

به نام خدا

مقدمه مؤلف

کتاب حاضر پس از چندین بار ویرایش جامع‌ترین کتاب طراحی الگوریتم برای آمادگی کنکور کارشناسی ارشد و دکتری رشته‌های علوم کامپیوتر، مهندسی فناوری اطلاعات و مهندسی کامپیوتر است. این کتاب براساس جدیدترین تغییرات منابع و تست‌های کنکور کارشناسی ارشد و دکتری، تألیف شده است. این کتاب در بازده فصل تنظیم شده است و برخی فصول آن مثل فصل‌های ۱ تا ۷ مطالب بسیار مفید و ارزنده‌ای را شامل می‌باشند و توصیه می‌شود با دقت و وسوس زیاد مطالعه شوند. در پایان برخی از فصل‌ها تمریناتی گنجانده شده است که پاسخ برخی از آنها آمده است، توصیه می‌شود تمرینات بی‌پاسخ را حل کنید و یا حداقل صورت این سؤالات را به خاطر بسپارید. البته پاسخ تمامی این تمرینات به همراه تست‌های بسیار مفید در کتاب نارنجی طراحی الگوریتم آمده است که توصیه می‌کنم برای تسلط به مطالب حتماً کتاب‌های نارنجی را مطالعه بفرمایید. همچنین توصیه می‌شود که کتاب «ساختمان داده‌ها» انتشارات پوران پژوهش را مطالعه بفرمایید. این دو کتاب تمام سرفصل‌های دروس ساختمان داده‌ها و طراحی الگوریتم را پوشش می‌دهد و کامل‌ترین منابع برای داوطلبان کنکور کارشناسی ارشد و همچنین برای دانشجویان رشته کامپیوتر هستند.

سعی شده است مطالب بدون اشکال باشند ولی ممکن است اشکالاتی در کتاب مشاهده بفرمایید و یا احساس کنید مطلبی که متوجه نمی‌شوید، نادرست است. اینجانب از شما خواننده گرامی تقاضا می‌کنم اشکالات احتمالی را مطرح بفرمایید تا با یکدیگر تبادل نظر کنیم. دوستان عزیزی در گرداوری این کتاب رحمت فراوان کشیده‌اند. همچنین، رحمات مدیر محترم تولید، آقای حسین رحیمی، و نیز خانم خسروی که با دقیق وصف ناپذیر آماده‌سازی کتاب را بر عهده داشتند، جای تقدیر ویژه دارد.

هادی یوسفی - زمستان ۱۴۰۱



@yousefi_hadi

۰۹۱۲۱۷۸۸۵۳۴

فهرست مطالب

فصل اول. مقدمات ریاضی، رشد توابع، نمادهای مجانبی	۱
خواص سیگما	۱
یافتن کران برای حاصل جمع	۳
لکاریتم	۶
رشد تابع (Growth of functions)	۷
تابع \lg^* (لگ استار)	۱۱
نمادهای مجانبی asymptotic notations	۱۲
تمرین	۲۰
سؤالهای چهارگزینه‌ای فصل اول	۲۵
فصل دوم. تحلیل الگوریتم‌های غیربازگشتی – آنالیز استهلاکی	۳۵
تحلیل الگوریتم‌های غیربازگشتی	۳۵
آنالیز استهلاکی (Amortized Analysis)	۴۲
تمرین	۴۸
سؤالهای چهارگزینه‌ای فصل دوم	۴۹
فصل سوم. روابط بازگشتی – تحلیل الگوریتم‌های بازگشتی – تقسیم و غلبه.	۶۵
حل روابط بازگشتی خطی همگن / ناهمگن ضریب ثابت	۶۷
قضیه اساسی	۷۱
درخت بازگشت	۷۳
قضیه بمب اتم	۷۵
الگوریتم‌های بازگشتی	۷۶
تقسیم و غلبه (divide & conquer)	۸۰
ضرب ماتریس‌ها	۸۰
ضرب دو چند جمله‌ای	۸۲
ضرب اعداد بزرگ	۸۶
مسائل معروف بازگشتی	۸۷
تمرین	۹۳
سؤالهای چهارگزینه‌ای فصل سوم	۹۶
فصل چهارم. جستجو و درهم‌سازی	۱۳۲
جستجوی دودویی (binary search)	۱۳۲

جستجوی دودویی در یک دنباله چرخشی	۱۳۳
جستجوی دودویی برای یک اندیس خاص	۱۳۴
جستجوی دودویی در دنباله با طول نامشخص	۱۳۴
جستجوی درون‌بابی (interpolation search)	۱۳۴
درهم‌سازی	۱۳۵
سؤال‌های چهارگزینه‌ای فصل چهارم	۱۳۹
فصل پنجم. مرتبه‌های آماری و مرتب‌سازی	
یافتن \min و \max در $A[1..n]$	۱۵۵
یافتن دومین مینیمم (یا دومین ماکزیمم)	۱۵۸
یافتن عنصر کمینه k ام (امین مینیمم)	۱۵۹
مرتب‌سازی	۱۶۴
روش‌های مرتب‌سازی غیر مقایسه‌ای	۱۷۱
تمرین	۱۷۷
سؤال‌های چهارگزینه‌ای فصل پنجم	۱۸۶
فصل ششم. مباحثی از درخت‌ها	
پیمایش درخت دودویی	۲۱۹
Heap	۲۲۱
صف اولویت priority queue	۲۲۵
هیپ دو جمله‌ای (Binomial heap)	۲۲۶
هیپ فیبوناچی	۲۲۸
درخت جستجوی دودویی (BST) Binary Search Tree	۲۲۹
AVL	۲۳۴
مجموعه‌های مجرزا	۲۳۷
تمرین	۲۴۰
سؤال‌های چهارگزینه‌ای فصل ششم	۲۴۱
فصل هفتم. گراف	
گراف	۲۷۶
پیمایش گراف	۲۷۶
پیمایش عمقی (DFS)	۲۷۷
پیمایش سطحی (BFS)	۲۸۶

۲۸۹	درخت پوشای مینیمم (Minimum Spanning Tree:MST)
۲۸۹	الگوریتم کراسکال (kruskal)
۲۹۱	الگوریتم پریم (Prim)
۲۹۲	کوتاهترین مسیرهای هم مبدأ (Single-Source Shortest Paths)
۲۹۴	الگوریتم بلمن فورد
۲۹۵	یافتن کوتاهترین مسیرهای هم مبدأ در گراف جهتدار بدون سیکل
۲۹۶	الگوریتم دایجسترا (Dijkstra)
۲۹۸	شار (جیان) بیشینه (max flow)
۳۰۶	تمرین
۳۱۰	سؤالهای چهارگزینه‌ای فصل هفتم
۳۵۵	فصل هشتم. روش‌های حریصانه (greedy)
۳۵۵	مقدمه
۳۵۶	۱- مساله کوله‌پشتی غیرصفر و یک (کوله پشتی کسری)
۳۵۹	۲- کدهافمن
۳۶۲	۳- زمان‌بندی بر مبنای کمینه کردن زمان کل
۳۶۲	۴- انتخاب (زمان‌بندی) فعالیتها
۳۶۳	۵- زمان‌بندی فعالیت‌ها با مهلت معین (Scheduling with Dead Lines)
۳۶۷	سؤالهای چهارگزینه‌ای فصل هشتم
۳۷۹	فصل نهم. برنامه‌نویسی پویا
۳۷۹	مقدمه
۳۸۲	۱- ضرب زنجیره‌ای ماتریس‌ها
۳۸۷	۲- الگوریتم فلود برای یافتن تمام کوتاهترین مسیرها
۳۹۳	۳- کوله پشتی ۰-۰ با ارزش ماکریم
۳۹۴	۴- فروشنده دوره‌گرد
۳۵۱	۵- درخت جستجوی دودویی بهینه (Optimal BST)
۳۹۹	۶- بزرگترین زیردنباله مشترک (LCS)
۴۰۱	۷- ضربی دو جمله‌ای
۴۰۲	۸- زمانبندی خط تولید (assembly-line scheduling)
۴۰۴	۹- خرد کردن سکه
۴۰۶	۱۰- برش میله
۴۰۸	۱۱- مسابقات جهانی
۴۱۰	سؤالهای چهارگزینه‌ای فصل نهم

