۱) الف)

a) $f(n) = n^{\log n} \quad g(n) = (\log n)^n$

فرض می کنیم $f(n) = O(g(n))$: $\quad \exists\, n_0, c_0 \;\; \forall n \geq n_0 \quad 0 \leq f(n) \leq c\, g(n)$

$c=1$ فرض کرده و به دنبال n_0 می گردیم.

$$n^{\log n} \leq (\log n)^n$$

$$\log^2 n \leq n \log\log n$$

چون $\log\log n$ تابعی صعودی است و به ازای مقادیری (مثلاً $n_0 \geq 10^{10}$) عبارت بزرگتر مساوی ۱ می شود، پس اگر ثابت کنیم $\log^2 n \leq n$ عملاً $\log^2 n \leq n\log\log n$ را هم اثبات کردیم

پس $n_0 = 10^{10}$: $\quad \log^2 n \leq n \Rightarrow \log n \leq \sqrt{n} \Rightarrow n \leq 10^{\sqrt{n}}$

$$\Rightarrow n^2 \leq 10^n$$

حال ثابت می کنیم اگر $n^2 \leq 10^n$ باشد $(n+1)^2 \leq 10^{n+1}$ است:

$$n^2 \leq 10^n \Rightarrow (n+1)^2 \leq 10^{n+1}$$

$$(n+1)^2 = \underbrace{n^2} \times \left(\frac{n+1}{n}\right)^2$$

$$10^{n+1} = \underbrace{10} \times 10^n$$

: $\left(\frac{n+1}{n}\right)^2 \leq 10 \Rightarrow \left(\frac{n+1}{n}\right) \leq \sqrt{10}$

فرض کرده بودیم $n \geq 10^{10}$ است $\quad \Rightarrow n+1 \leq \sqrt{10}\, n \Rightarrow \left(\frac{1}{\sqrt{10}-1}\right) \leq n$

پس فرض ما درست بوده و ثابت می شود فرض اولیه ما نیز درست است.

b) $f(n) = \frac{n^2}{\sqrt{\log n}}$ $\quad g(n) = n\log^2 n$

فرض می‌کنیم: $g(n) = O(f(n))$ $\quad \exists c, n_0 \; \forall n \geq n_0 \; 0 \leq g(n) \leq c\, f(n)$

$n\log^2 n \leq \frac{n^2}{\sqrt{\log n}} \Rightarrow n\log^3 n \leq n^2 \Rightarrow n(-\log^3 n + n) \geq 0$

می‌دانیم $n \in \mathbb{N} \geq 0$ باید ثابت کنیم $n - \log^3 n \geq 0$ است:

$n - \log^3 n \geq 0 \Rightarrow \sqrt[3]{n} - \log n \geq 0 \Rightarrow 10^{\sqrt[3]{n}} - n \geq 0$

مشابه این در سوال قبل ثابت شد؛ پس با فرض $c_0 = 1$ و $n_0 = 10$ به درستی فرض

خود می‌رسیم.

ب) ۱: X $\quad f(n) = n^2, \; g(n) = -n^2 \Rightarrow n^2 - n^2 \neq \Theta(n^2)$

۲: X $\quad f(n) = n^2, \; g(n) = n^3, \; h(n) = n^{-1}$

$n^2 < n^3 \Rightarrow n^{-2} \not< n^{-3}$

۳: ✓ $\quad \exists c_1, c_2, n_0 \; \forall n \geq n_0 \quad c_2 f(n) \leq f(n) + O(f(n)) \leq c_1 f(n)$

$g(n) = O(f(n)) \Rightarrow \exists c_0, n_0' \; \forall n \geq n_0' \quad 0 \leq g(n) \leq c_0 f(n)$

