۹۸۳۱۰۱۶ سارا تاجرنیا

۱-

فرض: بدنه حلقه while در کدام ز حلقه for به اندازه t_j تکرار می‌شود

← هرچه نابجایی‌ها بیشتر باشد زمان اجرای الگوریتم بیشتر است

insertion sort:

		هزینه
1-	for j=2 to n do	$c_1 \times n$
2-	key ← A[j]	$c_2(n-1)$
3-	i ← j-1	$c_3(n-1)$
4-	while i>0 && key<A[i]	$c_4(t_j+1)$
5-	A[i+1] ← A[i]	$c_5(t_j)$
6-	i ← i-1	$c_6(t_j)$
7-	A[i+1] ← key	$c_7(n-1)$

الف)

$$T(n) = c_1 n + c_2(n-1) + c_3(n-1) + c_7(n-1) + \sum_{j=2}^{n} c_4(t_j+1) + \sum_{j=2}^{n} c_5 t_j + \sum_{j=2}^{n} c_6 t_j \longrightarrow \text{زمان الگوریتم}$$

$$T_{(i,j)} = \sum_{j=2}^{n} c_4(t_j+1) + \sum_{j=2}^{n} c_5 t_j + \sum_{j=2}^{n} c_6 t_j \longrightarrow \text{تعداد نابجایی‌ها}$$

ب) زمانی که بین x_i و x_j یک نابجایی وجود دارد پس x_j باید $j-1$ بار به سمت چپ حرکت کند پس حداقل $j-1$ بار جابجایی داریم.

$x_1 \; x_2 \; — \; x_i \; \circ \circ \circ \circ - x_j$ حداقل $j-1$ بار

-2

```
1- x=0                          → c1
2- for (i=1, i<=n, i++){        → c2 (n+1)
3-     for (j=1, j<=n, j++) x++;  → c3 (n+1)×n + c4 (n)(n)
4-     j=1                      → c5 (n)
5-     while (j<n){             → c6 (n)(log2 n +1)
6-     x++; j=j*2;              → c7 (n)(log2 n) + c8 (n)(log2 n)
7-     }
8- }
```

$$T(n) = c_1 + c_2(n+1) + c_3 n(n+1) + c_4(n^2) + c_5 n + c_6(n)(\log_2 n + 1) + c_7(n)(\log_2 n) + c_8(n)(\log_2 n)$$

3.

$x + 3x + 6x = 1 \rightarrow x = \frac{1}{10}$

n ← تعداد داده‌ها

a در $\frac{1}{3}$ اول باشد $\rightarrow x\left(\frac{n}{3} \times \frac{1}{2}\right)$

a در $\frac{1}{3}$ دوم باشد $\rightarrow 3x\left(\frac{n}{3} + \left(\frac{n}{3} \times \frac{1}{2}\right)\right)$

a در $\frac{1}{3}$ سوم باشد $\rightarrow 6x\left(\frac{2n}{3} + \left(\frac{n}{3} \times \frac{1}{2}\right)\right)$

بهترین حالت این است که یکبار اجرا شود.

بدترین حالت این است که n بار اجرا شود.

$$\frac{nx}{6} + \frac{3nx}{2} + 5nx = \frac{40nx}{6} \xrightarrow{x=\frac{1}{10}} \frac{2n}{3}$$

حالت متوسط برابر $\frac{2n}{3}$ است.

4.

آرایه نامرتب

```
quicksort(arr[], low, high)
    if (low < high)
        pi = partition(array, low, high)
        quickSort(array, low, pi-1)
        quicksort(array, pi+1, high)
```

پیچیدگی زمانی آن ← $n\log(n) + n$ است ← $O(n\log(n))$

حالا که با توجه به کد آرایه نامرتب مرتب شد بقیه را از کد پایین می‌رویم

آرایه مرتب

```
j = n-1
for (i=0; i<n; i++)
    if (array[i] + array[j] > k)
        j--;
    else if (array[i] + array[j] < k)
        i++;
    else
        return array[i], array[j]
```

دارای حلقه for است پس

پیچیدگی زمانی آن ← $O(n)$ است

$O(n\log(n)) > O(n)$

a) $\lg(n!) = \Theta(\lg(n^n)) \rightarrow n! \leqslant n^n \rightarrow \lg n! \leqslant \lg n^n \rightarrow \lg n! = O(\lg(n^n))$

