گروه بابان
BABAN.IR

بسم الله الرحمن الرحیم

ساختمان داده و طراحی الگوریتم
مهندسی کامپیوتر۱۴۰۲

ابوالفضل گیلک
گروه بابان

C

آزمون ورودی دوره‌های کارشناسی ارشد ناپیوستهٔ داخل - سال ۱۴۰۲

جمهوری اسلامی ایران
وزارت علوم، تحقیقات و فناوری
سازمان سنجش آموزش کشور

«اگر دانشگاه اصلاح شود مملکت اصلاح می‌شود.»
امام خمینی (ره)

عصر جمعه
۱۴۰۱/۱۲/۱۲

مهندسی کامپیوتر (کد ۱۲۷۷)

تعداد سؤال: ۱۱۵

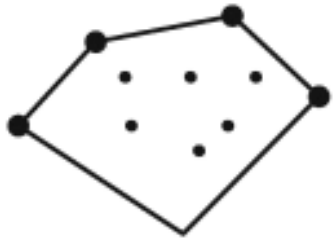
زمان پاسخ‌گویی: ۲۴۰ دقیقه

عنوان مواد امتحانی، تعداد و شمارهٔ سؤالات

ردیف	مواد امتحانی	تعداد سؤال	از شمارهٔ	تا شمارهٔ
۱	زبان عمومی و تخصصی (انگلیسی)	۲۵	۱	۲۵
۲	ریاضیات (ریاضی عمومی (۱و۲)، آمار و احتمال مهندسی، ریاضیات گسسته)	۲۰	۲۶	۴۵
۳	دروس تخصصی۱ (نظریه زبان‌ها و ماشین‌ها، سیگنال‌ها و سیستم‌ها)	۱۰	۴۶	۵۵
۴	دروس تخصصی ۲ (ساختمان داده‌ها، طراحی الگوریتم و هوش مصنوعی)	۲۰	۵۶	۷۵
۵	دروس تخصصی ۳ (مدار منطقی، معماری کامپیوتر و الکترونیک دیجیتال)	۲۰	۷۶	۹۵
۶	دروس تخصصی ۴ (سیستم‌های عامل، شبکه‌های کامپیوتری و پایگاه داده‌ها)	۲۰	۹۶	۱۱۵

دروس تخصصی ۲ (ساختمان داده‌ها، طراحی الگوریتم و هوش مصنوعی):

۵۶- در مسئلهٔ یافتن پوسته محدب، تعدادی نقطه در صفحه داده شده است. هدف پیداکردن کوچکترین (کمترین محیط) چندضلعی محدب است که شامل همه نقاط باشد. بهترین الگوریتم ممکن برای یافتن این چندضلعی، چه مرتبه زمانی خواهدداشت؟



۱) $\theta(n^2)$

۲) $\theta(n^3)$

۳) $\theta(n \log n)$

۴) $\theta(n^2 \log n)$

پاسخ: گزینه (۳)

یافتن پوش محدب convex hull

① یافتن نقطه با کمترین مقدار y. (نقطه‌ی P_1) $\Theta(n)$

② مرتب سازی سایر نقاط برحسب زاویه‌ی قطبی به مرکز P_1. $\Theta(n \log n)$

③ ابتدا P_1 و P_2 در پشته درج شوند.

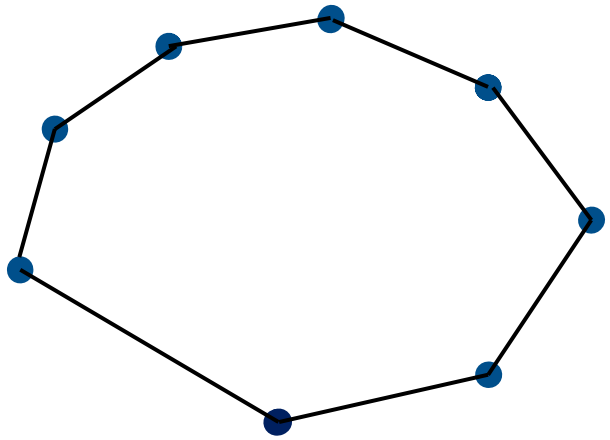
④ برای هر P_i چپ گرد بودن آن نسبت به دو نقطه بالای پشته بررسی شود.
اگر چپ گرد بود فقط درج P_i در پشته انجام شود.
اگر چپ گرد نبود ابتدا pop سپس بررسی مجدد.

