آرین قزوینی - شماره دانشجویی : ۹۹۳۱۰۴۵

سوال 1

الف)

| code | cost | times |
|---|---|---|
| `int j=0;` | | |
| `for (int i=0; i<n ; i++){` | C1 | $n+1$ |
| `if(A[i] != B[j]) {` | C2 | $n$ |
| `int K= j;` | C3 | $n$ |
| `while(A[i] != B[K]) {` | C4 | $\sum_{z=1}^{n}(z+1)$ |
| `K++;` | C5 | $\sum_{z=1}^{n}(z)$ |
| `}` | | |
| `swap(B[j], B[K])` | C6 | $n$ |
| `}` | | |
| `j++` | C7 | $n$ |
| `}` | | |

$$T(n) = C1(n+1) + (C2+C3+C6+C7)n + C4\underbrace{\sum_{z=1}^{n}(z+1)}_{\frac{(n+3)(n)}{2}} + C5\Big(\underbrace{\sum_{z=1}^{n} z}_{\frac{(n+1)n}{2}}\Big)$$

$$\Rightarrow T(n) = n\Big(C1+C2+C3+C6+C7+\frac{C5}{2}+\frac{3}{2}C4\Big) + n^2\Big(\frac{C4+C5}{2}\Big) + C1$$

$$\Rightarrow T(n) = O(n^2) \quad (\text{worst case})$$

best case: $T(n) = O(n)$

(وارد حلقه while نمی‌شود)

ب) این الگوریتم اولین اعضای از آرایه B که در آرایه A هم وجود داشته باشند را با اولین نظیرشان

در آرایه A یکسان می‌کند برای مثال :

$A=[1,2,3]$, $B=[3,1,2]$ $\xrightarrow{\text{اجرای الگوریتم}}$ $A=[1,2,3]$, $B=[1,2,3]$

## سوال 2

$X$ متغیر تصادفی باشد که تعداد دفعات چاپ شدن Hello world را نشان می دهد.

$$X_i = I\left\{ \text{در بار } i \text{ ام اجرای حلقه عدد } i \text{ در random تولید شود} \right\} = \begin{cases} 1 & \text{اگر عدد } i \text{ تولید شود} \\ 0 & \text{اگر عدد } i \text{ تولید نشود} \end{cases}$$

$$E[X_i] = \frac{1}{i}$$

(expected running time ، $E$)

$$E[X] = E\left[\sum_{i=1}^{n-1} X_i\right]$$

$$\Rightarrow E[X] = \sum_{i=1}^{n-1} E[X_i]$$

$$\Rightarrow E[X] = \sum_{i=1}^{n-1} \frac{1}{i} \quad (\text{سری هارمونیک})$$

$$\sum_{i=1}^{n-1} \frac{1}{i} \leq \underbrace{\int_1^n \frac{1}{x}\,dx}_{\ln(n)} \leq \lg(n)$$

بنابراین تعداد دفعات چاپ Hello World از $O(\log n)$ است

سوال 3

الف)

| | cost | times |
|---|---|---|
| $i \leftarrow 0$ | $c_1$ | 1 |
| while $i<n$ | $c_2$ | $n+1$ |
| if (Key == T[i]) | $c_3$ | $n$ |
| return i | $c_4$ | 1 |
| else | $c_5$ | $n-1$ |
| $i \leftarrow i+1$ | $c_6$ | $n-1$ |

(در بدترین حالت، آخرین عضو آرایه با Key برابر است)

$$T(n) = c_1 + c_2(n+1) + c_3(n) + c_4 + (c_5+c_6)(n-1)$$

$$= (c_2+c_3+c_5+c_6)n + c_1 + c_4 - c_5 - c_6$$

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

ب و ج) از binary search استفاده می‌کنیم با شروع از i-index ، اولین خانه که بیشترین عضو است)

```
startPoint ← 0
endPoint ← arr.length-1
checkNum ← 0
while( startPoint <= endPoint ){
    checkNum ← startPoint + (endPoint - startPoint) /2
    if( Key == arr[(checkNum + i-index) % arr.length])
        return (checkNum + i-index) % arr.length
    else if (Key < arr[(checkNum + i-index) % arr.length])
        endPoint ← checkNum -1
    else
        startPoint ← checkNum+1
}
```

در بدترین حالت بازه جستجو برای مقایسه کردن آنقدر نصف می‌شود تا به یک عدد برسد. با هر مقایسه

طول بازه نصف می‌شود بنابراین در بدترین حالت داریم $\frac{n}{2^k}=1 \Rightarrow k=\lg n$ است

