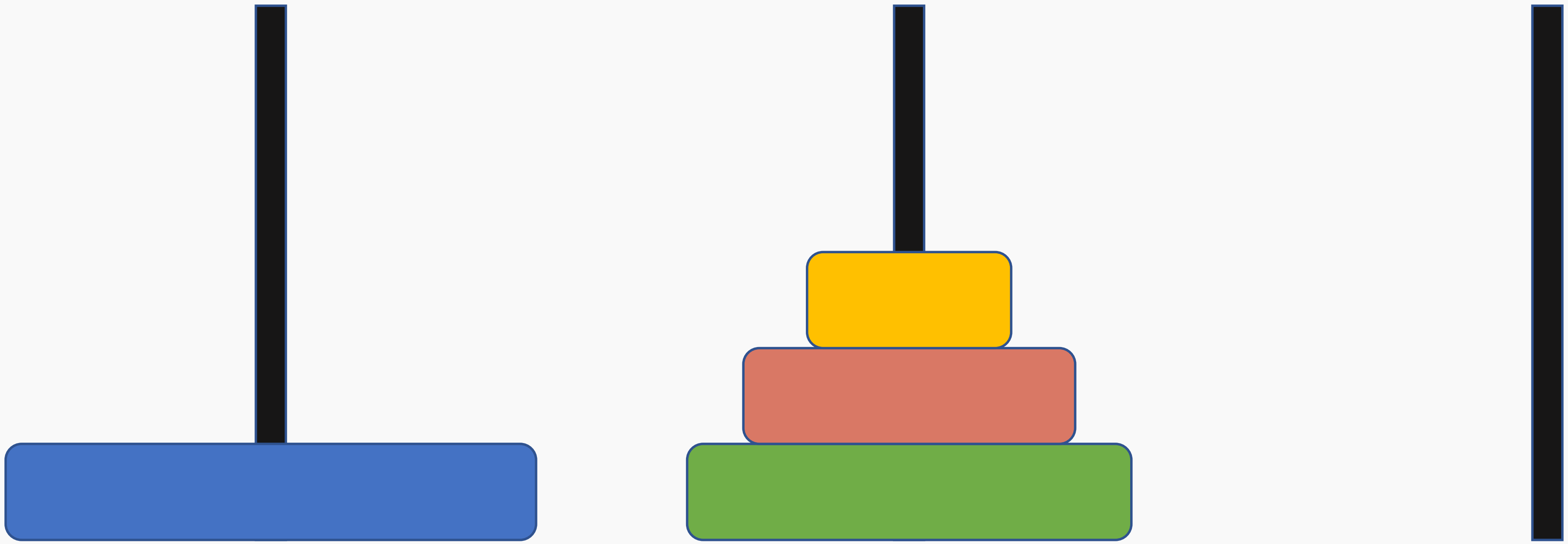
Problemă – Turnurile din Hanoi



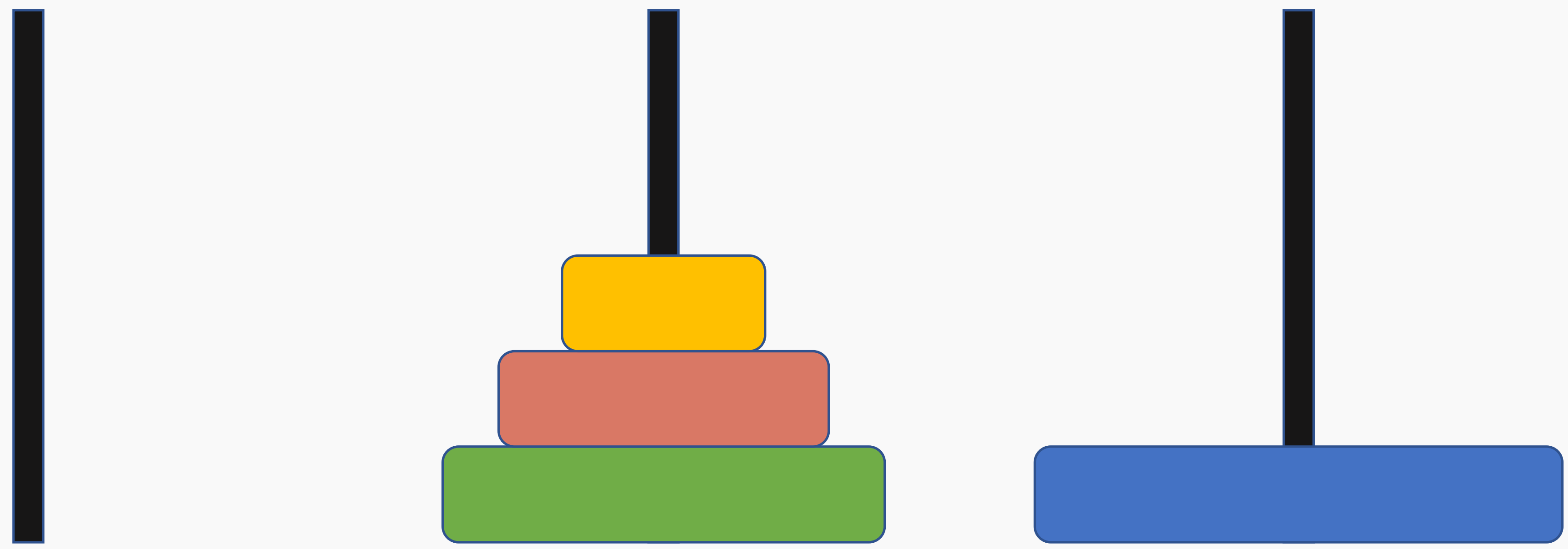
Problemă – Turnurile din Hanoi



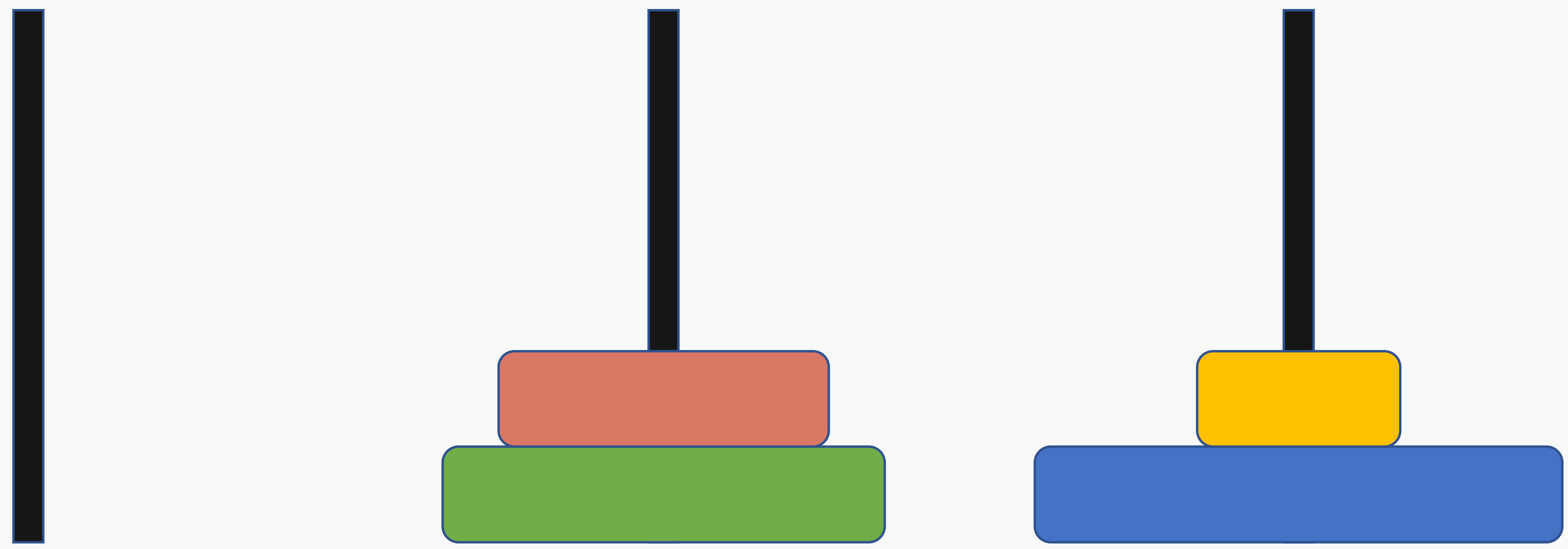