هر نقطه دقیقاً یک بار درج و حداکثر یک بار pop می‌شود. $\Theta(2n)$

توضیحات تکمیلی:

حداکثر n نقطه رأسی به دست می‌آید:

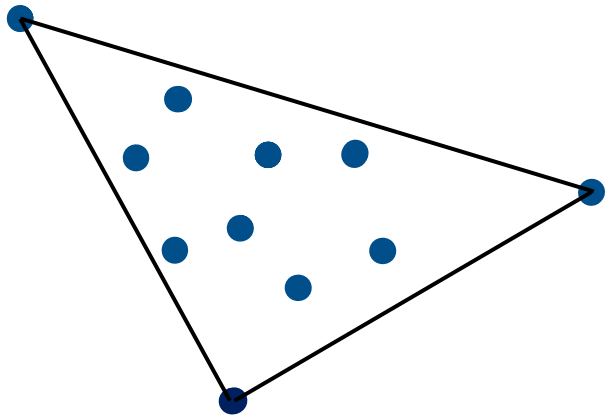
$|\text{Convex hull}| = n$



$|\text{Convex Hull}| = n$

حداقل ۳ نقطه رأسی به دست می‌آید:

$|\text{Convex hull}| = ۳$



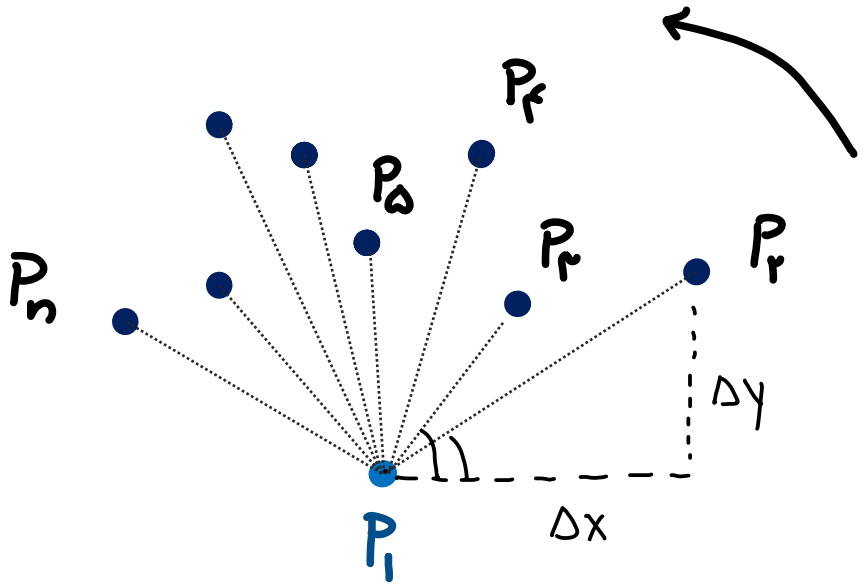
$|\text{Convex Hull}| = ۳$

توضیح مرحله‌ی ۲:

$\Delta x > 0 \qquad \theta = \text{Arctan} \frac{\Delta y}{\Delta x}$

$\Delta x < 0 \qquad \theta = \text{Arctan} \frac{\Delta y}{\Delta x} + \pi$

مرتب‌سازی برحسب θ : $O(n \log n)$



۵۷- مرتبه زمانی الگوریتم زیر کدام است؟

```
for(i = ۱ ; i <= n ; i++)
    for(j = ۱ ; j <= ⌊n/i⌋ ; j++)
        S.O ;
```

۱) $O(n)$

۲) $O(n \log n)$

۳) $O(n^{۲})$

۴) $\theta(n)$

پاسخ: گزینه ۲

در حلقه‌ی میانی متغیر j با گام ۱، از $j=1$ تا $j=\lfloor \frac{n}{i} \rfloor$ می‌آید.

مرتبه آن $\Theta(\frac{n}{i})$ است. در نتیجه:

$$T(n) = \sum_{i=1}^{n} \frac{n}{i} = n \sum_{i=1}^{n} \frac{1}{i} = O(n \log n)$$

