(2)

Сложност на алгоритъм

$$f(n) = \underbrace{2n^2 + 300}_{\text{стъпки, които алгоритъма ще извърши}}$$

n – големина на входа

$$\text{Alg 1} \quad vs \quad \text{Alg 2}$$

$$f(n) \qquad\qquad g(n)$$

$$f \; vs \; g$$

Коя ф-я нараства по-бавно?

Интересуваме се при $n \to \infty$!

При малки стойности на n

повечето алгоритми ще се справят ОК

$$\lim_{n\to\infty} \frac{f}{g} \quad \begin{array}{lll} \infty & f > g & \boxed{g} \\ const & f \approx g & \boxed{f \mid g} \\ 0 & f < g & \boxed{f} \end{array}$$

Ако имаме ограничение на входа?

Пример:

Сортиране на масиви с най-много 250 елемента

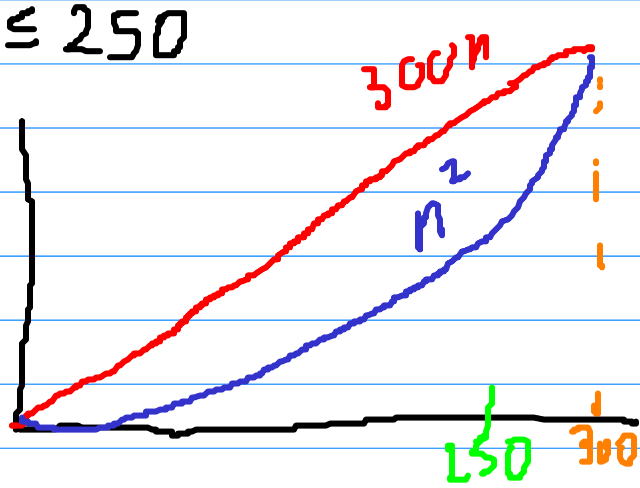
$n \le 250$

Alg 1	Alg 2
$\boxed{300 \cdot n}$	$\boxed{n^2}$

- при $n \to \infty$ $\boxed{\text{Alg 1} \checkmark}$

$$\underbrace{300n}_{\theta(n)} < \underbrace{n^2}_{\theta(n^2)}$$

- при $n \le 250$



$\boxed{\text{Alg 2} \checkmark}$

Анализ на итеративни алгоритми

- Чрез **сумиране**

Общи правила:

$$\sum_{i=1}^{n} i^{a} \qquad \begin{array}{ll} a \geq 1 & \Theta(n^{a+1}) \\ a = -1 & \Theta(\log(n)) \\ 0 \leq a < 1 & \Theta(n) \end{array}$$

$$\sum_{i=1}^{n} i^{1} = 1 + 2 + 3 \dots + n = \frac{n(n+1)}{2} = \Theta(n^2)$$

$$\sum_{i=1}^{n} i^{2} = 1^2 + 2^2 + \dots n^2 = \frac{n(n+1)(2n+1)}{6} = \Theta(n^3)$$

$$\sum i^{3} = 1^3 + 2^3 + \dots n^3 = \frac{n^2(n+1)^2}{4} = \Theta(n^4)$$

$$\sum_{i=1}^{n} i^{-1} = \underbrace{\frac{1}{1} + \frac{1}{2} + \frac{1}{3} \dots \frac{1}{n}}_{H_n \text{ - хармонично число}} \qquad \Theta(\log(n))$$

$$\sum_{\substack{i=1 \\ i*=2}}^{n} 1 = \theta(\log(n))$$

$$\underbrace{\underset{i=1}{1} + \underset{i=2}{1} + \underset{i=4}{1} + \underset{i=8}{1} \ldots + 1}_{\log(n)}$$

3

НАЙ-добър случай – безсмислен

Среден случай – най-ценен

НАЙ-лош случай – смята се по-лесно от средния случай

- Елементарна операция
  → не зависи от големината на входа

Задачи: Анализирайте дадените алгоритми.

