| | |
|---|---|
| عنوان درس: طراحی الگوریتم | دانشگاه: پیام نور تهران شمال |
| استاد: آقای علی رضوی | پاسخ سوالات فرد نیم‌سال اول ۹۴—۱۳۹۳ |
| نام دانشجو: صغری عزیزی | شماره دانشجویی: ۹۷۰۱۶۷۹۸۲ |

پاسخ سوال ۱: گزینه (ب) درست است. زیرا: می‌دانیم که $T_1(n) \in o(n^2)$ بنابراین $C_1$ و $n_1$ وجود دارد که برای: $\forall\, n \geq n_1 \quad T_1(n) \leq C_1(n^2)$

و همچنین $T_2(n) \in o(n^2)$ بنابراین $C_2$ و $n_2$ وجود دارد که برای:

$$\forall\, n \geq n_1 \quad T_2(n) \leq C_2(n \log n)$$

$$\Longrightarrow T_1(n) + T_2(n) \leq C_1(n^2) + C_2(n \log n) \Longrightarrow T_1(n) + T_2(n) \in O(n^2)$$

پاسخ سوال ۳: گزینه (د)، $T(n) = T(n-1) + T(n-2) + 1$ درست است زیرا: در مرحله فراخوانی، مقدار متغیرها در پشته $push$ می‌شوند. بنابراین برای $n = 3$، اول $fact(3)$ فراخوانی می‌شود. بازای $n = 3$ تابع دوباره فراخوانی می‌شود بنابراین مقادیر اول در پشته سیستم ذخیره می‌شود و عمل فراخوانی دوباره ادامه می‌یابد تا اینکه $n = 1$ شود. در اینصورت برای محاسبه عملیات لازم در توابع فراخوانی شده، مقدار یک بازگشت داده می‌شود. بازای هر مرحله بازگشت یک عمل حذف از بالای پشته انجام می‌گیرد و در عین حال عملیات لازم برای بازگشت بعدی انجام می‌گیرد تا زمانی که پشته به خانه ۱ برسد عمل بازگشت ادامه می‌یابد.

پاسخ سوال ۵: گزینه (ج) درست است زیرا:

$$if(tree == NULL)$$
$$\quad return o;$$
$$\quad else\, \{$$
$$\quad\quad inli = f(tree \to left);$$
$$\quad\quad inli = f(tree \to right); \implies \text{خروجی } = ۴$$
$$\quad\quad if(i > j)$$
$$\quad\quad\quad return\, ۱ + i;$$
$$\quad\quad else$$
$$\quad\quad\quad return\, ۱ + j;$$

پاسخ سوال ۷: گزینه (ج) درست است زیرا:

۱۱ ۱۲ ۱۸ ۲۰ ۲۱ $\boxed{۲۳}$ ۲۷ $\boxed{۴۰}$ ۷۵ ۸۰ ۸۵ (۱)

عنصر میانی عدد

نصر میانی عدد ۲۳ هست که با ۴۰ برابر نیست چون ۲۳ > ۴۰ پس باید به زیر آرایه سمت چپ

۲۷ $\boxed{۴۰}$ $\boxed{۷۵}$ ۸۰ ۸۵ (۲)

(۳) ۴۰ < ۷۵ $\Longrightarrow$ ۲۷ < ۴۰۵ (۴)

پاسخ سوال ۹: گزینه (د) درست است زیرا: در مرحله اول مرتب سازی سریع عناصر کوچکتر از عنصر محور را در سمت چپ عناصر بزرگتر را در سمت راست لیست قرار می‌دهیم:

| | | | | | | |
|---|---|---|---|---|---|---|
| ۱۰ | ۱۵ | $\boxed{۹}$ | ٦ | ۷ | ۸ | ۳ |

پاسخ سوال ۱۱: گزینه (ج) درست است زیرا: اگر $n \leq ۲$ باشد، $\max, \min$ را پیدا می‌کنیم و برای $n > ۲$، مسأله را به دو زیرمسأله با طول‌های $\frac{n}{۲}$ تقسیم می‌کنیم و آنها را حل می‌کنیم نتایج حاصل از حل زیر مسأله‌ها را برای حل مسأله اصلی باهم ترکیب می‌کنیم با $\frac{۳n}{۲} - ۲$ مقایسه بین اعداد ذخیره شده در این خانه‌ها بدست خواهد آمد.

پاسخ سوال ۱۳: گزینه (ب) درست است زیرا: طبق الگوریتم، ابتدا رأس $Y = \{v_۰\}$ انتخاب می‌شود و $F = \phi$ خواهد بود.