Problemă – Turnurile din Hanoi



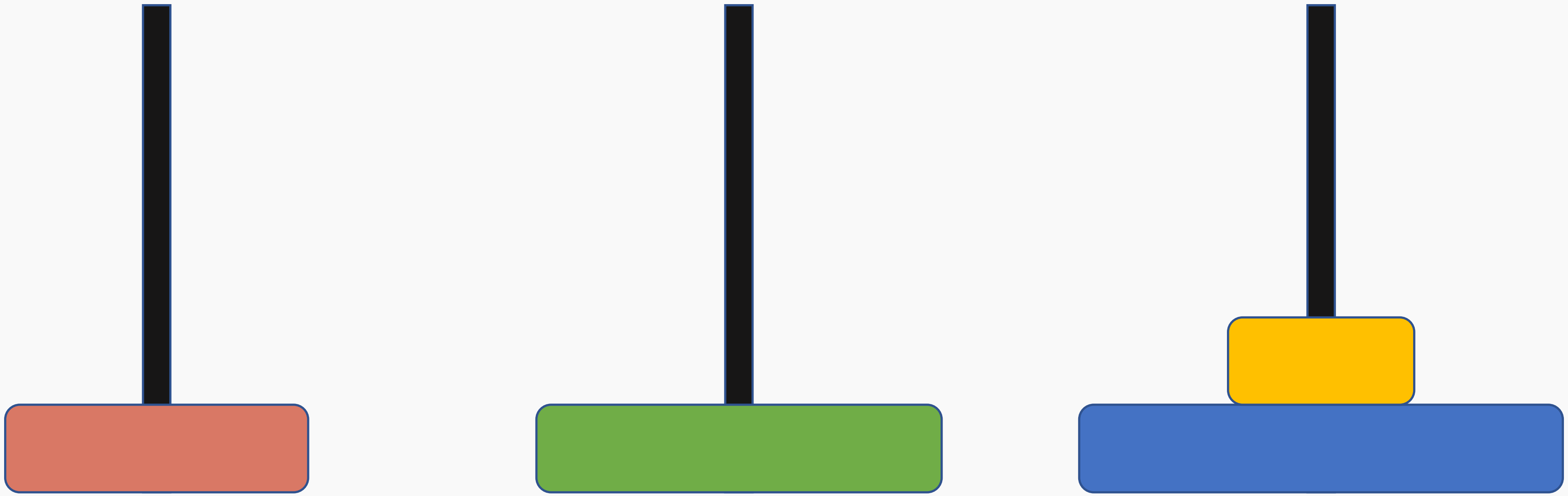
Problemă – Turnurile din Hanoi



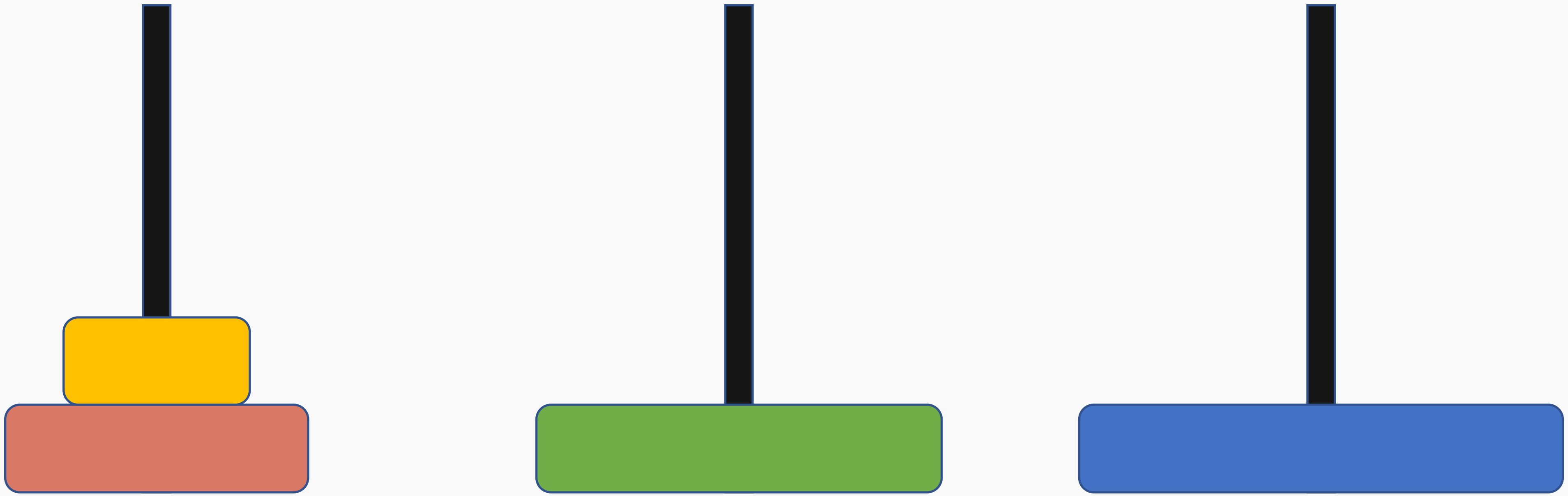
Problemă – Turnurile din Hanoi