۵۸- کدام‌یک از عبارات زیر درست است؟

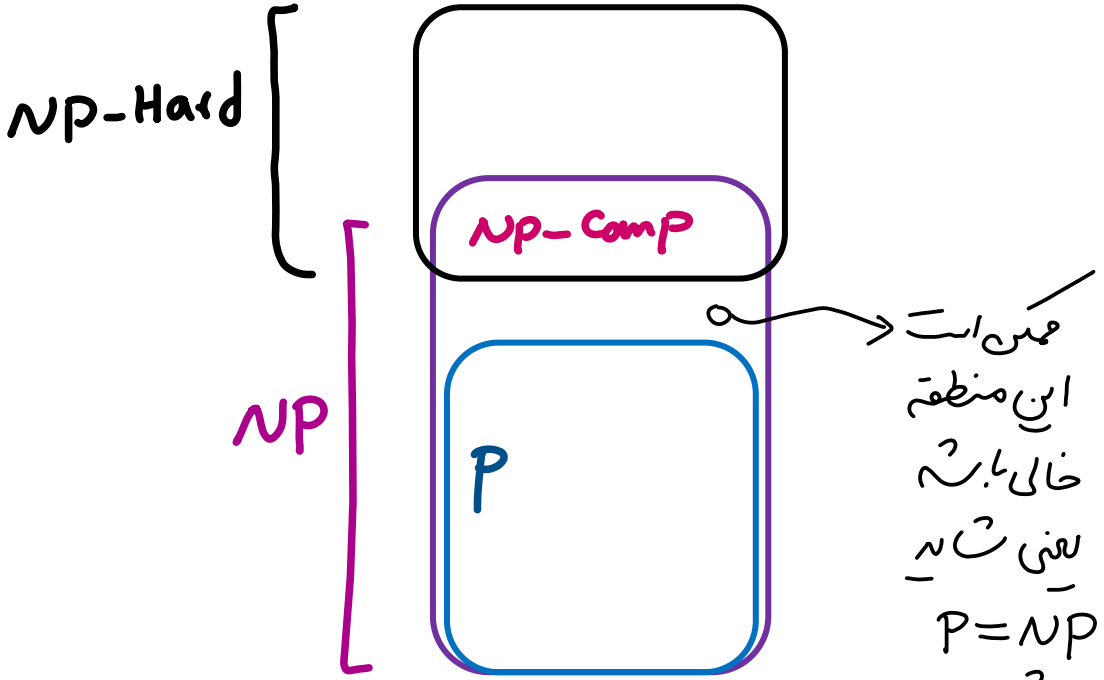
الف- هر مسئله‌ای NP- کامل یک مسئله‌ای NP- سخت است.

ب- هر مسئله‌ای NP- سخت یک مسئله‌ای NP- کامل است.

ج- هر مسئله‌ای P (چندجمله‌ای) یک مسئله‌ای NP هم هست.

۱) الف و ب
۲) الف و ج
۳) ب و ج
۴) الف، ب و ج

پاسخ: گزینه ۲ (خلاصه‌ی همه‌ی اطلاعاتی که لازم دارید این است:)



ممکن است این منطقه خالی باشد یعنی شاید $P=NP$ شود.

(اسلاید بعدی)