Зад 1

$$\sum_{i=1}^{n} \underbrace{\sum_{j=1}^{n} 1}_{n} + \sum_{k=0}^{\frac{n}{2}} 1 =$$

$$= \sum_{i=1}^{n} n + \frac{n}{2} =$$

$$n^2 + \frac{n}{2} \simeq \Theta(n^2)$$

~~Зад 2~~

$i=1$	1
$i=2$	4
$i=3$	9
$\vdots$	

$$\sum_{i=1}^{n} i^2 + \Theta(1) \simeq \Theta(n^3)$$

$$= \Theta(n^3)$$

~~Зад 3~~

$$1 + 2 + 4 + 8 \dots + \frac{n}{8} + \frac{n}{4} + \frac{n}{2} + n$$

$$= \sum_{\substack{i=n \\ i/=2}}^{1} \frac{n}{i} \simeq 2n - 1 \quad \boxed{\Theta(n)}$$

$n = 8$ $\quad 2n-1$

$1 + 2 + 4 + 8 = 15$

$n = 16$ $\quad 2n-1$

$1 + 2 + 4 + 8 + 16 = 31$

~~зад 4~~ ↓

~~зад 5~~

$$\sum_{\substack{i=1 \\ i*=2}}^{n} \sum_{j=1}^{n} 1 = \sum_{\substack{i=1 \\ i*=2}}^{n} n =$$

$$\underbrace{n + n + n \ldots n}_{\log(n)} \qquad \Theta(n.\log(n))$$

Ако $n=16$

$$\left.\begin{array}{l} i=1 \\ i=2 \\ i=4 \\ i=8 \\ i=16 \end{array}\right\} \log(n)+1 = \Theta(\log(n))$$

~~зад 6~~

$$\sum_{i=1}^{n} \sum_{\substack{j=1 \\ j+=i}}^{n} 1 = \sum_{i=1}^{n} \frac{n}{i} = n \underbrace{\sum_{i=1}^{n} \frac{1}{i}}_{\log(n)}$$

$$\Theta(n.\log(n))$$

~~зад 7~~

$$\underbrace{\sum\sum\sum 1}_{\theta(n^4)} + \underline{2^n} + \underbrace{const}_{\theta(1)}$$

$$\boxed{\theta(2^n)}$$

! работа ull (64 бита)
2^{63} → горна граница
↳ можем да приемем за константа ?

~~зад 4~~

за f : $\theta(n)$
за task4 (n)
↓
извиква n пъти f

$$\Rightarrow task4 \rightarrow \underbrace{f, f, f \ldots f}_{n \text{ пъти}}$$

$$\cancel{\theta(n^2)}$$

По време на изпълнението на task4 променливата res е 0 или 1

Приемаме f в рамките на task4 за $\theta(1)$

$$\sum_{\substack{i=n \\ i=1}}^{i=0} 1 = \underbrace{1 + 1 + 1 \cdots + 1}_{\log(n)} \quad \theta(\log(n))$$

Стойности на i $\longrightarrow$ $\underbrace{n, \frac{n}{2}, \frac{n}{4} \cdots 1}_{\log(n)}$

Анализ на някои алгоритми

1. Алгоритми за търсене

1) Линейно търсене

Най-добър $\Theta(1)$

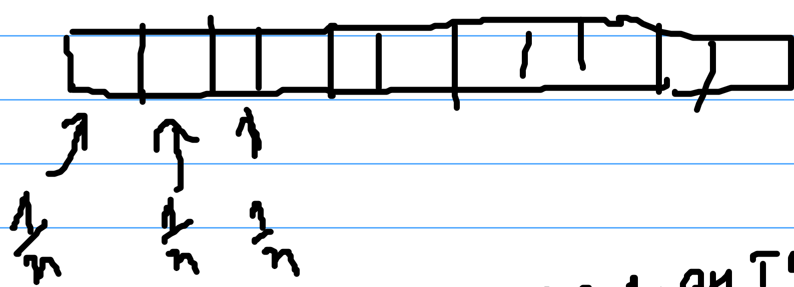
Среден $\Theta(n)$

Най-лош $\Theta(n)$

Търсим елементът x.

Приемаме, че го има точно веднъж.

→ Къде може да е x?



Вероятността елемента x да е на k-та позиция е $\frac{1}{n}$

стъпки	1,	2,	3,	4	...	n
вер.	$\frac{1}{n}$	$\frac{1}{n}$	$\frac{1}{n}$	$\frac{1}{n}$		$\frac{1}{n}$

вероятността x да е на 3-та позиция

Пример за очаквана стойност хвърляне на зар - брой точки.