Problemă – Turnurile din Hanoi

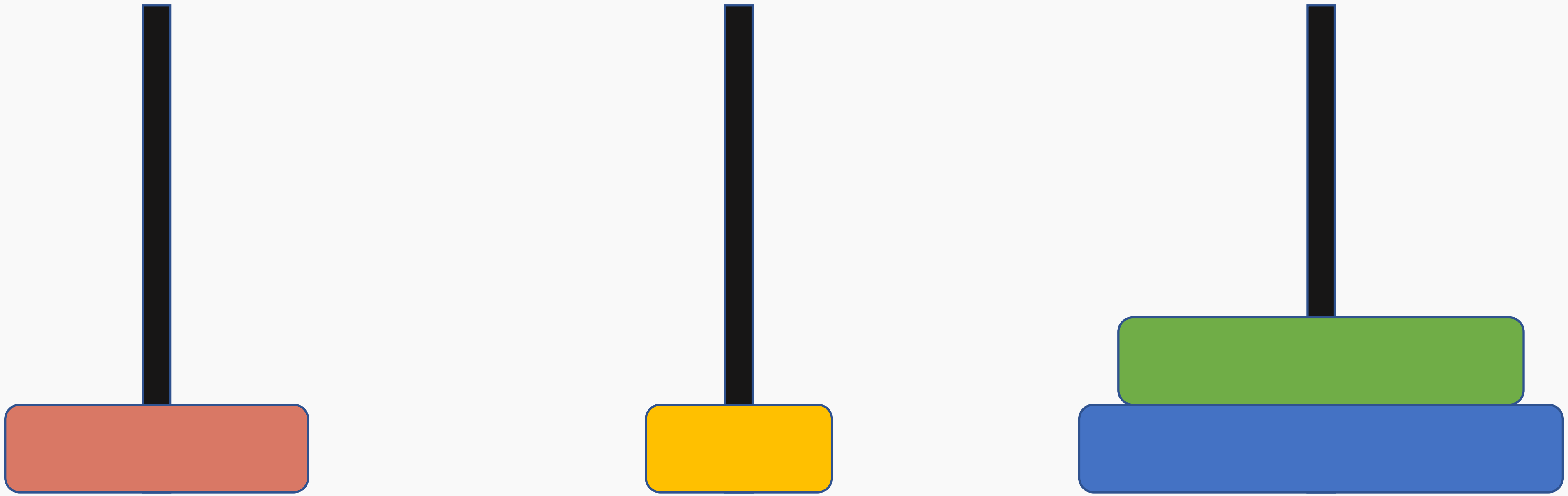


Problemă – Turnurile din Hanoi

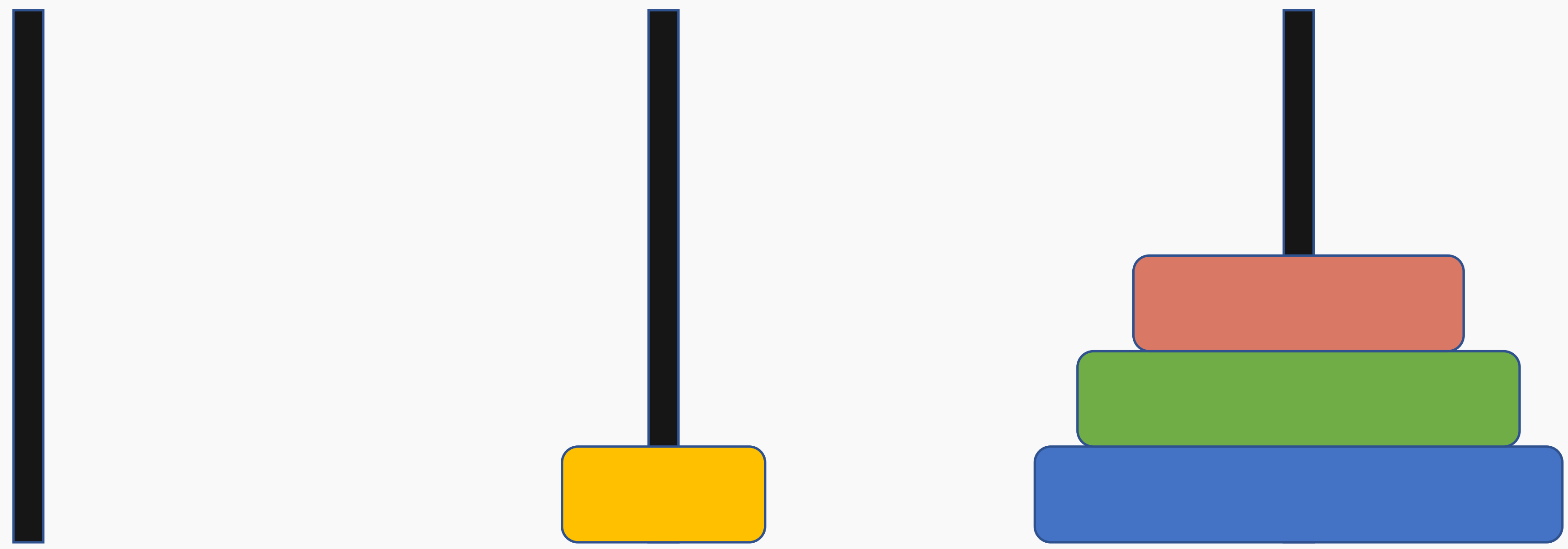


Problemă – Turnurile din Hanoi

Problemă – Turnurile din Hanoi



Problemă – Turnurile din Hanoi



Problemă – Turnurile din Hanoi

Problemă – Turnurile din Hanoi

Cazul simplu : N=1 se mută discul de pe A pe C.

Dacă N=2 se fac mutările : AB, AC, BC. Dacă N>2:

- Mutăm N-1 discuri de pe A pe B, utilizând tija C;
- Mutăm discul mare rămas pe tija C;
- Mutăm cele N–1 discuri de pe B pe C,utilizând tija A.