مرحله اول تمام رئوس مجاور $v_۰$ را پیدا می‌کنیم. بنابراین خواهیم داشت:

$\{v_1, v_2\}, e_{01} = 1, e_{02} = v$، به‌وضوح رئوس $v_1$ رأس انتهای خوهد بود و یال $e_{01}$ به $f$ اضافه می‌شود و داریم: $f = \{e_{01}\}$.

مرحله دوم: حال رئوس مجاور به $Y$ را انتخاب می‌کنیم، نزدیک‌ترین رأس که $V_2$ بوده انتخاب می‌کنیم.

مرحله سوم: نزدیک‌ترین رأس به $Y$ رأس $V_5$ بوده.

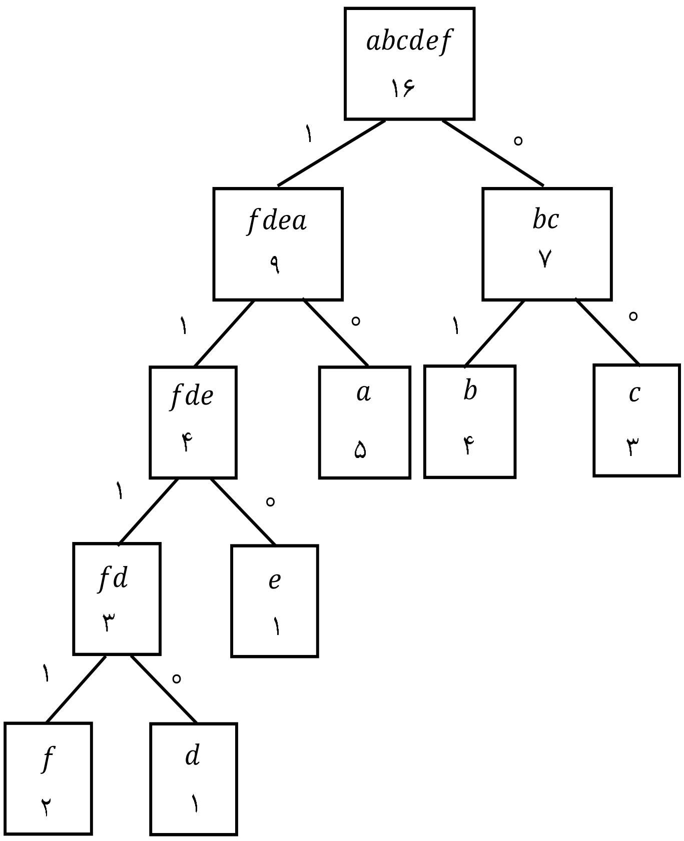
مرحله چهارم: نزدیک‌ترین رأس به $Y$ رأس $V_4$ بوده.

مرحله پنجم: نزدیک‌ترین رأس به $Y$ رأس $V_5$ بوده.

مرحله ششم: چون $v = Y$ می‌باشد، بنابراین گراف حاصل به‌عنوان درخت پوشای مینیمم با مقدار هزینه‌ای می‌باشد.

پاسخ سوال ۱۵: گزینه (د) درست است زیرا: $abcabbaccaabdffc$ (از هر صفر یا یک در هر سطح رد شویم یک بیت می‌شود.)

$(5 \times 2) + (4 \times 2) + (3 \times 2) + (1 \times 4) + (1 \times 3) + (2 \times 4) = 38$ بیت



پاسخ سوال ۱۷: گزینه (د) درست است زیرا: $A_{۱۳*۵}\ ,\ B_{۵*۸۹}\ ,\ C_{۸۹*۳}\ ,\ D_{۳*۳۴}$

باید سعی کنیم اعداد بزرگی مانند ۸۹ و ۳۴ تنها یک‌بار استفاده شوند، چون ۳۴ از ابعاد وسطی است می‌توان با یک ضرب آن‌را از بین برد.

$$M = A \times B \times C \times D \longrightarrow m = (A((BC)D))$$

در داخلی‌ترین ضرب $B_{۵*۸۹} * C_{۸۹*۳}$ تعداد ضرب‌ها برابر است با ۳ × ۸۹ × ۵. حالا ماتریس حاصل از ضرب $BC$ یک ماتریس ۳ × ۵ است و وقتی در $D$ ضرب شود، تعداد ضرب‌های آن برابر است با ۵ × ۳ × ۳۴ ماتریس حاصل تا اینجای کار ۵ × ۳۴ خواهد بود که برای ضرب $A$ در آن به ۱۳ × ۵ × ۳۴ ضرب نیاز است، پس تعداد کل ضرب‌ها برابر است با: ۲۸۵۶

پاسخ سوال ۱۹: گزینه (الف ) درست است زیرا: با $n$ عنصر متمایز می‌توانیم $\frac{۱}{n۱}\begin{pmatrix} ۲n \\ n \end{pmatrix}$ درخت جست‌وجوی دو دویی بسازیم که همان عدد کاتالان است.