فصل دهم. بازگشت به عقب و انشعاب و تحدید (مطالعه آزاد).....	۴۲۹
سوال‌های چهارگزینه‌ای فصل دهم	۴۴۱
فصل یازدهم. آشنایی با نظریه NP.....	۴۴۵
سوال‌های چهارگزینه‌ای فصل یازدهم	۴۵۴
کنکورهای سراسری ارشد و دکتری ۱۳۹۶-۱۴۰۱	۴۶۱

فصل ۱

مقدمات ریاضی، رشد توابع، نمادهای مجانبی

مقدمات ریاضی

برای تحلیل و بررسی الگوریتم‌ها نیاز به دانستن برخی مفاهیم ریاضی است. مثلاً برای محاسبه تعداد اجرای برخی از حلقه‌ها از سیگما استفاده می‌کنیم.

خواص سیگما

حاصل جمع $\sum_{k=1}^n a_k$ می‌نویسیم. و اگر تعداد جملات

نامتناهی باشد یعنی $\lim_{n \rightarrow \infty} \sum_{k=1}^n a_k$ یا $\sum_{k=1}^{\infty} a_k$ می‌نویسیم. به خاطر

سپاری روابط زیر در تحلیل الگوریتم‌ها مفید است:

$$1) \sum_{k=1}^n k = 1 + 2 + 3 + \dots + n = \frac{n(n+1)}{2} \quad (\text{تصاعد حسابی با قدر نسبت } 1)$$

$$2) \sum_{k=1}^n k^r = 1^r + 2^r + 3^r + \dots + n^r = \frac{n(n+1)(2n+1)}{6}$$

$$3) \sum_{k=1}^n k^r = 1^r + 2^r + 3^r + \dots + n^r = \left(\frac{n(n+1)}{2} \right)^r = \frac{n^r(n+1)^r}{4}$$

$$4) \sum_{k=0}^n ax^k = a + ax + ax^r + \dots + ax^n = a(1 + x + x^r + \dots + x^n)$$

$$= \frac{a(x^{n+1} - 1)}{x - 1} = \frac{a(1 - x^{n+1})}{1 - x} \quad (\text{تصاهد هندسی با قدر نسبت } x)$$

در تصاعد هندسی اگر $x < 1$ و $n \rightarrow \infty$ آنگاه $x^{n+1} \rightarrow 0$ و داریم:

$$5) \sum_{k=0}^{\infty} ax^k = a + ax + ax^2 + \dots = \frac{a}{1-x}$$

توجه: برای اثبات تساوی‌های فوق می‌توان از استقرای ریاضی کمک گرفت. یک مدل از استقرای ریاضی چنین است که فرض کنید $P(n)$ حکمی است راجع به عدد صحیح n . حال اگر $P(1)$ درست باشد و از درستی $P(k)$ درستی $P(k+1)$ نتیجه شود آنگاه $P(n)$ به ازای همه اعداد صحیح $n \geq 1$ درست است. پس برای اثبات حکم $P(n)$ با استقراء باید سه مرحله را طی کنیم:

۱- نشان دهیم $P(1)$ درست است.

۲- فرض کنیم $P(k)$ درست است.

۳- ثابت کنیم $P(k+1)$ درست است.

مثال ۱: درستی تساوی $\sum_{k=1}^n k^3 = \frac{n^2(n+1)^2}{4}$ را ثابت کنید.

پاسخ: فرض کنید $P(n)$ عبارت است از $\sum_{k=1}^n k^3 = \frac{n^2(n+1)^2}{4}$. حال با استقرا اثبات می‌کنیم.

مرحله ۱: $P(1)$ یعنی $1^3 = \frac{1^2(1+1)^2}{4}$ درست است.

مرحله ۲: فرض می‌کنیم $P(k)$ یعنی $\sum_{k=1}^k k^3 = \frac{k^2(k+1)^2}{4}$ درست است.

مرحله ۳: ثابت می‌کنیم $P(k+1)$ یعنی $\sum_{k=1}^{k+1} k^3 = \frac{(k+1)^2(k+2)^2}{4}$ درست است. که برای اثبات از مرحله ۲ کمک می‌گیریم و بجای k^3 $(k+1)^3$ مقدار

قرار می‌دهیم پس داریم:

$$\begin{aligned} 1^3 + 2^3 + \dots + k^3 + (k+1)^3 &= \frac{k^2(k+1)^2}{4} + (k+1)^3 = \frac{(k+1)^2(k^2 + 4k + 4)}{4} \\ &= \frac{(k+1)^2(k+2)^2}{4} \quad \blacksquare \end{aligned}$$

توجه: می‌توان از سری هندسی (روابط ۴ و ۵) مشتق یا انتگرال گرفت و فرمول‌های جدیدی یافت.

مثال ۲: حاصل $\sum_{k=1}^{\infty} \frac{k}{2^k}$ را بیابید.

پاسخ: با توجه به فرمول ۵ داریم $\frac{1}{1-x} = 1 + x + x^2 + x^3 + \dots$

می‌گیریم و داریم: $\frac{1}{(1-x)^2} = 1 + 2x + 3x^2 + \dots$ حال طرفین را در x ضرب می‌کنیم و داریم:

$$x + 2x^2 + 3x^3 + \dots = \frac{x}{(1-x)^2}$$

$$\frac{1}{2} + 2 \times \frac{1}{2^2} + 3 \times \frac{1}{2^3} + \dots = \sum_{k=1}^{\infty} k \times \frac{1}{2^k} = \frac{\frac{1}{2}}{\left(1 - \frac{1}{2}\right)^2} = 2 \quad \blacksquare$$

توجه: در این درس بعضی مواقع نیاز به محاسبه حد توابع هست. می‌دانیم چند جمله‌ای

وقتی $a_n x^n + a_{n-1} x^{n-1} + \dots + a_0$ هم ارز است.

مثال ۳: حد $\frac{1+2+3+\dots+n}{3n^2+2n-5}$ وقتی $n \rightarrow \infty$ را بیابید.