$$P \subseteq NP$$

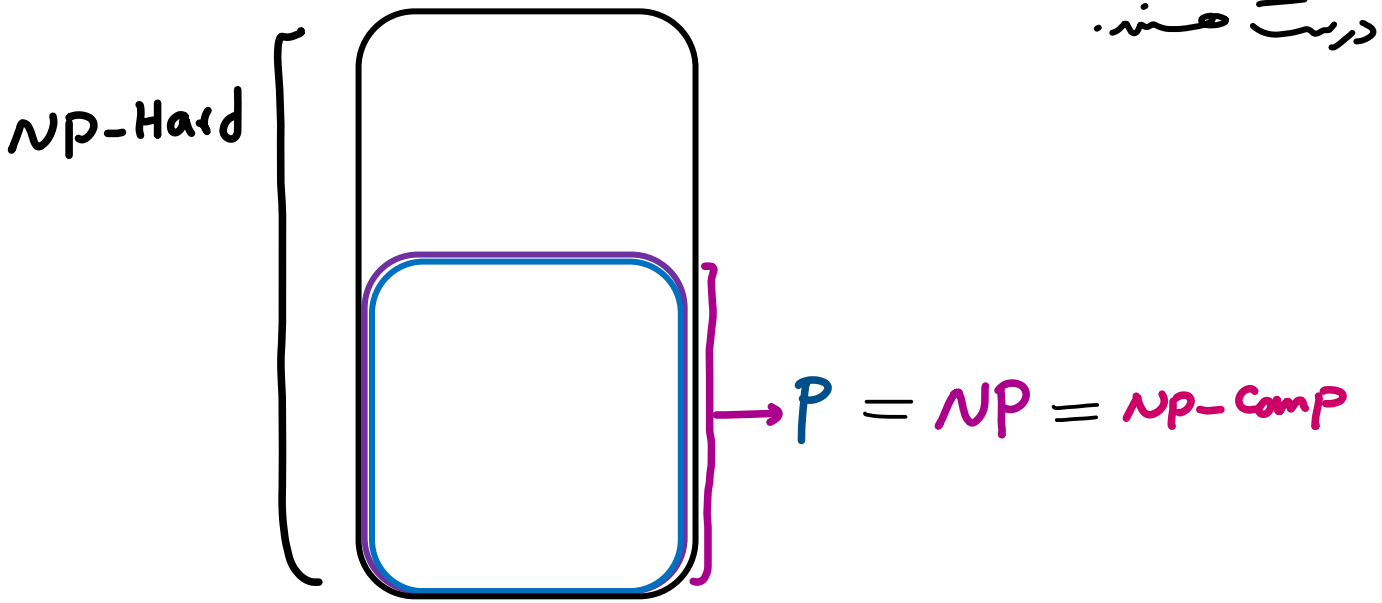
$$NP\text{-}Comp \subseteq NP$$

$$NP\text{-}Comp \subseteq NP\text{-}Hard$$

$$NP\text{-}Comp = NP \cap NP\text{-}Hard$$

اگر $P = NP$ نمود این حالت رخ می‌دهد:

البته روابط صفحه‌ی قبل باز هم درست هستند.



۵۹- **فرض کنید که یک مسئله را به‌صورت بهینه به توان هم با روش تقسیم و حل، هم با روش برنامه‌ریزی پویا و هم با روش حریصانه حل کرد. در این‌صورت از لحاظ پیچیدگی زمانی کدام‌یک ارجحیت دارد؟**

۱) حریصانه

۲) تقسیم و حل

۳) برنامه‌ریزی پویا

۴) به مسئله بستگی دارد.

پاسخ: گزینه (۱)

در تمام مسائل با n ورودی، اگر روش حریصانه نتواند پاسخ بهینه را بدهد قطعاً از نظر زمان اجرا بهتر از روش پویا است زیرا در روش پویا ناچاریم راه‌حل را برای تمام زیرمسئله‌ها هم پیدا کنیم تا به پاسخ مسئله اصلی برسیم. روش تقسیم و حل نیز به دلیل overlap و حل تکراری مسئله‌ها زمان بیشتری نسبت به حریصانه دارد.

۶۰- **کدام‌یک از مسائل زیر در زمان خطی برحسب تعداد رئوس و یال‌های گراف ورودی قابل‌حل نیست؟**

۱) تشخیص همبندی گراف ساده

۲) تشخیص دوبخشی‌بودن گراف ساده

۳) پیداکردن درخت پوشای کمینه گراف وزن‌دار همبند

۴) پیداکردن ترتیب توپولوژیکی رئوس گراف غیرحلقوی جهت‌دار (DAG)

پاسخ : گزینه ۳

اندازه‌ی ورودی $V+E$ است. (تعداد گره‌ها و یال‌ها)

گزینه‌های (۱)، (۲) و (۴) از مسائل شناخته شده‌ای هستند که با کمک پیمایش‌های BFS یا DFS حل می‌شوند (رجوع به جزوه درس حبات ۱۵ و ۱۶ و نکته تست گراف ها). زمان اجرای آنها $O(V+E)$ است.
بنابراین نسبت به اندازه ورودی، خطی هستند.

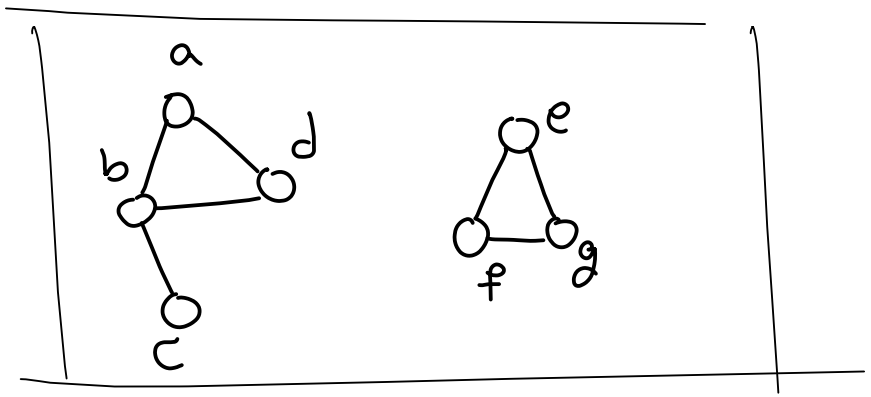
اما درخت MST در گراف وزن دار، در زمان $O(E \log V)$ به دست می‌آید.