پاسخ سوال ۲۱: گزینه (ج) درست است زیرا: در حالت کلی دو وزیر $a_{ij}$ و $a_{kl}$ مورد حمله یکدیگر خواهند بود اگر یکی از سه حالت زیر رخ دهد:

۱) $j = ۱$ دو وزیر در یک ستون باشند.

۲) $i - j = k - ۱$ یا $i + j = k + ۱$ برای نوشتن برنامه $n$ وزیر لبتدا تابع $promising$ را طراحی می‌کنیم که وظیفه چک کردن سه شرط گفته‌شده را دارد. اگر هیچ‌کدام از سه شرط فوق برقرار نباشد خروجی الگوریتم یک وگرنه صفر خواهد بود.

پاسخ سوال ۲۳: گزینه (د) درست است زیرا: از بین مهره‌های موجود در شکل که توسعه نیافته‌اند گره‌ای را که دارای کمترین ارزش است ادامه می‌دهیم.

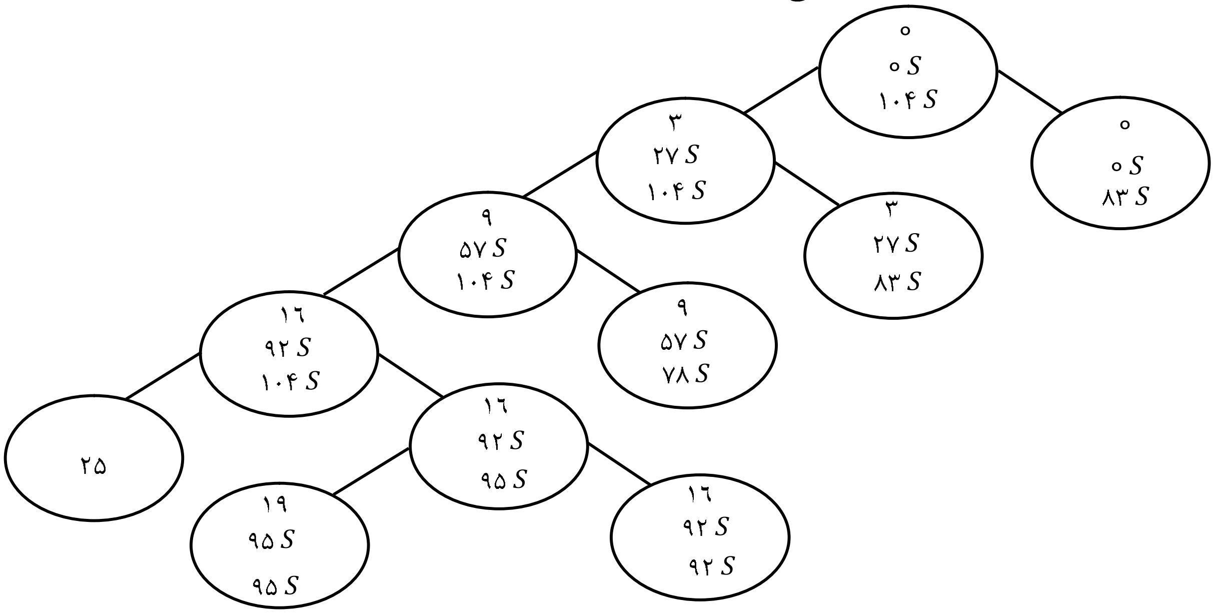
پاسخ سوال ۲۵: گزینه (الف) درست است زیرا: تعریف $NP$ به‌درستی ذکر شده است، اما در گزینه‌های دیگر به‌درستی ذکر نشده است چراکه مسائلی که نوشتن یک الگوریتم کارآمد برای آنها غیرممکن است، مسائل رام‌نشدنی می‌گویند. مسائلی که الگوریتم کارا برای آنها ابلاغ شده است ولی غیر ممکن بودن آنها نیز هنوز به اثبات نرسیده را مسائل $NP$ کامل می‌گویند.