پاسخ: است که با $\frac{1}{2}n^2$ هم ارز است و 5 با $3n^2 + 2n$ هم ارز است پس این حد برابر $\frac{n(n+1)}{2}$ است که برابر $\frac{1}{2}n^2$ هم ارز است پس این حد برابر حد $\frac{1}{2}$ می‌باشد. ■

یافتن کران برای حاصل جمع

تکنیک‌های مختلفی وجود دارد که برای یک حاصل جمع می‌تواند کران بالا و پایین مشخص کند، یک تکنیک مهم انتگرال است. ابتدا تعریف تابع صعودی و نزولی را یادآوری می‌کنیم. تابع $y = f(x)$ را صعودی گویند هرگاه اگر $j > i$ نتیجه شود $f(i) \leq f(j)$. مثلاً تابع $y = f(x) = 2x + 1$ صعودی است. تابع $y = f(x) = x$ را نزولی گوییم هرگاه اگر $j > i$ نتیجه شود $f(i) \geq f(j)$.

برای حاصل جمع $\sum_{k=m}^n f(k)$ اگر $f(x)$ صعودی باشد می‌توان با انتگرال برای این حاصل

جمع کران مشخص کرد:

$$\int_{m-1}^n f(x) dx \leq \sum_{k=m}^n f(k) \leq \int_m^{n+1} f(x) dx$$

و اگر $f(x)$ نزولی باشد داریم:

$$\int_m^{n+1} f(x) dx \leq \sum_{k=m}^n f(k) \leq \int_{m-1}^n f(x) dx$$

مثال ۴: سری همسازه (هارمونیک) $\sum_{k=1}^n \frac{1}{k} = 1 + \frac{1}{2} + \frac{1}{3} + \dots + \frac{1}{n}$ است. کران بالا و پایین این سری را بیابید.

پاسخ: تابع $f(k) = \frac{1}{k}$ نزولی است پس کران پایین به صورت زیر بدست می‌آید:

$$\sum_{k=1}^n \frac{1}{k} \geq \int_1^{n+1} \frac{dx}{x} = \ln(n+1)$$

برای کران بالا داریم: $(\sum_{k=1}^n \frac{1}{k}) \leq \ln(n+1)$ شروع می‌کنیم که برای انتگرال مشکل ایجاد نشود)

$$\sum_{k=1}^n \frac{1}{k} \leq \int_1^n \frac{dx}{x} = \ln n \Rightarrow \sum_{k=1}^n \frac{1}{k} \leq \ln n + 1$$

بنابراین سری همسازه بین $\ln(n+1)$ و $\ln(n+1) + 1$ است. پس می‌توان نتیجه گرفت

$$\sum_{k=1}^n \frac{1}{k} = \ln n + c \quad \text{که } c \text{ ثابت است.}$$

توجه: اگر در دنباله $\sum_{k=1}^n a_k$ و برای هر $k \geq 1$ داشته باشیم $a_k \leq r$ ثابت بین ۰ و ۱ است

آنگاه می‌توان برای این دنباله با کمک سری هندسی کران بالا یافت:

$$\frac{a_{k+1}}{a_k} \leq r \rightarrow a_{k+1} \leq a_k r \rightarrow a_k \leq a_{k-1} r \leq a_{k-2} r^2 \leq \dots \leq a_1 r^k$$

$$\sum_{k=1}^n a_k \leq \sum_{k=1}^{\infty} a_1 r^k = \frac{a_1}{1-r}$$

در نتیجه:

مثال ۵: برای $\sum_{k=1}^{\infty} \frac{k}{3^k}$ کران بالا بیابید.

پاسخ: ابتدا سیگما را طوری می‌نویسیم که از $k=0$ شروع شود:

$$\left(\sum_{k=m}^n f(k) \right) = \sum_{k=m-P}^{n-P} f(k+P)$$

$$\sum_{k=1}^{\infty} \frac{k}{3^k} = \sum_{k=0}^{\infty} \frac{k+1}{3^{k+1}}$$

جمله اول $a_0 = \frac{1}{3}$ است و می‌توان ثابت r را برابر $\frac{2}{3}$ انتخاب کرد زیرا:

$$\frac{a_{k+1}}{a_k} = \frac{(k+2)/3^{k+2}}{(k+1)/3^{k+1}} = \frac{1}{3} \frac{k+2}{k+1} \leq \frac{2}{3} \rightarrow r = \frac{2}{3}$$

بنابراین با توجه به $\sum_{k=0}^{\infty} a_k \leq \frac{a_0}{1-r}$ داریم:

$$\sum_{k=1}^{\infty} \frac{k}{r^k} = \sum_{k=0}^{\infty} \frac{k+1}{r^{k+1}} \leq \frac{\frac{1}{r}}{1-\frac{1}{r}} = 1$$

توجه: مراقب استفاده نادرست از این تکنیک باشید! مثلًا برای سری همسازه $\sum_{k=1}^{\infty} \frac{1}{k}$ نسبت

$\frac{a_{k+1}}{a_k} = \frac{k}{k+1}$ از یک کوچکتر است ولی نمی‌توان مقدار $r < 1$ را یافت. زیرا $\frac{k}{k+1}$ به ۱ میل می‌کند و هر مقدار برای r در نظر بگیرید به ازای بعضی مقادیر k نادرست است. وقت کنید r باید ثابت باشد. در واقع $r < 1$ به ازای هیچ $r < 1$ درست نیست. در مورد سری همسازه اگر این اشتباه را مرتكب شوید و از این تکنیک استفاده کنید، سری همسازه را به یک عدد ثابت محدود می‌کنید ولی می‌دانیم سری همسازه با Lnn معادل است و به ∞ میل می‌کند.

تسنی ۱: در یک زمستان سرد، یک خرس قطبی، n قطعه گوشت دقیقاً به اندازه‌های ۱ و ۲ و n را در غاری ذخیره کرده است. او هر روز یکی از این قطعه گوشت‌ها را به صورت تصادفی انتخاب می‌کند. اگر اندازه گوشت عدد فرد باشد، آن را کاملاً می‌خورد. اگر زوج باشد، نصف گوشت را خورده و نصف دیگر را مجدداً در غار قرار می‌دهد. اگر گوشتی موجود نباشد، خرس می‌میرد. برای n های بزرگ، خرس حدوداً چند روز زنده است؟

$$n^2 (4) \quad 2n (3) \quad 2^n (2) \quad n \lg n (1)$$

پاسخ: n قطعه گوشت یعنی حداقل n روز خرس زنده است. این n قطعه، نصفشان زوج و نصف فرد است. فردها خورده می‌شوند ولی زوج‌ها نصف می‌شوند. پس خرس $\frac{n}{2}$ روز دیگر زنده می‌ماند. این $\frac{n}{2}$ قطعه گوشت نصفشان زوج است پس خرس $\frac{n}{4}$ روز دیگر زنده می‌ماند و به همین ترتیب تا اینکه تعداد قطعات گوشت $1 = \frac{n}{2^k}$ و سپس صفر شود و خرس بمیرد. پس تعداد روزهای زنده ماندن خرس برابر است با:

$$n + \frac{n}{2} + \frac{n}{4} + \dots + \frac{n}{2^k} \leq n + \frac{n}{2} + \frac{n}{4} + \dots = n(1 + \frac{1}{2} + \frac{1}{4} + \dots) = n \times \frac{1}{1 - \frac{1}{2}} = 2n$$

پس گزینه (۳) صحیح است.