(توضیحات تکمیلی در ادامه آمده است)

تشخیص همبندی گراف ساده: می‌توانیم از DFS استفاده کنیم و متغیر سراسری time را از 1 شروع کنیم. هنگام خروج از رشته‌ی درخت اگر time=2V بود معلوم می‌شود تمام گره‌ها در درخت DFS قرار دارند و G همبند است. در غیر این صورت گراف ناهمبند است. O(V+E)

این مسئله با BFS هم قابل حل است. اگر درخت BFS دارای V گره بود، گراف همبند است. O(V+E)

مثال تشخیص ناهمبندی این گراف با اجرای DFS



۱+۲+۴=۷ دی ۱+۲+۱۱=۱۴

اگر در لحظه خروج از ریشه، time < ۱۴ باشد گراف ناهمبند است.

DFS(a)

	a	b	c	d	e	f	g
ورود	۱	۲	۳	۵			
خروج	۸	۷	۴	۶			

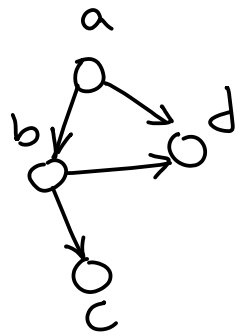
گراف G ناهمبند است.

ترتیب توپولوژیک برای DAG : کاربرد مهمی از DFS است. پس از اجرای DFS(a) گره‌ها را با اولویت لغطی خروج از آن‌ها در لیست پیوندی درج می‌کنیم.

$$O(V + E + V)$$

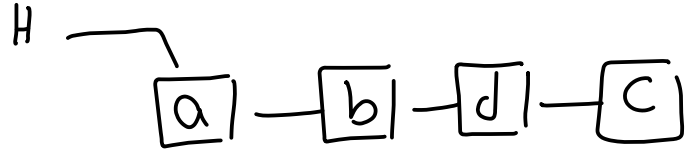
مثال یافتن ترتیب توپولوژیک DAG با استفاده از DFS:

DAG



	a	b	c	d	e	f	g
ورود	۱	۲	۳	۵			
خروج	۸	۷	۴	۶			

لیست پیوندی:

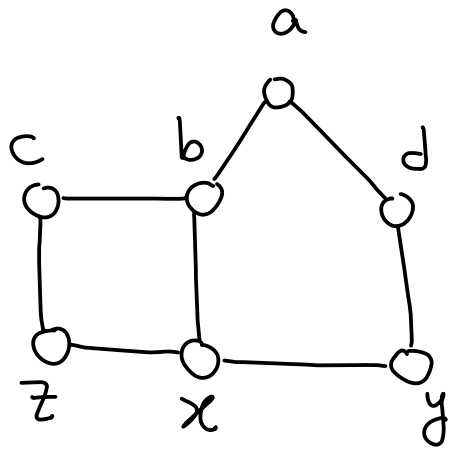


تشخیص دوبخشی بودن :

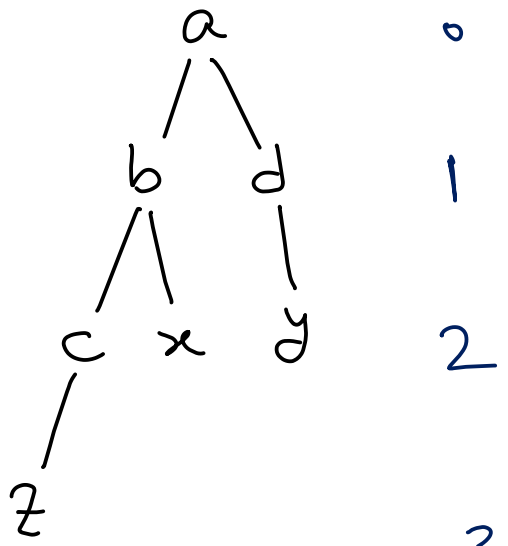
از کاربردهای معروف BFS است. پس از اجرای BFS از رأس دلخواه a و تعیین سطح هر گره، می‌توان در گراف G به تک‌تک یال‌ها دقت کرد اگر اختلاف سطح دو سر یال‌ها فرد باشد G دوبخشی است.

$O(V+E+E)$

مثال تشخیص دوبخشی بودن یا نبودن با BFS.