# تحلیل سوالات فرد نیم‌سال اول ۹۴—۹۳ (سوالات تشریحی)

۱. $\{e_{۱۲}, e_{۳۶}, e_{۴۶}, e_{۲۶}, e_{۳۵}\}$

۳.
$$\begin{bmatrix} ۰ & ۰/۷ & ۱/۱ & ۱/۴ \\ & ۰ & ۰/۲ & ۰/۴ \\ & & ۰ & ۰/۱ \\ & & & ۰ \end{bmatrix}$$

۵. سود بیشینه برابر با ۹۵ می‌باشد.




## سوالات زوج

۲. (ج) فصل اول

ترتیب مرتبه زمانی از کوچک‌تر به بزرگ‌تر: ($a$ عدد ثابت است.)

$$o(۱) < o(\sqrt{\log n}) < o(\sqrt{\log_۲ n}) < o(n) < o(n\log n) < o(n^۲) < o(n^۳) < o(n^a)$$

$$< o\left(\left(\frac{۳}{۲}\right)^n\right) < o(n^۲) < o(a^n) < o(n!) < o(n^n)$$

در چندجمله‌ای داده‌شده جمله $۵n\log n$ دارای بالاترین مرتبه زمانی است، پس پیچیدگی زمانی برابر می‌شود با $o(n\log n)$.

۴. (ب) فصل اول

تعداد ضرب‌های حاصل‌ضرب دو ماتریس $n \times n$ برابر می‌شود با $n \times n \times n$، لذا مرتبه زمانی آن برابر با $n^3$ می‌شود.

٦. (الف) فصل دوم

تابع سری فیبوناچی به‌صورت زیر است:

$$F(n) := \begin{cases} 0 & if \quad n = 0 \\ 1 & if \quad n = 1 \\ F(n-1) + F(n-2) & if \quad n > 1 \end{cases}$$

$$fib(4) = fib(3) + fib(2)$$
$$fib(3) = fib(2) + fib(1)$$
$$fib(2) = fib(1) + fib(0)$$
$$fib(1) = 1$$
$$fib(0) = 0$$
$$fib(2) = fib(1) + fib(0) = 1 + 0 = 1$$
$$fib(3) = fib(2) + fib(1) = 1 + 1 = 2$$
$$fib(4) = fib(3) + fib(2) = 2 + 1 = 3$$
$$fib(4) + fib(3) = 3 + 2 = 5$$

٨. (ب) فصل سوم

تابع بازگشتی به‌صورت $T(n)$ بیان می‌گردد. در گزینه‌های ۱، ۳ و ۴، پرانتز $T$ براساس $n$ بیان نشده است، لذا صحیح نیستند.

١٠. (ج) فصل چهارم

| | بهترین حالت | حالت میانگین | بدترین حالت |
|---|---|---|---|
| جستجوی موفق | $\Omega(1)$ | $\theta(\log n)$ | $o(\log n)$ |
| جستجوی ناموفق | $\Omega(\log n)$ | $\theta(\log n)$ | $\theta(\log n)$ |

١٢. (ب) فصل چهارم

در ماتریس استراسن:

$$\begin{cases} P = (A_{۱۱} + A_{۲۲})(B_{۱۱} + B_{۲۲}) \\ Q = (A_{۲۱} + A_{۲۲})B_{۱۱} \\ R = A_{۱۱}(B_{۱۲} - B_{۲۲}) \\ S = A_{۲۲}(B_{۲۱} - B_{۱۱}) \\ T = (A_{۱۱} + A_{۱۲})B_{۲۲} \\ U = (A_{۲۱} - A_{۱۱})(B_{۱۱} + B_{۱۲}) \\ V = (A_{۱۲} - A_{۲۲})(B_{۲۱} + B_{۲۲}) \end{cases}$$

۱۴. (ب) فصل چهارم

اعداد صحیح بزرگ نصف می‌شوند و اگر $n$ رقم داشته باشند، از وسط به دو عدد تبدیل می‌شوند، مثلاً $U$ به $x$ و $y$ تبدیل می‌شود و $m = \left|\frac{n}{۲}\right|$.

۱۶. (ب) فصل پنجم

پیچیدگی زمانی الگوریتم پریم $o(n^۲)$ می‌باشد.