لگاریتم

لگاریتم معکوس تابع توان است. مثلاً $f(x) = 2^x$ آنگاه $f^{-1}(x) = \log_2 x$. به عبارت ساده‌تر، اگر $a^b = c$ آنگاه $\log_a^c = b$ و بر عکس. مثلاً $\log_2^8 = 3$ زیرا $2^3 = 8$. یا $\log_{10}^{1000} = 3$ زیرا $10^3 = 1000$. در این درس معمولاً لگاریتم‌ها در پایه ۲ هستند و بنابراین پایه ۲ را نمی‌نویسیم و به طور خاص برای لگاریتم پایه ۲ از علامت \lg استفاده می‌کنیم پس قرار داد می‌کنیم $\log_2^n = \lg n = \log n$. فرض کنید می‌خواهیم $\lg 1000$ را بیاییم چون $2^9 < 1000 < 2^{10}$ پس $\lg 1000$ بین ۹ و ۱۰ است و سقف آن برابر ۱۰ می‌شود. علامت \lceil سقف (ceiling) و علامت \lfloor کف (floor) است. $\lceil x \rceil$ برابر کوچکترین عدد صحیح بزرگتر یا مساوی x است و $\lfloor x \rfloor$ برابر بزرگترین عدد صحیح کوچکتر یا مساوی x است. مثلاً $\lceil 2/5 \rceil = 1$ و $\lfloor 2/5 \rfloor = 0$ یا $\lceil 3 \rceil = 3$ و $\lfloor 3 \rceil = 2$. برخی خواص لگاریتم عبارتند از:

$$1) \log_c^{ab} = \log_c^a + \log_c^b \quad 2) \log_c^{\frac{a}{b}} = \log_c^a - \log_c^b$$

$$3) \log_{b^n}^{a^m} = \frac{m}{n} \log_b^a \quad 4) \log_b^a = \frac{\log_c^a}{\log_c^b}$$

$$5) \log_b^a = \frac{1}{\log_a^b} \quad 6) b^{\log_b^a} = a$$

$$7) a^{\log_c^b} = b^{\log_c^a} \quad 8) \log_b^a \times \log_c^b = \log_c^a$$

مثال ۶: تساوی‌های زیر را نشان دهید:

$$a) n^{\frac{1}{\lg n}} = 2 \quad b) (\lg n)^{\lg n} = n^{\lg \lg n} \quad c) 4^{\lg n} = n^2$$

$$d) 2^{\lg n} = n \quad e) (\sqrt{2})^{\lg n} = \sqrt{n} \quad f) 2^{\sqrt{2 \lg n}} = n^{\sqrt{\frac{2}{\lg n}}}$$

پاسخ: شماره خواصی که استفاده می‌کنیم را ذکر می‌کنیم:

$$a) \text{طبق خاصیت ۵، } n^{\log_2^n} = 2^{\frac{1}{\lg n}} \text{ و طبق خاصیت ۶، } \log_2^n = \frac{1}{\lg n}$$

$$b) \text{طبق خاصیت ۷، در } n^{\lg \lg n} \text{ می‌توان جای } n \text{ و } \lg n \text{ را عوض کرد پس } . n^{\lg \lg n} = (\lg n)^{\lg n}$$

$$c) \text{طبق خاصیت ۷، می‌توان ۴ را با } n \text{ تعویض کرد پس } . 4^{\lg n} = n^{\lg 4} = n^2$$

$$d) \text{طبق خاصیت ۷، می‌توان ۲ را با } n \text{ تعویض کرد پس } . 2^{\lg n} = n^{\lg 2} = n$$

$$\sqrt{2^{\lg n}} = (2^{\frac{1}{2}})^{\lg n} = 2^{\frac{1}{2} \lg n} = 2^{\lg n^{\frac{1}{2}}} = n^{\frac{1}{2}} = \sqrt{n} \quad (\text{e})$$

طبق خاصیت ۶ می‌توان گفت $x = n^{\log_n^x}$ پس: (f)

$$\begin{aligned} \sqrt[2]{\sqrt[2]{\lg n}} &= n^{\log_n^{\sqrt[2]{\lg n}}} = n^{\sqrt[2]{\lg n} \times \log_n^{\sqrt[2]{} \cdot \frac{1}{2}}} = n^{\sqrt[2]{\lg n} \times \frac{1}{2}} \\ &= n^{\sqrt{\lg n \times \frac{1}{2}}} = n^{\sqrt{\frac{1}{2} \lg n}} \end{aligned}$$

رشد توابع (Growth of functions)

می‌خواهیم بررسی کنیم وقتی n را خیلی بزرگ کنیم، زمان اجرای الگوریتم‌ها بر حسب n (ن اندازه ورودی الگوریتم است) چگونه زیاد می‌شود. می‌خواهیم اینباری ارائه دهیم که بتوان رشد تابع را با هم مقایسه کرد و بر اساس آن تشخیص دهیم که مثلاً زمان اجرای الگوریتمی کم یا زیاد است.

مثال ۷: تابع n و $\lg n$ و 2^n و n^2 و n^3 را به ترتیب افزایش رشد مرتب کنید.

پاسخ: در جدول مقابل مقدار این تابع به ازای برخی n ‌ها آمده است.

ترتیب رشد این توابع به صورت $\lg n < n < n \lg n < n^2 < n^3 < 2^n$ است. توجه کنید تابع n^3 تا $n=8$ از تابع 2^n مقدار بیشتری دارد ولی رشد 2^n بیشتر است. برای تشخیص رشد خیلی موقع نمی‌توان عددگذاری انجام داد. چون از یک نقطه به بعد ممکن است رفتار یک تابع مشخص شود.

	۱	۲	۴	۸	۱۶	۳۲
تابع						
$\lg n$	۰	۱	۲	۳	۴	۵
n	۱	۲	۴	۸	۱۶	۳۲
$n \lg n$	۰	۲	۸	۲۴	۶۴	۱۶۰
n^2	۱	۴	۱۶	۶۴	۲۵۶	۱۰۲۴
n^3	۱	۸	۶۴	۵۱۲	۴۰۹۶	۳۲۷۶۸
2^n	۲	۴	۱۶	۲۵۶	۶۵۵۳۶	۴۲۹۴۹۶۷۲۹۶

مثال ۸: رشد دو تابع n^3 و $2n^2$ را با هم مقایسه کنید.

پاسخ: به ازای هر مقدار n ، مقدار تابع $2n^2$ از n^3 بیشتر است (دو برابر است) ولی به این معنی نیست که رشد $2n^2$ از n^3 بیشتر است. این دو تابع هم رشد هستند یعنی ضریب ثابت در رشد تاثیری ندارد. ■

یک روش برای مقایسه رشد دو تابع $f(n)$ و $g(n)$ استفاده از حدگیری است. حد نسبت وقتی $n \rightarrow +\infty$ اگر برابر صفر شد یعنی رشد $g(n)$ بیشتر است. اگر ∞ شد یعنی رشد $\frac{f(n)}{g(n)}$ بیشتر است و اگر عدد ثابت $k \neq 0$ شد یعنی هم رشد هستند:

$$\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = \begin{cases} 0 & \Leftrightarrow \text{رشد } g(n) \text{ از } f(n) \text{ کمتر است.} \\ \infty & \Leftrightarrow \text{رشد } g(n) \text{ از } f(n) \text{ بیشتر است.} \\ k \neq 0 & \Leftrightarrow \text{رشد } g(n) \text{ با } f(n) \text{ هم رشد است.} \end{cases}$$

مثال ۹: رشد $f(n) = \lg n$ و $g(n) = \sqrt{n}$ را با هم مقایسه کنید.