$\rightarrow \exists x \in \mathbb{R} \mid x n^n \leqslant n! \rightarrow \lg(x n^n) \leqslant \lg(n!) \rightarrow \lg(n!) = \Omega \lg(n^n)$

$\Big\} \lg(n!) = \Theta \lg(n^n)$

b) $n^{\frac{1}{\lg n}} = \Theta(1)$ هر دو عدد ثابت اند $\rightarrow$ رشد برابر دارند

$= 2$

c) $n! = \omega(2^n)$ $\quad n! = \sqrt{2\pi n} \times \left(\frac{n}{e}\right)^n = n^{n+\frac{1}{2}} \times e^{-n} \times \sqrt{2\pi}$

$n + \frac{1}{2} \geqslant n \rightarrow n > 2 : n! > 2^n \rightarrow n! = \omega(2^n)$

d) $n! = O(n^n)$ $\quad n! = \sqrt{2\pi n} \times \left(\frac{n}{e}\right)^n = n^n \times e^{-n} \times \sqrt{2\pi n} = \frac{n^n \times \sqrt{2\pi n}}{e^n}$

$n \in R, n > 1 \quad e^n > \sqrt{2\pi n} \rightarrow 1 > \frac{\sqrt{2\pi n}}{e^n} \rightarrow n^n > \frac{n^n \sqrt{2\pi n}}{e^n} \rightarrow n^n > n! \rightarrow n! = O(n^n)$

-6

a) $f_{(n)} = O(g_{(n)}) \rightarrow g_{(n)} = O(f_{(n)})$ نادرست $f_{(n)} = n \quad g_{(n)} = n^2$

b) $f_{(n)} + g_{(n)} = \Theta(\min(f_{(n)}, g_{(n)}))$ نادرست $f_{(n)} = n \quad g_{(n)} = n^2$

c) $f_{(n)} = O(g_{(n)}) \rightarrow \lg(f_{(n)}) = O(\lg(g_{(n)}))$, $\forall n : \lg(g_{(n)}) \geqslant 1$ and $f_{(n)} \geqslant 1$

$f_{(n)} \leqslant g_{(n)} \rightarrow \lg(f_{(n)}) \leqslant \lg(g_{(n)}) \rightarrow \lg(f_{(n)}) = O \lg(g_{(n)})$ درست

d) $f_{(n)} = O(g_{(n)}) \rightarrow 2^{f(n)} = O(2^{g(n)})$

$f_{(n)} \leqslant g_{(n)} \rightarrow 2^{f(n)} \leqslant 2^{g(n)} \rightarrow 2^{f(n)} \leqslant O(2^{g(n)})$ درست

e) $f_{(n)} = O(f_{(n)}^2)$ $\quad f_{(n)} = \frac{1}{n} \quad f_{(n)} = \frac{1}{n^2}$ نادرست

f) $O(g_{(n)}) = f_{(n)} \rightarrow g_{(n)} = \Omega(f_{(n)})$ درست خاصیت پادتقارنی

g) $f_{(n)} = \Theta(f(\frac{n}{2}))$ درست $\rightarrow$ ضریب تاثیری در رشد ندارد و تنها توان مهم است

h) $f_{(n)} + O(f_{(n)}) = \Theta(f_{(n)})$ $\quad f_{(n)} = n \quad O(f_{(n)}) = n^2$ نادرست

A	B	O	o	Ω	ω	θ
n^2	n^3	✓	✓	✗	✗	✗
$\lg^k n$	n^ε	✓	✓	✗	✗	✗
n^k	c^n	✓	✓	✗	✗	✗
2^n	$2^{n/2}$	✗	✗	✓	✓	✗
$n^{\lg c} = c^{\lg n}$	$c^{\log n}$	✗	✗	✓	✓	✗
$4^{\lg n} = n^2$	n^2	✓	✗	✓	✗	✓
$n!$	$n2^n$	✗	✗	✓	✓	✗
$\sqrt{2}^{\lg n} = \lg^{1/2}$	$2^{\sqrt{2\lg(n)}} =$	✗	✗	✓	✓	✗
$(\lg(n))!$	2^{2^n}	✓	✓	✗	✗	✗
$n^{\lg(\lg(n))}$	$(\lg(n))^{\lg(n)}$	✓	✗	✓	✗	✓