۲۰. (د) فصل ششم

$D^۰$ همان ماتریس مجاورت گراف می‌باشد.

$$D_۰ = \begin{bmatrix} ۰ & ۱ & \infty & ۱ & ۵ \\ ۹ & ۰ & ۳ & ۲ & \infty \\ \infty & \infty & ۰ & ۴ & \infty \\ \infty & \infty & ۲ & ۰ & ۳ \\ ۳ & \infty & \infty & \infty & ۰ \end{bmatrix}$$

برای $D^۱$ داریم: ابتدا سطر و ستون ۱ را مانند جدول قبل قرار می‌دهیم.

$$D^۱(۲,۲) = ۰ \ , \ D^n(۲,۲) = D^n(۳,۳) = ۰$$

$$D^۱(۲,۳) = \min(D^۰(۲,۱) + D^۰(۱,۳) \ , \ D^۰(۲,۳)) = \min(۹ + \infty, ۳) = ۳$$

منظور از $D^۱(۲,۴)$ این است که کوتاهترین مسیر از گره ۲ به گره ۴ با گره واسطه ۱ کدام است.

$$D^۱(۲,۴) = \min(D^۰(۲,۱) + D^۰(۱,۴) \ , \ D^۰(۲,۴)) = \min(۹ + ۱, ۲) = ۲$$

$$D_۱ = \begin{bmatrix} ۰ & ۱ & \infty & ۱ & ۵ \\ ۹ & ۰ & ۳ & ۲ & ۱۴ \\ \infty & \infty & ۰ & ۴ & \infty \\ \infty & \infty & ۲ & ۰ & ۳ \\ ۳ & \infty & \infty & \infty & ۰ \end{bmatrix}$$

۲۲. (ج) فصل هفتم

در مسئله حاصل جمع زیرمجموعه‌ها می‌بایست تمام زیرمجموعه‌هایی از قطعات را بیابیم به‌طوری‌که مجموع اوزان آنها به اندازه وزن کوله یعنی ۲۱ باشد.

تشریحی :

۲– فصل چهارم

ابتدا $j$ در عناصر کوچکتر از ۱۷ متوقف شده و سپس $i$ در عنصر بزرگتر از ۱٦ و پس از آن عناصر تعویض می‌شوند. اگر $j$ کل آرایه را پیمایش کند و عنصر کوچکتر نباشد محل $i$ با عنصر اول جابجا می‌شود.

| | $i$ | $j$ | | | | | |
|---|---|---|---|---|---|---|---|
| ۱۷ | ۲۰ | ۱۰ | ۲۵ | ۱۱ | ۸ | ۱۸ | ۲۳ |

$\Rightarrow$

| | | $i$ | | $j$ | | | |
|---|---|---|---|---|---|---|---|
| ۱۷ | ۱۰ | ۲۰ | ۲۵ | ۱۱ | ۸ | ۱۸ | ۲۳ |

| | | | $i$ | | $j$ | | |
|---|---|---|---|---|---|---|---|
| ۱۷ | ۱۰ | ۱۱ | ۲۵ | ۲۰ | ۸ | ۱۸ | ۲۳ |

$\Rightarrow$

| $j$ | | | $i$ | | | | |
|---|---|---|---|---|---|---|---|
| ۱۷ | ۱۰ | ۱۱ | ۸ | ۲۰ | ۲۵ | ۱۸ | ۲۳ |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| ۸ | ۱۰ | ۱۱ | ۱۷ | ۲۰ | ۲۵ | ۱۸ | ۲۳ |

در این حالت عنصر ۱۷ در محل خود در آرایه مرتب شده قرار می‌گیرد در مرحله بعد، همین اعمال برای آرایه سمت چپ و راست ۱۷ تکرار خواهد شد. برای آرایه سمت راست ۱۷:

$i$ در عناصر بزرگتر از ۲ و $j$ در عناصر کوچکتر از ۲۰ قرار گرفته جابجا می‌شوند.

| | $i$ | $j$ | |
|---|---|---|---|
| ۲۰ | ۲۵ | ۱۸ | ۲۳ |

$\Rightarrow$

| $i$ | $j$ | | |
|---|---|---|---|
| ۲۰ | ۱۸ | ۲۵ | ۲۳ |

| | | | |
|---|---|---|---|
| ۱۸ | ۲۰ | ۲۵ | ۲۳ |

عدد ۲۰ در محل خود قرار می‌گیرد و این اعمال برای آرایه سمت راست و چپ آن تکرار خواهد شد.

۴– در الگوریتم پریم، در هر بار رأس جدیدی به مجموعه اضافه می‌گردد با این شرط که این رأس با وزن یال کمتری به مجموعه وصل شود. ابتدا $V_0$ را در نظر می‌گیریم از مجموعه رئوسی که به آن متصل است راس $V_1$ با کمترین با کمترین وزن انتخاب می‌شود، سپس از مجموعه رئوسی دیگری که به رئوس $V_0$ و $V_1$ متصلند $V_2$ با کمترین وزن انتخاب می‌شود راس سوم $V_2$ خواهد بود و به همین ترتیب . . .