پاسخ: از حد استفاده می‌کنیم و می‌دانیم اگر حد مبهم $\frac{\infty}{\infty}$ یا $\frac{0}{0}$ شود می‌توان از هوپیتال کمک

$$\log_a^n = \lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = \lim_{n \rightarrow \infty} \frac{\frac{1}{n}}{\frac{1}{2\sqrt{n}}} = \lim_{n \rightarrow \infty} \frac{1}{\frac{n}{2\sqrt{n}}} = \lim_{n \rightarrow \infty} \frac{1}{\frac{n}{\sqrt{n}}} = \lim_{n \rightarrow \infty} \frac{1}{\sqrt{n}} = 0$$

برابر $\frac{1}{n \ln a}$ است پس:

$$\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = \lim_{n \rightarrow \infty} \frac{\lg n}{\sqrt{n}} = \frac{\infty}{\infty} \text{ مبهم}$$

$$\rightarrow \lim_{n \rightarrow \infty} \frac{f'(n)}{g'(n)} = \lim_{n \rightarrow \infty} \frac{\frac{1}{n \ln 2}}{\frac{1}{2\sqrt{n}}} = \lim_{n \rightarrow \infty} \frac{2\sqrt{n}}{\ln 2 \times n} = \lim_{n \rightarrow \infty} \frac{2}{\ln 2 \times \sqrt{n}} = 0$$

پس رشد $\lg n$ از \sqrt{n} کمتر است.

مثال ۱۰: رشد $f(n) = \log_4^n$ و $g(n) = \log_3^n$ را با هم مقایسه کنید.

پاسخ: از حد کمک می‌گیریم و می‌دانیم $\log_b^a = \frac{\ln a}{\ln b}$ پس:

$$\lim_{n \rightarrow \infty} \frac{\log_4^n}{\log_3^n} = \lim_{n \rightarrow \infty} \frac{\frac{\ln n}{\ln 4}}{\frac{\ln n}{\ln 3}} = \lim_{n \rightarrow \infty} \frac{\ln 4}{\ln 3} = \log_3 4 \neq 0$$

بنابراین \log_4^n با \log_3^n هم رشد است. ■

لازم به ذکر است که ما فرض می‌کنیم دامنه توابع اعداد طبیعی $\{1, 2, 3, \dots\} = \mathbb{N}$ است.

(توجه کنید در برخی منابع $\{0, 1, 2, 3, \dots\} = \mathbb{N}$ تعریف می‌شود) و همچنین توابع در این درس،

مثبت هستند یعنی $f(n) \geq 0$. البته توابع می‌توانند صعودی یا نزولی یا حتی متناوب باشند.

مثال ۱۱ : رشد $f(n) = \frac{1}{n}$ و $g(n) = \frac{1}{n^2}$ را با هم مقایسه کنید.

پاسخ: رشد $\frac{1}{n}$ از $\frac{1}{n^2}$ کمتر است:

$$\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = \lim_{n \rightarrow \infty} \frac{\frac{1}{n}}{\frac{1}{n^2}} = \lim_{n \rightarrow \infty} n = \infty$$

مثال ۱۲ : رشد $f(n) = 2^n$ و $g(n) = 2^{n^2}$ را با هم مقایسه کنید.

پاسخ: هم رشد هستند چون حاصل حد عدد ثابت مخالف صفر است:

$$\lim_{n \rightarrow \infty} \frac{2^n}{2^{n^2}} = \lim_{n \rightarrow \infty} \frac{2^n}{2^n \cdot 2^{n^2-n}} = \lim_{n \rightarrow \infty} \frac{1}{2^{n^2-n}} = 0$$

در ادامه نکاتی را مطرح می‌کنیم که در بررسی رشد توابع بسیار مفید هستند:

۱- ضریب ثابت تاثیری در رشد ندارد مثلاً n^4 با n^2 هم رشد است. دقت کنید n^2 رشد کمتری

از n^4 دارد. یعنی تابع $f(n)$ که $c \cdot f(n)$ با c ثابت، هم رشد است.

۲- رشد تابع n^a از n^b بیشتر است که $a > b$ عدد ثابت هستند و $a > b$. مثلاً n^3 از n^2 رشد

بیشتری دارد. یا رشد $n^{1/2}$ از رشد $n^{1/5}$ بیشتر است. یا رشد n^{-1} از n^{-2} بیشتر است.

۳- رشد a^n از b^n بیشتر است که a, b عدد ثابت هستند و $a > b$. مثلاً رشد 2^n از $(1/5)^n$

بیشتر است. یا رشد $n^{(1/9)}$ از $n^{(1/8)}$ بیشتر است.

۴- رشد تابع a^n از b^n به شرطی که a, b ثابت و $a > 1$, بیشتر است. مثلاً رشد تابع نمایی 2^n

از رشد تابع چند جمله‌ای n^{200} بیشتر است. یا رشد $n^{1/10}$ از رشد $n^{1/100}$ بیشتر است. دقت کنید

رشد $n^{9/10}$ از n^2 کمتر است. چون $n^{9/10}$ تابع نزولی است و n^2 صعودی است.

۵- رشد n^a از n^b که a, b ثابت و $a > b$, بیشتر است. مثلاً $\sqrt{n} = n^{1/2}$ رشد

بیشتری از $n^{1/3}$ دارد.

۶- رشد $(\lg n)^b$ از $(\lg \lg n)^a$ که a, b ثابت و $a > b$, بیشتر است. مثلاً

رشد $\lg^4 n$ از $\lg^2 n$ رشد بیشتری دارد.

۷- رشد \log_a^n با \log_b^n که $a, b > 1$ ثابت، یکسان است. مثلاً \log_4^n با \log_2^n هم رشد هستند.

دقت کنید تابعی مثل $\log_{1/2}^n$ مقادیر منفی دارد و ما تابع مثبت را بررسی می‌کنیم پس چنین

تابعی اصلاً مورد بحث نیست.

-۸- رشد n^n از $n!$ بیشتر است.

-۹- تابع $\lg(n!)$ با $n \cdot \lg n$ هم رشد است.

-۱۰- اگر $f(n)$ و $g(n)$ صعودی باشند و $\lg(g(n))$ رشد بیشتری داشته باشد آنگاه

$\lg(f(n))$ رشد بیشتری دارد. مثلاً رشد $g(n) = n^{\lg n}$ از $f(n) = \lg^n n$ بیشتر است

زیرا $\lg(g(n)) = \lg^n n > \lg(f(n)) = n \cdot \lg \lg n$ رشد بیشتری دارد.