Problemă – Turnurile din Hanoi

```c
void hanoi(int n, char a, char b, char c)
{
    if (n == 1)
    {
        printf("Mutăm discul 1 de pe %c pe %c\n", a, c);
        return;
    }
    hanoi(n - 1, a, c, b);
    printf("Mutăm discul %d de pe %c pe %c\n", n, a, c);
    hanoi(n - 1, b, a, c);
}
```

Problemă – Turnurile din Hanoi - Complexitate

$$T(n) = 2T(n-1) + c$$

Problemă – Turnurile din Hanoi - Complexitate

$$\boldsymbol{T(n) = 2T(n-1) + c}$$

1. $f(n) = O\left(n^{\log_b a-\epsilon}\right), \epsilon > 0 \Rightarrow \boldsymbol{T(n) = \Theta\left(n^{\log_b a}\right)}$
2. $f(n) = \Theta\left(n^{\log_b a}\right) \Rightarrow \boldsymbol{T(n) = \Theta\left(n^{\log_b a} \lg n\right)}$
3. $f(n) = \Omega\left(n^{\log_b a+\epsilon}\right), \epsilon > 0,\ af\left(\frac{n}{b}\right) \leq cf(n), c < 1 \ \Rightarrow \boldsymbol{T(n) = \Theta(f(n))}$

Problemă – Turnurile din Hanoi - Complexitate

$$T(n) = 2T(n-1) + c$$

Problemă – Turnurile din Hanoi - Complexitate

$$T(n) = 2T(n-1) + c$$

$$T(n) = 2(2T(n-2) + c) + c$$

Problemă – Turnurile din Hanoi - Complexitate

$$T(n) = 2T(n-1) + c$$

$$T(n) = 2(2T(n-2) + c) + c$$

$$T(n) = 2(2(2T(n-3) + c) + c) + c$$

Problemă – Turnurile din Hanoi - Complexitate

$$T(n) = 2T(n-1) + c$$

$$T(n) = 2(2T(n-2) + c) + c$$

$$T(n) = 2(2(2T(n-3) + c) + c) + c$$

$$T(n) = \sum_{i=1}^{n} c2^{i}$$

Problemă – Turnurile din Hanoi - Complexitate

$$T(n) = 2T(n-1) + c$$

$$T(n) = 2(2T(n-2) + c) + c$$

$$T(n) = 2(2(2T(n-3) + c) + c) + c$$

$$T(n) = \sum_{i=1}^{n} c2^i = 2c(2^n - 1)$$

Problemă – Turnurile din Hanoi - Complexitate

$$T(n) = \sum_{i=1}^{n} c2^i = 2c(2^n - 1)$$

$$T(n) = \Theta(2^n)$$

Problemă – Rezolvarea unor ecuații

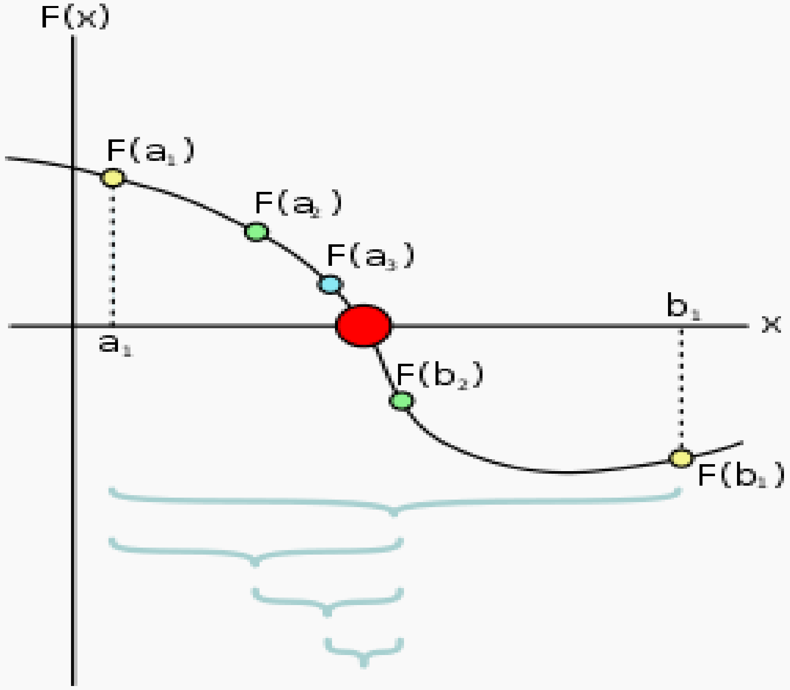
Metoda dihotomiei: cea mai simplă metodă de rezolvare a ecuaţiilor algebrice. Se presupune localizarea soluţiei $f(x) = 0$ în intervalul $[a, b]$.