BFS(a)



یال (x,y) که در درخت نیست:

$d(x) - d(y) = 2 - 2 = 0 \rightarrow$ زوج

پس G دوبخشی نیست.

اگر برای هر یال (x,y) داشته باشیم

فرد $= |d(x) - d(y)|$ آنگاه G دوبخشی نیست.

۶۱- **برای پیاده‌سازی مؤثر کدام‌یک از الگوریتم‌های زیر، ساختمان داده heap لازم نیست؟**

۱) پریم

۲) دایکسترا

۳) هافمن

۴) کراسکال

پاسخ: گزینه ۴

در الگوریتم‌های پریم، دایکسترا و هافمن، در هر گام از بین انتخاب‌های موجود، کمینه را انتخاب می‌کنیم. به همین دلیل به ساختار هرمی نیاز داریم تا هر بار بتوانیم کاهش کلید یا استخراج کمینه را مدیریت کنیم.

اما در کراسکال همان ابتدا همه یال‌ها را برحسب وزن آن‌ها به شکل غیرنزولی (از سبک‌ترین به سنگین‌ترین) مرتب می‌کنیم و در هر مرحله نیازی به استخراج کمینه نداریم.

۶۲- یک آرایه دوبعدی $n \times n$ از اعداد داده شده‌است که هر سطر به‌صورت صعودی از چپ به راست و هر ستون به‌صورت صعودی از بالا به پایین مرتب شده‌است. کمترین پیچیدگی زمانی برای پیداکردن یک عدد داده‌شده در این آرایه دوبعدی کدام است؟

۱) $O(n \log n)$

۲) $O(\log n)$

۳) $O(n^2)$

۴) $O(n)$

پاسخ: گزینه (۴)

(مسائلی معروف و پرتکرار آزمون‌های ارشد و دکتری)

حداکثر در ۲۱ گام جنگو به پایان می‌رسد. توضیح کامل‌تر در اسلایدهای بعدی:

جستجوی کلید K :

شروع: $i=1$, $j=n$

تا وقتی که $1 \le i \le n$ و $1 \le j \le n$ تکرار کنیم:

IF $K > x_{ij} \Longrightarrow i=i+1$

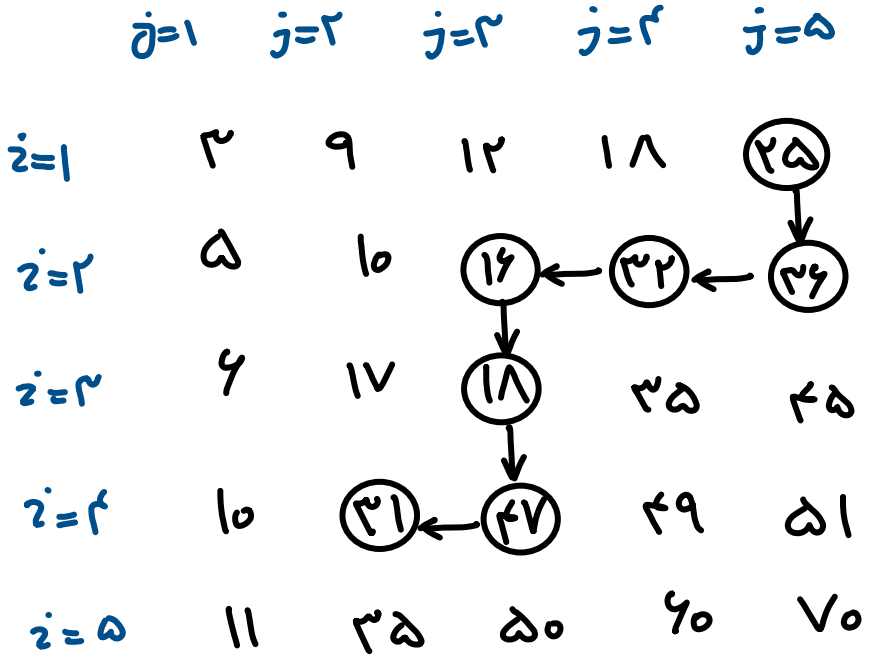
IF $K < x_{ij} \Longrightarrow j=j-1$

IF $K = x_{ij} \Longrightarrow$ END

در خاتمه: اگر $K=x_{ij}$ جستجو موفق بوده

اگر $i>n$ یا $j>n$ جستجو ناموفق بوده

مثال: جستجوی $k=۳۱$.