اگر $c_0 = \frac{1}{2}$ و $n_0 = n_0'$ در نظر بگیریم داریم:

$$g(n) \le \frac{1}{2} f(n) \Rightarrow g(n) + f(n) \le \frac{3}{2} f(n)$$

$$\left.\begin{array}{l} g(n) \le \frac{1}{2} f(n) \\ f(n) - g(n) \le f(n) + g(n) \end{array}\right\} \Rightarrow \frac{1}{2} f(n) \le f(n) - g(n) \le f(n) + g(n)$$

$$\Rightarrow \underbrace{\frac{1}{2}}_{c_2} f(n) \le f(n) + g(n) \le \underbrace{\frac{3}{2}}_{c_1} f(n)$$

با پیدا شدن c_1 ، c_2 و n_0 صورت مسأله اثبات می‌شود .

$$f(n) = e^{2n} \Rightarrow e^{2n} \overset{?}{=} \Theta(e^n)$$

۴: X

$$c_2 e^n \le e^{2n} \le c_1 e^n$$

مشاهده می‌شود هیچ c_1 و c_2 که عدد ثابت باشد پیدا نمی‌شود

۳) الف) $T(n) = (2^1 - 2^0)n + (2^2 - 2^1)\frac{n}{2} + (2^3 - 2^2)\frac{n}{2^2} + \dots + 2^{\log_2 n}\left(\frac{n}{2^{\log_2 n}}\right)$

$= \underbrace{n + n + \dots + n}_{K} \quad : K = \log_2 n \quad \Rightarrow T(n) = n \times \log_2 n$

ب) $T(n) = n + \frac{n}{2} + \frac{n}{3} + \dots + \frac{n}{n} < \underbrace{n + n + n + \dots + n \log_2 n}_{H(n)}$

$H(n) = O(n \log n) \Rightarrow T(n) = O(n \log n)$

ج) $T(n) = 1 + 2 + 3 + \dots + n = \frac{n(n+1)}{2} = \frac{1}{2}n^2 + \frac{1}{2}n$

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

د) I) $1 \leq a_1 < a_2 < \dots < a_m$ $\quad A(n) = a_1 + a_2 + \dots + a_m = n$

$B(n) = 1 + 2 + \dots + m = n'$

مشاهده می‌شود که به ازای هر i، $B(i) < A(i)$ پس:

$\frac{m(m+1)}{2} < (a_1 + a_m)\frac{m}{2} = n \Rightarrow \frac{m^2 + m}{2} = n \Rightarrow m^2 + m - 2n = 0$

$\Rightarrow m = \frac{-1 \pm \sqrt{1 + 8n}}{2} \Rightarrow m = O(\sqrt{n})$

۳) الف)

$$i=1 \Rightarrow n-1$$

$$i=2 \Rightarrow \frac{n-3}{2}$$

$$\vdots$$

$$i=k \Rightarrow \frac{n-2k+1}{k}$$

$$\vdots$$

$$i=\frac{n}{2} \Rightarrow \frac{1}{\frac{n}{2}}$$

$$\Rightarrow T=\sum_{i=1}^{\frac{n}{2}} \frac{n+1}{i} -2 = -n+\sum_{i=1}^{\frac{n}{2}} \frac{n+1}{i} = -n+(n+1)\underbrace{\sum_{i=1}^{\frac{n}{2}} \frac{1}{i}}_{I}$$

از مورد ب سوال قبل می‌دانیم $I=O(\log n)$ $\Rightarrow$ $T=-n+n\log n+\log n$

$$\Rightarrow T=O(n\log n)$$

ب) احتمال درستی شرط if $(a==1)$ در هر iterate $\frac{1}{i}$ است پس:

$$T=\sum_{i=1}^{n}\frac{1}{i}$$

با توجه به سوال قبل مورد ب) حاصل T می‌شود: $T=O(\log n)$

۳) ج) ۱: اگر مورد ۲ را ثابت کنیم با توجه به بیان زیر مورد ۱ را هم ثابت کرده ایم:

$$\log n < n \Rightarrow \log\log n < \log n \Rightarrow n\log\log n < n\log n$$

حال وقتی تابعی از مرتبه $O(n\log\log n)$ باشد پس از مرتبه $O(n\log n)$ نیز است.