Ipoteze: $f(x)$ continuă, x_0 este singura rădăcină în intervalul $[a, b]$, iar $f(a) \ * \ f(b) < 0$.

Se verifică valoarea lui f la jumătatea intervalului.

Dacă $|f(m)| < \varepsilon$, $x_0 = m$

Dacă $f(m) * f(a) < 0$, se caută o soluţie în $[a, m]$



Problemă – Exponențiere rapidă

Calculul valorii x^n poate fi efectuat cu mai puţin de $n-1$ operaţii de înmulţire, în mod recursiv, astfel:

$$x^n = \begin{cases} 1, & n = 0 \\ x^k \cdot x^k, & n = 2k \\ x \cdot x^k \cdot x^k, & n = 2k+1 \end{cases}$$

$$x^{10} = (x^5)^2 \;\;;\;\; x^5 = x(x^2)^2$$

$$x^2 = (x^1)^2 \;\;;\;\; x^1 = x * 1 * 1 \Rightarrow 4 \text{ înmulţiri}$$

$$T(n) = T\left(\frac{n}{2}\right) + c \Rightarrow \boldsymbol{T(n) = \theta(ln(n))}$$

Problemă – Multiplicare

$$X = \overline{x_1 x_2} = x_1 B^m + x_2$$
$$Y = \overline{y_1 y_2} = y_1 B^m + y_2$$

$$XY = (x_1 B^m + x_2)(y_1 B^m + y_2) = \alpha B^{2m} + \beta B^m + \gamma$$
$$\alpha = x_1 y_1$$
$$\beta = x_1 y_2 + x_2 y_1$$
$$\gamma = x_2 y_2$$

Problemă – Multiplicare

$$X = \overline{x_1 x_2} = x_1 B^m + x_2$$
$$Y = \overline{y_1 y_2} = y_1 B^m + y_2$$

$$XY = (x_1 B^m + x_2)(y_1 B^m + y_2) = \alpha B^{2m} + \beta B^m + \gamma$$
$$\alpha = x_1 y_1$$
$$\beta = x_1 y_2 + x_2 y_1$$
$$\gamma = x_2 y_2$$

$$\boldsymbol{T(n) = 4T\left(\frac{n}{2}\right) + \theta(n) \Rightarrow T(n) = \theta(n^2)}$$

Problemă – Multiplicare - Karatsuba

$$X = \overline{x_1 x_2} = x_1 B^m + x_2$$
$$Y = \overline{y_1 y_2} = y_1 B^m + y_2$$

$$XY = (x_1 B^m + x_2)(y_1 B^m + y_2) = \alpha B^{2m} + \beta B^m + \gamma$$
$$\alpha = x_1 y_1$$
$$\beta = (\boldsymbol{x_1} + \boldsymbol{x_2})(\boldsymbol{y_1} + \boldsymbol{y_2}) - \boldsymbol{\alpha} - \boldsymbol{\gamma}$$
$$\gamma = x_2 y_2$$