۶۳- یک الگوریتم یک آرایه دوبعدی $n \times n$ را به‌عنوان ورودی دریافت می‌کند و زمان اجـرای آن برحسـب n عبـارت است از $T(n) = O(n \log n)$، زمان الگوریتم برحسب اندازه ورودی $m = n^2$ کدام است؟

۱) $O(m^2 \log m)$

۲) $O(m^2 \log^2 m)$

۳) $O(\sqrt{m} \log m)$

۴) $O(m \log m)$

پاسخ: گزینه ۳

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

$$m = n^2 \Rightarrow n = \sqrt{m}$$

$$T(n) = O(\sqrt{m} \log \sqrt{m}) = O(\sqrt{m} \log m^{\frac{1}{2}}) = O(\frac{1}{2} \sqrt{m} \log m) = O(\sqrt{m} \log m)$$

۶۴- **فرض کنید برای حل یک مسئله باید از بین چهار الگوریتم انتخاب کنید. کدام‌یک ارجحیت دارد؟**

۱) الگوریتم A نمونه‌ای به اندازه n را با حل بازگشتی بیست نمونه با اندازه $\frac{n}{۳}$ حل می‌کند و سپس راه‌حل‌های آنها را در زمان $O(n^۲)$ ترکیب می‌کنند.

۲) الگوریتم B نمونه‌ای به اندازه n را با حل بازگشتی هشت نمونه با اندازه $\frac{n}{۲}$ حل می‌کند و سپس راه‌حل‌های آنها را در زمان $O(n^۳)$ ترکیب می‌کنند.

۳) الگوریتم C نمونه‌ای به اندازه n را با حل بازگشتی دو نمونه با اندازه $۲\times n$ حل می‌کند و سپس راه‌حل‌های آنها را در زمان $O(n)$ ترکیب می‌کنند.

۴) الگوریتم D نمونه‌ای به اندازه n را با حل بازگشتی دو نمونه با اندازه $n-۱$ حل می‌کند و سپس راه‌حل‌های آنها را در زمان ثابت ترکیب می‌کنند.

پاسخ تمرین (۱)

① $T(n) = ۲۰\,T(\frac{n}{۳}) + O(n^۲)$

$p = \log_{۳}^{۲۰}$ $\quad ۲<p<۳$ $\Rightarrow$ $T(n) = O(n^{p}) = O(n^{\log_{۳}^{۲۰}})$

② $T(n) = ۸\,T(\frac{n}{۲}) + O(n^۳)$

$p = \log_{۲}^{۸} = ۳$ $\Rightarrow$ $T(n) = O(f(n)\log n) = O(n^۳ \log n)$

③ $T(n) = ۲\,T(۲n) + O(n)$ $\Rightarrow$ $T(n) = \infty$

④ $T(n) = ۲\,T(n-۱) + c$ $\Rightarrow$ $T(n) \geq ۲\,T(n-۱)$ $\Rightarrow$ $T(n) = \Omega(۲^n)$

۶۵- فرض کنید $A = [a_1, a_2, \ldots, a_n]$ آرایه‌ای از اعداد صحیح متمایز باشد و k یک عدد صحیح داده شده باشد. هدف این است که دو عدد متمایز از A را پیدا کنید که مجموع آنها دقیقاً k باشد، یا گزارش دهید که چنین عناصری وجود ندارد. یک الگوریتم کارا برای حل این مسئله چه زمانی خواهد داشت؟

۱) $O(\log n)$

۲) $O(n)$

۳) $O(n \log n)$

۴) $O(n^2)$

پاسخ: گزینه (۳)

ابتدا: مرتب سازی لیست A (از اعداد صحیح تشکیل شده اگر کران بالایش $\Theta(n)$ بود در $O(n)$ مرتب می‌شد) $(a_i \leq n)$

اما در حالت کلی زمان مرتب سازی $O(n\log n)$ است.

پس: برای هر $1 \leq i \leq n$ ، عملیات جبری $k - a_i$ در لیست مرتب را انجام دهیم:

$$O(n \log n)$$

۶۶- الگوریتم مرتب‌سازی ادغام (Megesort) را درنظر بگیرید. به‌جای اینکه آرایه ورودی را به دو قسمت تقریباً مساوی تقسیم کنیم، آن را به **m** بخش تقسیم می‌کنیم که لزوماً مساوی نیستند (**m** یک ثابت است.) سپس به‌صورت بازگشتی الگوریتم ادغام اصلاح‌شده را روی هر قسمت اجرا می‌کنیم و **m** آرایه مرتب‌شده را ادغام می‌کنیم. زمان اجرای الگوریتم ادغام اصلاح‌شده در حالتی که $m > ۲$ کدام است؟