Точки:	1	2	3	4	5	6
Вероятност:	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$

↳ Очакваната стойност на получените точки:

$$1*\frac{1}{6} + 2*\frac{1}{6} + 3*\frac{1}{6} + 4*\frac{1}{6} + 5*\frac{1}{6} + 6*\frac{1}{6} = \boxed{3,5}$$

1000 зара $\frac{\text{сума на точките}}{1000} \approx 3.5$

1 2 3 4 5 6

средно на хвърляне получаваме 3.5 точки

Връщаме се на пресмятане на средния случай на линейното търсене

$$1*\frac{1}{n} + 2*\frac{1}{n} + 3*\frac{1}{n} \dots n*\frac{1}{n} =$$

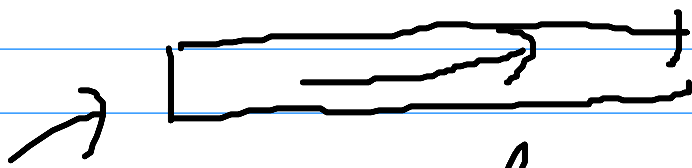
$$= 1 * \frac{1}{n} + 2 * \frac{1}{n} + \dots n * \frac{1}{n} =$$

$$= \frac{1+\dots n}{n} = \frac{\frac{n(n+1)}{2}}{n} = \frac{n+1}{2} \approx \Theta(n)$$

среден брой стъпки

$\Rightarrow$ В средния случай: $\Theta(n)$

2/ Двоично търсене



сортиран масив.

Най-добър: $\Theta(1)$ елемент в средата

Среден: Домашно $\Theta(\log(n))$

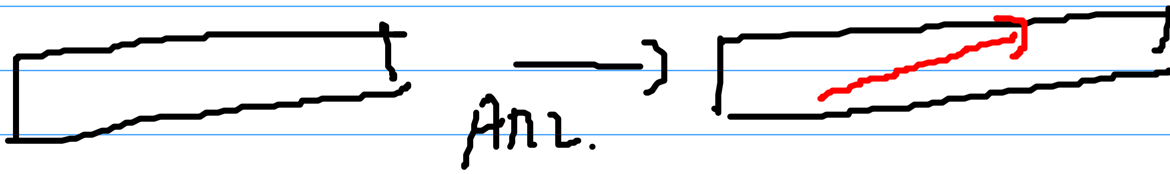
Най-лош: $\Theta(\log(n))$

$$n \xrightarrow{\Theta(1)} \frac{n}{2} \xrightarrow{\Theta(1)} \frac{n}{4} \xrightarrow{\Theta(1)} \frac{n}{8} \xrightarrow{\Theta(1)} \dots \xrightarrow{\Theta(1)} 1$$

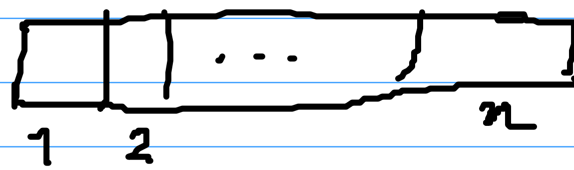
3) Търсене чрез скачане
4) експ. търсене
5) интерполационно търсене

Не ги изучаваме

11) Анализ на Алгоритми за Сортиране



arr[1...n]



<u>def</u>: Инверсия в масив

$i, j \in \{1 \dots n\}$

$i < j \wedge arr[i] > arr[j]$

1, 9, 6, 10

1 2 3 4

инверсия

$2 < 3 \wedge 9 > 6$

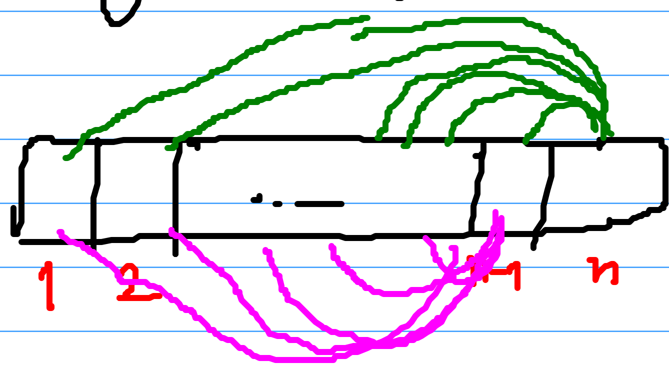
Масивът е сортиран, когато в него няма инверсии.