Problemă – Multiplicare - Karatsuba

$$X = \overline{x_1x_2} = x_1B^m + x_2$$
$$Y = \overline{y_1y_2} = y_1B^m + y_2$$

$$XY = (x_1B^m + x_2)(y_1B^m + y_2) = \alpha B^{2m} + \beta B^m + \gamma$$
$$\alpha = x_1y_1$$
$$\beta = \boldsymbol{(x_1 + x_2)(y_1 + y_2) - \alpha - \gamma}$$
$$\gamma = x_2y_2$$

$$\boldsymbol{T(n) = 3T\left(\frac{n}{2}\right) + \theta(n) \Rightarrow T(n) = \theta\left(n^{1.58}\right)}$$

Problemă – Înmulțire de matrici

$$\begin{pmatrix} \boldsymbol{A_{11}} & \boldsymbol{A_{12}} \\ \boldsymbol{A_{21}} & \boldsymbol{A_{22}} \end{pmatrix} \begin{pmatrix} \boldsymbol{B_{11}} & \boldsymbol{B_{12}} \\ \boldsymbol{B_{21}} & \boldsymbol{B_{22}} \end{pmatrix} = \begin{pmatrix} \boldsymbol{C_{11}} & \boldsymbol{C_{12}} \\ \boldsymbol{C_{21}} & \boldsymbol{C_{22}} \end{pmatrix}$$

$$\begin{aligned} C_{11} &= A_{11}B_{11} + A_{12}B_{21} \\ C_{12} &= A_{11}B_{12} + A_{12}B_{22} \\ C_{21} &= A_{21}B_{11} + A_{22}B_{21} \\ C_{22} &= A_{21}B_{12} + A_{22}B_{22} \end{aligned}$$

$$T(n) = 8T\left(\frac{n}{2}\right) + \Theta(n^2); 2 < log_2 8 = 3$$

$$\boldsymbol{T(n) = \theta\left(n^{log_2 8}\right) = \theta\left(n^3\right)}$$

Problemă – Înmulțire de matrici – Strassen

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix} = \begin{pmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{pmatrix}$$

$$M_1 = (A_{11} + A_{22})(B_{11} + B_{22})$$
$$M_2 = (A_{21} + A_{22})B_{11}$$
$$M_3 = A_{11}(B_{12} - B_{22})$$
$$M_4 = A_{22}(B_{21} - B_{11})$$
$$M_5 = (A_{11} + A_{12})B_{22}$$
$$M_6 = (A_{21} - A_{11})(B_{11} + B_{12})$$
$$M_7 = (A_{12} - A_{22})(B_{21} + B_{22})$$

$$C_{11} = M_1 + M_4 - M_5 + M_7$$
$$C_{12} = M_3 + M_5$$
$$C_{21} = M_2 + M_4$$
$$C_{22} = M_1 - M_2 + M_3 + M_6$$

Problemă – Înmulțire de matrici – Strassen

$$M_1 = (A_{11} + A_{22})(B_{11} + B_{22})$$
$$M_2 = (A_{21} + A_{22})B_{11}$$
$$M_3 = A_{11}(B_{12} - B_{22})$$
$$M_4 = A_{22}(B_{21} - B_{11})$$
$$M_5 = (A_{11} + A_{12})B_{22}$$
$$M_6 = (A_{21} - A_{11})(B_{11} + B_{12})$$
$$M_7 = (A_{12} - A_{22})(B_{21} + B_{22})$$

$$C_{11} = M_1 + M_4 - M_5 + M_7$$
$$C_{12} = M_3 + M_5$$
$$C_{21} = M_2 + M_4$$
$$C_{22} = M_1 - M_2 + M_3 + M_6$$

$$T(n) = 7T\left(\frac{n}{2}\right) + \Theta(n^2); 2 < log_2 7$$

$$\boldsymbol{T(n) = \theta\left(n^{log_2 7}\right) = \theta\left(n^{2.81}\right) < \theta\left(n^3\right)}$$