-۱۱- اگر رشد $f(n)$ از $g(n)$ بیشتر باشد آنگاه رشد $\lg(g(n))$ از رشد $\lg(f(n))$ بیشتر یا مساوی است. (f و g توابع صعودی)

توجه: برای اثبات ۱ تا ۸ می‌توانید از حد استفاده کنید. اثبات ۹ را خواهیم دید.

مثال ۱۲: توابع هر قسمت را به ترتیب افزایش رشد مرتب کنید.

$$f_1 = n^r, f_2 = \lg^r n, f_3 = \left(\frac{r}{2}\right)^n, f_4 = 2^{rn}, f_5 = \lg(n!), f_6 = n^{\frac{1}{\lg n}} \quad (a)$$

$$f_1 = r^{\lg n}, f_2 = e^n, f_3 = (n+2)! , f_4 = \sqrt{\lg n}, f_5 = 2^{\lg n} \quad (b)$$

$$f_1 = r^{(\lg n + \lg \lg n)}, f_2 = n^r \lg^r \sqrt{n} \quad (c)$$

$$f_1 = n!, f_2 = (1.005)^n, f_3 = n^{1000} \quad (d)$$

$$f_1 = 2^n, f_2 = e^{\sqrt{n}}, f_3 = e^{\sqrt{n}}, f_4 = e^{\frac{n}{r}} \quad (e)$$

$$f_1 = n(\log_r n)^r, f_2 = n^{1/r}, f_3 = \lg n, f_4 = \frac{n}{\lg n} \quad (f)$$

پاسخ:

(a) می‌دانیم $\lg n! \sim n \cdot \lg n$ و $n^{\frac{1}{\lg n}} = 2$ (" ~ " یعنی هم رشد هستند) پس:

$$f_6 < f_2 < f_5 < f_1 < f_3 < f_4$$

$$f_4 < f_5 < f_1 < f_2 < f_3 \quad 2^{\lg n} = n \text{ و } r^{\lg n} = n^r \quad (b)$$

و f_2 را ساده می‌کنیم: (c)

$$f_1 = r^{\lg n + \lg \lg n} = r^{\lg n} \times r^{\lg \lg n} = n^{\lg r} \times (\lg n)^{\lg r} = n^r \cdot \lg^r n$$

$$f_2 = n^r \lg^r \sqrt{n} = n^r (\lg \sqrt{n})^r = n^r \left(\frac{1}{2} \lg n\right)^r \sim n^r \lg^r n$$

در نتیجه f_1 از f_2 رشد کمتری دارد.

(d) با توجه به اینکه چند جمله‌ای از نمایی و نمایی از فاکتوریل رشد کمتری دارد پس

$$f_3 < f_2 < f_1$$

$e^{\frac{n}{\sqrt{n}}} = \sqrt{e^n}$ از $e^{2\sqrt{n}}$ رشد کمتری دارد و هر دو از $e^{\frac{n}{2}}$ و 2^n رشد کمتری دارند و $e^{\sqrt{n}}$ و $\sqrt{e^n}$ پس $f_3 < f_2 < f_4 < f_1 < f_5$ ، در نتیجه $f_1 < 2^n < e^{\sqrt{n}} < \sqrt{e^n} < 2^{\lg n}$ با توجه به اینکه $\lg n$ پس طرفین را به $\lg n$ تقسیم کنیم برای مقایسه $\frac{n}{\lg n}$ و $\lg n$ داریم $\frac{n}{\lg n} < \lg n$. سایر توابع به راحتی قابل بررسی هستند: $f_3 < f_4 < f_1 < f_2 < f_5$.

تابع \lg^* (لگ استار)

$\lg^* n$ برابر است با تعداد باری که باید از n لگاریتم پایه ۲ بگیریم که حاصل برابر عدد ۱ یا کمتر از ۱ شود. مثلاً $\lg^* 16 = 3$ زیرا اگر از ۱۶ سه بار لگاریتم پایه ۲ بگیریم، حاصل برابر ۱ می‌شود. $(\lg \lg \lg 16 = 1)$. به چند نمونه دقت کنید:

$\lg^* 2 = 1$, $\lg^* 2^2 = \lg^* 4 = 2$, $\lg^* 2^3 = \lg^* 16 = 3$, $\lg^* 2^4 = \lg^* 65536 = 4$, $\lg^* 2^{65536} = 5$

$\lg^* n$ تابعی است با رشد بسیار ناچیز. رشد این تابع از تمام توابع صعودی که تاکنون معرفی کردیم کمتر است.

مثال ۱۴ : توابع هر قسمت را به ترتیب افزایش رشد مرتب کنید.

$$f_1 = n^\gamma, f_2 = \gamma^{lg^* n}, f_3 = n!, f_4 = 2^{\gamma^{n+1}}, f_5 = (\sqrt{2})^{\lg n} \quad (a)$$

$$f_1 = 2^n, f_2 = \lg n, f_3 = \lg \lg n, f_4 = n \cdot 2^n, f_5 = \sqrt{\lg n}, f_6 = \lg^* n, f_7 = n^{\lg \lg n} \quad (b)$$

$$f_1 = n, f_2 = \left(\frac{3}{2}\right)^n, f_3 = 2\sqrt{2 \lg n}, f_4 = \lg^* n^n, f_5 = n \cdot \lg n, f_6 = n^{\gamma + \lg n} \quad (c)$$

پاسخ:

$$\sqrt{2} \lg n = 2^{\frac{1}{2} \lg n} = 2^{\lg \sqrt{n}} = \sqrt{n}. \quad (a)$$

برای مقایسه $2^{\gamma^{n+1}}$ با $n!$ ، از این دو تابع لگاریتم بگیریم.

$\lg 2^{\gamma^{n+1}} = \gamma^{n+1} \lg 2$ و $\lg n! = \sum_{k=1}^n \lg k$ رشد بیشتری از $n!$ دارد:

$$f_2 < f_5 < f_1 < f_3 < f_4$$

$$f_6 < f_3 < f_1 < f_5 < f_2 < f_4 \quad (b)$$

$$f_4 < f_3 < f_1 < f_5 < f_6 < f_2 \quad (c)$$

نمادهای مجانبی asymptotic notations

برای بیان زمان اجرا یا حافظه مصرفی الگوریتم‌ها از نمادهای مجانبی استفاده می‌شود که عبارتند از $O, \Omega, \Theta, o, \omega$. این نمادها را به زبان ساده تعریف می‌کنیم و سپس تعریف دقیق‌تر آن را بیان می‌کنیم.

نماد O : گوییم تابع $f(n)$ از مرتبه O تابع $g(n)$ است و می‌نویسیم $f(n) \in O(g(n))$ یا $f(n) = O(g(n))$ هرگاه رشد تابع $f(n)$ از تابع $g(n)$ کمتر یا مساوی باشد. مثلاً $3n^3 + 2n + 4 \in O(n^3)$ هم رشد است. یا $2n + \lg n \in O(2^n)$ زیرا $2n + \lg n$ رشد کمتری از 2^n دارد.