1, 9, 0, 5

Колко инверсии има?

3 инверсии

Колко най-много инверсии може да има в масив?



Всички наредени двойки (i, j) $i<j$ ще са в инверсия

$$(n-1)+(n-2)+(n-3)\dots+1=$$

$$=\sum_{i=1}^{n-1} i=\frac{n(n-1)}{2}=\binom{n}{2}$$

Сортиране по метода на мехурчето

Bubble Sort:

сложност време:

- най-добро: $\Theta(n)$
- среден: $\Theta(n^2)$
- най-лош: $\frac{n(n-1)}{2} = \Theta(n^2)$

сложност по памет: $\Theta(1)$

стабилност

→ Ако масивът е сортиран наобратно:

$\Rightarrow$ имаме $\frac{n(n-1)}{2}$ инверсии

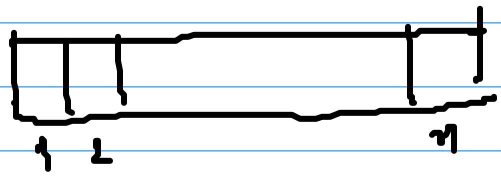
При една стъпка премахваме точно 1 инверсия

$\Rightarrow$ най-лош случай $\Theta(n^2)$

Анализ в средния случай:

Колко средно инверсии има в масив?

Нека Б.О.О. сортираме масив с числата $\{1 \dots n\}$



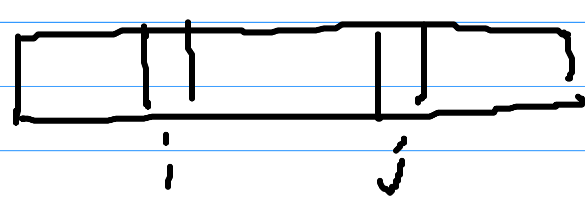
Възможните масиви са $n!$

Средно колко инверсии има в масива

Произволна двойка от индекси

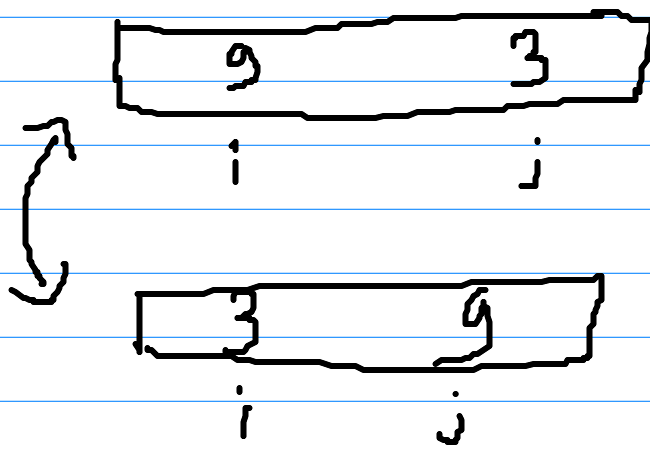
(i, j)

$i < j$



Каква е вероятността (i, j) да е в инверсия?

вероятност (i,j) инверсия $= \dfrac{\text{перм. в които } (i,j) \text{ е инв.}}{\text{всички перм } (= n!)}$



Има биекция м/у пермутациите, в които (i, j) Е инверсия и пермутациите, в които (i, j) Не е инверсия!

$\Rightarrow$ пермутациите, в които (i, j) е инверсия са точно половината $\left(\frac{n!}{2}\right)$

$\Rightarrow$ Вероятността в произволен масив (i, j) да е инверсия $\frac{n!}{2} / n! = \frac{1}{2}$

Колко средно инверсии има в масив?

$\frac{n(n-1)}{2}$

- наредени двойки индекси (i,j) $i<j$
- възможности за инверсии

Всяка инверсия има вероятност $\frac{1}{2}$

$\Rightarrow$ Средно имаме $\underbrace{\frac{n(n-1)}{2}}_{\text{брой възможности за инверсия}} \cdot \underbrace{\frac{1}{2}}_{\text{вероятност за инв.}} =$

Среден брой инверсии: $= \frac{n \cdot (n-1)}{4}$

Метода на мехурчето премахва на една стъпка една инверсия

стъпки $\geq \frac{n(n-1)}{4} = \Theta(n^2)$

$$\left| \underset{i<j}{(i, j)} \right| = \binom{n}{2}$$