و الگوریتم از $O(\lg n)$ می‌باشد

سوال 4

الف)

| code | cost | times |
| --- | --- | --- |
| for $i \leftarrow n$ down to 1 do | $C_1$ | $n+1$ |
| for $j \leftarrow 1$ to $i$ do | $C_2$ | $\sum_{i=1}^{n} i+1$ |
| if $A[j] > A[j+1]$: | $C_3$ | $\sum_{i=1}^{n} i$ |
| swap$(A[j], A[j+1])$ | $C_4$ | $\sum_{i=1}^{n} i$ |
| end if | | |

$\boxed{\text{بدترین حالت:}}$ $$C_1(n+1) + C_2\Big(\underbrace{\sum_{i=1}^{n}(i+1)}_{\left(\frac{(n+2)n}{2}\right)}\Big) + (C_3+C_4)\Big(\underbrace{\sum_{i=1}^{n} i}_{\frac{n(n+1)}{2}}\Big)$$

واضح است در این عبارت یک جمله درجه ۲ می باشد پس پیچیدگی زمانی آن از $O(n^2)$ می باشد

$\boxed{\text{بهترین حالت}}$ : بهترین حالت زمانی رخ می دهد که آرایه ما مرتب شده باشد در الگوریتم داده شده

وقتی آرایه مرتب شده است تنها خط swap اجرا نمی شود. هر حلقه هر بار وارد حلقه دوم می شود

و همچنین شرط if نیز هر بار چک می شود بنابراین می توان گفت:

$$C_1(n+1) + C_2\Big(\underbrace{\sum_{i=1}^{n}(i+1)}_{\frac{(n+2)n}{2}}\Big) + C_3\underbrace{\sum_{i=1}^{n} i}_{\frac{n(n+1)}{2}}$$

همانند بدترین حالت، برای بهترین حالت هم یک تابع درجه ۲ بدست آوردیم و در نتیجه پیچیدگی زمانی آن از $O(n^2)$ است

$\boxed{\text{حالت متوسط}}$ : (معمولاً به عملاً برابر زمان اجرای بدترین حالت است) در این الگوریتم هم چون بدترین

و هم بهترین حالت از $O(n^2)$ است پس حالت متوسط هم از $O(n^2)$ می باشد.

ادامه سوال 4 ب)

```
for i ← n down to 1 do
    flag ← 0
    for j ← 1 to i do
        if A[j] > A[j+1]:
            flag ← 1
            swap(A[j], A[j+1])
        endif
    if (flag == 1):
        break
```

با اضافه کردن متغیر flag در صورتی که هیچ گاه عملیات swap صورت نپذیرد نشان دهنده

این است که آرایه سورت شده بوده و حلقه بیرونی break می‌شود

در این شرایط (بهترین حالت) پیچیدگی زمانی از مرتبه $O(n)$ می‌شود.

(سوال 5)

a) تعریف $f(n) = O(g(n)) = \{ f(n) : \exists c > 0, n_0 > 0 \mid \forall n \geq n_0$ که $0 \leq f(n) \leq c\, g(n) \}$

$n^{(\lg n)^{-1}} \longrightarrow \lg n = K \Rightarrow 2^K = n \xrightarrow[K \text{ به ازای}]{\text{طرفین به توان}}$

$(2^K)^{\frac{1}{K}} = n^{\frac{1}{K}} \Rightarrow n^{\frac{1}{\lg n}} = 2$

بنابراین یک c مطمئناً وجود دارد $\Rightarrow 0 \leq n^{\frac{1}{\lg n}} = 2 \leq c \cdot 1$

پس $n^{\frac{1}{\lg n}} = O(1)$

تعریف $f(n) = \Omega(g(n)) = \{ f(n) : \exists c > 0, n_0 > 0 \mid \forall n \geq n_0 : 0 \leq c\, g(n) \leq f(n) \}$

به همین ترتیب c دلخواه وجود دارد پس $0 \leq c \cdot 1 \leq n^{\frac{1}{\lg n}}$

$n^{\frac{1}{\lg n}} = \Omega(1)$ در نتیجه ثابت می‌شود $n^{\frac{1}{\lg n}} = \Theta(1)$

b) اگر $f(n) = \omega(g(n))$ آنگاه: $\lim_{n \to \infty} \frac{f(n)}{g(n)} = \infty$

$n! \xrightarrow{\text{تقریب استرلینگ}} \sqrt{2\pi n} \left(\frac{n}{e}\right)^n \left(1 + \Theta\left(\frac{1}{n}\right)\right)$

$\lim_{n \to \infty} \frac{n!}{2^n} = \sqrt{2\pi n} \left(\frac{n}{2e}\right)^n \left(1 + \Theta\left(\frac{1}{n}\right)\right)$

اگر فرض کنیم $n_0 = 10$ و $\forall n \geq 10$ آنگاه $1 < \left(\frac{n}{2e}\right)$ پس $\lim_{n \to \infty} \left(\frac{n}{2e}\right)^n = \infty$