نماد Ω : گوییم تابع $f(n)$ از مرتبه Ω تابع $g(n)$ است و می‌نویسیم $f(n) \in \Omega(g(n))$ یا $f(n) = \Omega(g(n))$ هرگاه رشد تابع $f(n)$ از تابع $g(n)$ بیشتر یا مساوی باشد. مثلاً $5n^3 + 2n + 4 \in \Omega(5n^3)$ هم رشد است. یا $3n^3 + 2n \notin \Omega(n^3)$ زیرا $3n^3 + 2n + 4 \in \Omega(n^3)$ رشد بیشتری دارد. ولی $3n^3 + 2n \in \Omega(n^3)$ چون $3n^3 + 2n$ رشد کمتری از n^3 دارد.

نماد Θ (تتا): گوییم تابع $f(n)$ از مرتبه $\Theta(g(n))$ است و می‌نویسیم $f(n) = \Theta(g(n))$ یا $f(n) \in \Theta(g(n))$ هرگاه $f(n)$ و $g(n)$ هم رشد باشند. مثلاً $3n^3 + 2n + 4 \in \Theta(5n^3)$ چون $5n^3 + 2n + 4 \in \Theta(n \cdot 2^n + 3^n + n)$ چون $n \cdot 2^n + 3^n + n$ رشد است. یا $3n^3 + 2n + 4 \in \Theta(n^3)$ هم رشد است. دقت کنید در جمع تعدادی تابع، تابعی که رشد بیشتری دارد برنده است. و $3n^3 + 2n + 4 \in \Theta(n \cdot 2^n + n)$ و از $n \cdot 2^n + n$ رشد بیشتری دارد پس n^3 برنده است.

نماد o یا little oh: گوییم تابع $f(n)$ از مرتبه $o(g(n))$ است و می‌نویسیم $f(n) \in o(g(n))$ یا $f(n) = o(g(n))$ هرگاه $f(n) \in o(g(n))$ از $g(n)$ کمتر باشد. مثلاً $n \cdot 2^n + n \in o(n^3)$ چون رشد $n^3 + 2n + n$ از $n \cdot 2^n + n$ کمتر است. یا $3n^3 + 2n + 4 \in o(n^3)$ چون رشد $n \cdot 2^n + n$ از $n \cdot 2^n + n + 4$ کمتر است. ولی $3n^3 + 2n + 4 \in o(n^3)$ با n^3 یکسان است و کمتر نیست.

نماد ω : گوییم تابع $f(n)$ از مرتبه $\omega(g(n))$ است و می‌نویسیم $f(n) \in \omega(g(n))$ یا $f(n) = \omega(g(n))$ هرگاه $f(n) \in \omega(g(n))$ از $g(n)$ بیشتر باشد. مثلاً $\sqrt{n} + \lg n \in \omega(\lg n)$ زیرا رشد $\sqrt{n} + \lg n$ از $\lg n$ بیشتر است. یا $\sqrt{n} + \lg n \in \omega(5n)$ رشد $\sqrt{n} + \lg n$ با \sqrt{n} یکسان است که از $\lg n$ بیشتر است. یا $\sqrt{n} + \lg n \in \omega(\sqrt{n})$ ولی $\sqrt{n} + \lg n \notin \omega(\sqrt{n})$.

تست ۲: توابع $h(n) = \log^r n$ و $g(n) = (\log n)^{\log n}$ را در نظر بگیرید. کدام یک از گزاره‌های زیر صحیح است؟
(تخصیص هوش مصنوعی ۸۵)

$$f(n) \in O(g(n)), f(n) \in \Omega(h(n)) \quad (1)$$

$$g(n) \in \Omega(h(n)), h(n) \in \Omega(f(n)) \quad (2)$$

$$f(n) \in \Theta(h(n)), g(n) \in \Omega(f(n)) \quad (3)$$

$$h(n) \in O(g(n)), f(n) \in \Theta(g(n)) \quad (4)$$

پاسخ: با توجه به اینکه $n^{\lg \lg n} = n^{\lg \lg n}, \log^{n^2} = n^{\lg \lg n}$ رشد کمتری دارد (چون ۲ از $\lg \lg n$ رشد کمتری دارد) پس $f(n) \in O(g(n))$ یا $g(n) \in \Omega(f(n))$ از طرفی $h(n)$ از $\log n$ رشد بیشتری دارد (چون ۲ از $\log n$ رشد بیشتری دارد) و همچنین $n^{\lg \lg n}$ از $\log n$ رشد بیشتری دارد پس ترتیب رشدها $g < f < h$ است که گزینه ۱ صحیح است.

تست ۳: کدام گزینه صحیح است؟
(علوم کامپیوتر ۸۴)

$$\sqrt{n} = O(\lg n) \quad (2) \qquad n^r \lg n = O(n^{r+\varepsilon}), \varepsilon > 0 \quad (1)$$

$$n^r = O\left(\frac{n^r}{\lg n}\right) \quad (4) \qquad n^{r+\varepsilon} = O(n \cdot \lg n), \varepsilon > 0 \quad (3)$$

پاسخ: می‌دانیم n^r از $\lg n$ رشد بیشتری دارد (به ازای هر $\varepsilon > 0$). پس اگر در n^r ضرب کنید $n^r \lg n$ از $n^{r+\varepsilon}$ رشد بیشتری دارد بنابراین گزینه ۱ صحیح است. رشد \sqrt{n} از $n \cdot \lg n$ بیشتر است. رشد $n^{r+\varepsilon}$ از $n \cdot \lg n$ بیشتر است ($\varepsilon > 0$). و رشد n^r از $\frac{n^r}{\lg n}$ بیشتر است.

تست ۴: کدام عبارت صحیح است؟
(علوم کامپیوتر ۸۱)

$$2^n = O(2^n) \quad (2) \qquad \sum_{i=0}^n i^r = O(n^r) \quad (1)$$

$$\frac{n^r}{\lg n} = \Omega(n^r) \quad (4) \qquad n^r \lg n = O(n^r) \quad (3)$$

پاسخ: می‌دانیم $\sum_{i=1}^n i^r = \frac{n(n+1)(2n+1)}{6} = \Theta(n^r)$ از 2^n پس گزینه ۱ صحیح است. رشد n^r از

$\frac{n^r}{\lg n}$ بیشتر است. رشد $n^r \lg n$ از n^r بیشتر است. رشد n^r از $\frac{n^r}{\lg n}$ کمتر است.

نکات و خواص نمادها

۱- حدگیری: می‌توان با حدگرفتن رابطه بین توابع را مشخص کرد:

$$\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = \begin{cases} \infty & \Leftrightarrow f(n) \in o(g(n)) \rightarrow f(n) \in O(g(n)) \\ \infty & \Leftrightarrow f(n) \in \omega(g(n)) \rightarrow f(n) \in \Omega(g(n)) \\ k \neq \infty & \Leftrightarrow f(n) \in \theta(g(n)) \end{cases}$$

توجه کنید اگر $f(n) \in O(g(n))$ آنگاه $f(n) \in o(g(n))$ ولی لزوماً عکس این مطلب صحیح نیست. زیرا اگر رشد $f(n)$ از $g(n)$ کمتر باشد می‌توان گفت رشد $f(n)$ از $g(n)$ یا کمتر یا مساوی است. ولی اگر رشد $f(n)$ کمتر یا مساوی $g(n)$ باشد نمی‌توان گفت رشد $f(n)$ کمتر از $g(n)$ است. به همین ترتیب اگر $f(n) \in \omega(g(n))$ آنگاه $f(n) \in \Omega(g(n))$ ولی نه لزوماً بر عکس.