۲:

```
for (i=2 ; i<n ; i++)   → n+1
    j = i               → n
    while ( j ≤ n )     → n*a
        j = j*j
```

شرط while $\left((j^2)^a < n\right)$ حداکثر تا $(j^2)^a = n$ برقرار است پس:

$$(j^2)^a = n \Rightarrow \log n = 2^a \Rightarrow \log\log n = a$$

پس خط سوم $O(n\log\log n)$ بار اجرا می‌شود

پس مورد ۱ هم ثابت می‌شود

۴) الف

اثبات درستی: این الگوریتم مرتب سازی بسیار شبیه به insertion-sort است با این تفاوت که از خانه start شروع کرده، مرتب می کند و وقتی به انتهای آرایه می رسد از ابتدای آرایه شروع می کند. یعنی به جای اینکه از خانه ۱ تا n-1 از آرایه n عنصری مرتب کند، از start+1 تا n-1 و سپس از ۰ تا start-1 مرتب می کند.

loop-invariant: قبل از شروع iteration ام، زیر آرایه A[start ... i+n-1 mod n] به صورت دایره‌ای مرتب شده است.

initialization: قبل از شروع اولین itrate زیر آرایه A[start...start+1+n-1 mod n] که همان A[start] است مرتب شده است.

maintenance: وقتی به whileloop می رسیم آرایه A[start... i+n-1 mod n] مرتب شده است. مادامی که A[j] از key بزرگتر است به جلو shift داده می شود تا جای مناسب برای key پیدا شود و نهایتاً key در جای خود قرار می گیرد. اگر shift دادن و itrate کردن به صورت دایره‌ای است. بعد از این iteration زیر آرایه A[start ... i+n mod n] به صورت دایره‌ای مرتب شده است.

Termination: داده‌ها به همین صورت sort شده تا i=start-1 شود پس آرایه از start تا start-1 مرتب می شود و در مرحله بعدی i=start الگوریتم تمام می شود.

	Cost	times
for(i=(start+1)mod n ; i!=start ; i=i+1 mod n	c_1	n
key = A[i]	c_2	$n-1$
j = (i+n-1) mod n	c_3	$n-1$
while(j!=(start+n-1) mod n and A[j] > key	c_4	$\sum_{start+1}^{n-1} t_i + \sum_{0}^{start} t_i$
A[(j+1)mod n] = A[j]	c_5	$\sum_{start+1}^{n-1} t_i - 1 + \sum_{0}^{start} t_i - 1$
j = (j+n-1) mod n	c_6	$\sum_{start-1}^{n-1} t_i - 1 + \sum_{0}^{start} t_i - 1$
A[(j+1) mod n] = key	c_7	$n-1$

$$T(n) = c_1 n + c_2(n-1) + c_3(n-1) + c_4\left(\sum_{start+1}^{n-1} t_i + \sum_{0}^{start} t_i\right)$$

$$+ c_5\left(\sum_{start+1}^{n-1} t_i - 1 \quad \sum_{0}^{start} t_i - 1\right) + c_6\left(\sum_{start+1}^{n-1} t_i - 1 + \sum_{0}^{start} t_i - 1\right) + c_7(n-1)$$

در بهترین حالت (Sorted A) $t_i = 1$ می‌شود با جایگذاری در بالا <= $T(n) = O(n)$

در بدترین حالت (reversed sorted A) $t_i = j$ " " <= $T_n = O(n^2)$

ب) اگر بخواهیم اعداد ۱ تا n را در آرایه مرتب کنیم بیشترین حالت می‌شود:

A[start] = 1

A[start+1] = 2

⋮

A[n-1] = n - start

→ A[0] = n - start + 1

⋮

A[start - 1] = n