$\infty$ می‌شود و در نتیجه حاصل حد نیز بی‌نهایت است

c) $n! = n \times n-1 \times n-2 \times n-3 \times \cdots 1$ (شامل n عبارت)

$n^2 = \underbrace{n \times n \times n \times \cdots n}_{n \text{ تا}}$

می‌دانیم که $n > n-1, n > n-2, n > n-3, \cdots n > 1$

بنابراین به ازای هر c مثبت و یک ثابت $n_0 > 0$ به ازای هر $n \geq n_0$

$0 \leq n! \leq c n^n$

سوال b

a) غلط ـ مثال نقض: $f(n)=n$, $g(n)=n^2$ $\left\{ \begin{array}{l} n=O(n^2)\ ✓ \\ n^2 \neq O(n) \end{array} \right.$

b) غلط ـ مثال نقض: $f(n)=2n$, $g(n)=n$ $\left\{ \begin{array}{l} 2n = O(n)\ ✓ \\ 2^{2n} \neq O(2^n) \end{array} \right.$

اثبات: $2^{2n} \leqslant c2^n \Rightarrow 2^n \leqslant c$

از آنجایی که $c$ یک ثابت است نمی‌توان $c$ ای را پیدا کرد که از هر $2^n$ $(\forall n \geqslant n_0)$ بزرگتر یا مساوی باشد.

c) غلط ـ مثال نقض: $f(n)=\frac{1}{n}$ $\quad \frac{1}{n} \neq O(\frac{1}{n^2})$

$\frac{1}{n} \leqslant \frac{c_1}{n^2} \Rightarrow 1 \leqslant \frac{c}{n} \Rightarrow n \leqslant c$

امکان پذیر نیست زیرا $c$ یک ثابت است ( $\exists n_0 \geqslant 0$ , $\forall n \geqslant n_0$ )

d) درست ـ $f(n) = O(g(n)) \longrightarrow 0 \leqslant f(n) \leqslant cg(n)$

$\Rightarrow \underbrace{0 \leqslant \frac{f(n)}{c} \leqslant g(n)}$

$g(n) = \Omega(f(n))$

e) غلط ـ مثال نقض $\rightarrow f(n)=2^n$, $0 \leqslant 2^n \leqslant c \cdot 2^{\frac{n}{2}}$

(شبیه حالت b)

$0 \leqslant 2^{\frac{n}{2}} \cdot 2^{\frac{n}{2}} \leqslant c \cdot 2^{\frac{n}{2}}$

$0 \leqslant 2^{\frac{n}{2}} \leqslant c$ ← هیچ $c$ ای یافت نمی‌شود که صدق کند زیرا $c$ ثابت است

f ـ درست $f(n)+g(n) = \Theta(f(n)) \leftarrow g(n)=o(f(n))$ : صحت عبارت را می‌توان به این شکل نشان داد

$g(n) = o(f(n)) \Rightarrow [\, 0 \leqslant g(n) < c\,f(n) \quad \forall c > 0 , \ \exists n_0 > 0 , \ \forall n \geqslant n_0 \,]$

چون $g(n) \geqslant 0$ $\Rightarrow f(n)+g(n) \geqslant f(n) \Rightarrow \boxed{f(n)+g(n) = \Omega(f(n))}$

$0 \leqslant g(n) < cf(n) \xrightarrow{+f(n)} 0 \leqslant g(n)+f(n) < \underbrace{(c+1)}_{c'}f(n) \Rightarrow \boxed{f(n)+g(n) = O(f(n))}$

( $f(n)$ مطمئناً نامنفی است )

$\Rightarrow f(n)+g(n) = \Theta(f(n))$

سوال 7

| A | B | O | o | Ω | ω | Θ |
|---|---|---|---|---|---|---|
| $n^2$ | $n^3$ | yes | yes | no | no | no |
| $\lg^k n$ | $n^\epsilon$ | yes | yes | no | no | no |
| $n^k$ | $c^n$ | yes | yes | no | no | no |
| $2^n$ | $2^{n/2}$ | no | no | yes | yes | no |
| $2^{2\lg n}$ | $n^2$ | yes | no | yes | no | yes |
| $n!$ | $n \cdot 2^n$ | no | no | yes | yes | no |
| $n^{\lg(\lg(n))}$ | $(\lg(n))^{\lg(n)}$ | yes | no | yes | no | yes |
| $n^{\lg n}$ | $(\lg n)^2$ | no | no | yes | yes | no |
| $\frac{n^2}{\log n}$ | $n(\lg n)^2$ | no | no | yes | yes | no |
| $n^{\frac{1}{n}}$ | $\sqrt{2}^{\lg n}$ | yes | yes | no | no | no |