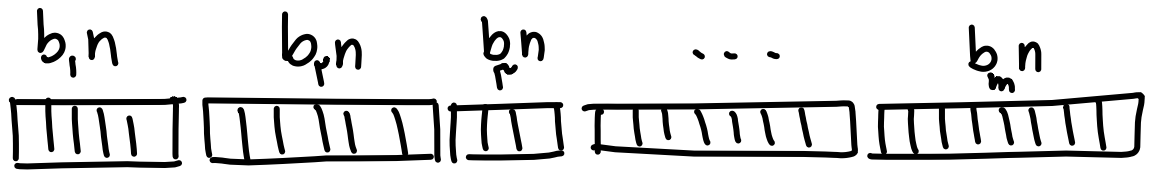
۱) $O(mn)$

۲) $O(m \log n)$

۳) $O(\log \log n)$

۴) $O(n \log n)$

پاسخ: گزینه ۴



$0 < b_i < 1$ $\qquad b_1 n + b_2 n + \cdots + b_m n = n \implies b_1 + b_2 + \cdots + b_m = 1$

$$T(n) = \sum_{i=1}^{m} T(b_i n) + \Theta(n) \qquad (a_i = 1)$$

$$p = 1 \qquad T(n) = \Theta\left(n^p \left(1 + \int \frac{f(n)}{n^{p+1}} dn\right)\right)$$

$$= \Theta\left(n \left(1 + \int \frac{n}{n^2} dn\right)\right)$$

$$= \Theta(n \log n)$$

۶۷- کدام‌یک از روابط بازگشتی زیر را نمی‌توان مستقیماً با قضیه اصلی حل کرد؟

۱) $T(n) = ۱۶\, T(\frac{n}{۴}) + n$

۲) $T(n) = T(\frac{n}{۵}) + ۲۰$

۳) $T(n) = ۳\, T(\frac{n}{۴}) + n \log n$

۴) $T(n) = ۲\, T(\frac{n}{۲}) + n \log n$

پاسخ: گزینه (۴)

همه گزینه‌ها توسط قضیه‌ی اساسی حل می‌شوند اما نسخه‌ی اصلی و قدیمی این قضیه برای گزینه (۴) جواب را مشخص نمی‌کند. به توضیحات کامل تر در ادامه دقت کنید.

شکل اصلی قضیه‌ی اساسی به این صورت است: (به ویژه به قسمت (۲) دقت کنید)

Theorem 4.1 (Master theorem)

Let $a \geq 1$ and $b > 1$ be constants, let $f(n)$ be a function, and let $T(n)$ be defined on the nonnegative integers by the recurrence

$$T(n) = aT(n/b) + f(n) ,$$

where we interpret n/b to mean either $\lfloor n/b \rfloor$ or $\lceil n/b \rceil$. Then $T(n)$ has the following asymptotic bounds:

1. If $f(n) = O(n^{\log_b a - \epsilon})$ for some constant $\epsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$.
2. If $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \lg n)$.
3. If $f(n) = \Omega(n^{\log_b a + \epsilon})$ for some constant $\epsilon > 0$, and if $af(n/b) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$. ■

اگر به قسمت (۲) این قضیه دقت کنید در حالتی که $f(n)$ و n^p از نظر درجه‌ی چند جمله‌ای یکسان نمی‌شوند فقط برای $f(n) = \Theta(n^p)$ جواب را مشخص کرده است: $T(n) = \Theta(f(n) \log n)$

البته ما می‌دانیم که اگر $f(n) = n^p \log(n)$ باشد باز هم جواب رابطه بازگشتی به دست می‌آید:

$$T(n) = \Theta(f(n) \log n) = \Theta(n^p \log n \cdot \log n)$$

اما شکل اصلی قضیه‌ی اساسی این حالت را حل نمی‌کند.

① $f(n)=n$ و $n^{\log_4 16}=n^2$ پس $T(n)=\Theta(n^2)$

② $f(n)=20=\Theta(1)$ و $n^{\log_5 1}=n^0=1$ پس $T(n)=\Theta(f(n)\log n)=\Theta(\log n)$

③ $f(n)=n\log n$ و $n^{\log_4 3}<1$ پس $T(n)=\Theta(f(n))=\Theta(n\log n)$

④ $f(n)=n\log n$ و $n^{\log_2 1}=n^1$ در این حالت قضیه اصلی اساسی جواب را نمی‌دهد.