۲- ماکزیمم‌گیری: $\theta(f+g)$ با $\theta(\max(f,g))$ برابر است. مثلاً $\theta(n^3 + 2n)$ با $\theta(n^3)$ مساوی است. به همین ترتیب $O(f+g)$ با $O(\max(f,g))$ و $\Omega(f+g)$ با $\Omega(\max(f,g))$ برابر است.

۳- بازتابی (انعکاسی): نمادهای O و Ω و θ بازتاب هستند یعنی $f(n) \in \theta(f(n))$ و $f(n) \notin \omega(f(n))$ و $f(n) \notin o(f(n))$ ولی $f(n) \in \Omega(f(n))$ و $f(n) \in O(f(n))$

۴- تقارنی: فقط نماد θ متقارن است یعنی اگر $f(n) \in \theta(g(n))$ و $g(n) \in \theta(f(n))$ آنگاه $f(n) \in \theta(g(n))$ و $g(n) \in \theta(f(n))$ بر عکس.

۵- تقارنی ترانهاده (transpose symmetric): اگر $f(n) \in O(g(n))$ آنگاه $g(n) \in \Omega(f(n))$ و برعکس. و اگر $f(n) \in \omega(g(n))$ آنگاه $g(n) \in o(f(n))$ و برعکس.

۶- تعیدی (transitive): همه نمادها خاصیت تعیدی دارند. اگر $f(n) \in \theta(g(n))$ و $g(n) \in \theta(h(n))$ آنگاه $f(n) \in \theta(h(n))$. بجای نماد θ می‌توان نمادهای O, ω, Ω, o قرار داد.

تست ۵: فرض کنید f و g دو تابع دلخواه به شکل $f, g : \mathbb{N} \rightarrow \mathbb{R}^+$ بعلاوه فرض کنید:

$$(ساختمان داده ۸۲) \quad \text{کدام گزینه صحیح است؟} \quad \lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = \infty$$

$$f(n) \in O(g(n)), g(n) \notin \Omega(f(n)) \quad (1)$$

$$g(n) \in O(f(n)), f(n) \in O(g(n)) \quad (2)$$

$$f(n) \in \theta(g(n)), g(n) \notin \Omega(f(n)) \quad (3)$$

$$f(n) \in \Omega(g(n)), f(n) \notin \theta(g(n)) \quad (4)$$

پاسخ: چون حاصل حد برابر بینهایت است پس رشد $f(n) \in \Omega(g(n))$ بیشتر است پس $f \in \theta(g)$ ولی $f \notin \omega(g)$. می‌توان گفت $f \in \theta(g)$. گزینه ۴ صحیح است.

(علوم کامپیوتر ۸۰)

تست ۶: گزینه درست کدام است؟

$$f(n) \in O(g(n)), h(n) \in \Omega(g(n)) \Rightarrow f(n) \in \theta(h(n)) \quad (1)$$

$$f(n) \in \theta(g(n)), g(n) \in O(h(n)) \Rightarrow h(n) \in O(f(n)) \quad (2)$$

$$f(n) \in \Omega(g(n)), h(n) \in O(g(n)) \Rightarrow f(n) \in \theta(g(n)) \quad (3)$$

$$f(n) \in \Omega(g(n)), g(n) \in \theta(h(n)) \Rightarrow f(n) \in \Omega(h(n)) \quad (4)$$

پاسخ: در گزینه ۱، $f \leq g$ (منظور رشد توابع است) و $h \geq g$ از این دو نمی‌توان نتیجه گرفت f و h هم رشد هستند.

در گزینه ۲، $f \approx g$ و $h \leq g$. از این دو نمی‌توان نتیجه گرفت $h \leq f$ بلکه می‌توان گفت $f \leq h$ یعنی $f \in O(h)$. (منظور از \approx یعنی هم رشد هستند)

در گزینه ۳، $f \geq g$ و $h \leq g$. از این دو نمی‌توان نتیجه گرفت $h \leq f$ و نمی‌توان نتیجه گرفت $g \leq f$.

در گزینه ۴، $f \geq g$ و $g \approx h$ که می‌توان نتیجه گرفت $f \geq h$. پس گزینه ۴ صحیح است. ■

- اشتراک: اگر $f(n) \in \theta(g(n))$ و $f(n) \in O(g(n))$ آنگاه $f(n) \in \theta(g(n)) \cap O(g(n))$ در واقع

اشتراک O , O برابر θ است یعنی $O(f(n)) \cap \Omega(f(n)) = \theta(f(n))$

- اشتراک: اشتراک $O(f(n)) \cap \omega(f(n)) = \phi$ با $O(f(n))$ برابر تهی است:

مثال ۱۵: $O(n^3)$ و $\Omega(n^3)$ شامل چه توابعی هستند؟

پاسخ: $O(n^3)$ یعنی توابعی با رشد کمتر یا مساوی n^3 . در واقع $f \in O(n^3)$ اگر رشد f از رشد n^3 کمتر یا مساوی باشد. $\Omega(n^3)$ یعنی توابع با رشد بیشتر یا مساوی n^3 . بنابراین $O(n^3) \cap \Omega(n^3)$ یعنی $\theta(n^3)$. دقت کنید $O(n^3) \cap \Omega(n^3) = O(n^3)$ یک مجموعه است که شامل بیشمار تابع است که رشد این توابع از n^3 کمتر یا مساوی است. پس بهتر است از علامت عضویت استفاده شود. مثلاً بگوییم $n^2 \in O(n^3)$ نه $2n^3 \in O(n^3)$ هر چند استفاده علامت تساوی درست نیست ولی مراجع آن را پذیرفته‌اند پس هم می‌توان نوشت $2n^3 \in O(n^3)$ و هم $O(n^3) \subseteq 2n^3$.

تعریف ریاضی نمادهای مجانبی

مفهوم نمادها را اکنون می‌دانیم. می‌خواهیم نمادها را به طور دقیق‌تری تعریف کنیم:

نماد **O**: تابع $f(n)$ از مرتبه $O(g(n))$ است هرگاه بتوان اعداد بزرگ‌تر از صفر n_0 و c را

طوری یافت که به ازای همه اعداد صحیح $n \geq n_0$ ، داشته باشیم $f(n) \leq c \cdot g(n)$

$$\exists c, n_0 > 0, \forall n \geq n_0 : f(n) \leq c \cdot g(n)$$

مثالاً $(c = 5 \in O(n^2))$ زیرا $4n^2 + 2n + 3 \leq cn^2$ به ازای مثلاً $c = 5$ می‌تواند هر عدد بزرگتر از ۴ باشد) نامساوی برای های بیشتر یا مساوی $n_0 = 3$ برقرار است. و یا $4n^2 + 2n + 3 \notin O(n)$ زیرا نامساوی $4n^2 + 2n + 3 \leq cn$ برقرار نیست. زیرا c را هر مقداری انتخاب کنیم، از یک نقطه n_0 به بعد $4n^2 + 2n + 3$ از cn بزرگتر می‌شود چون تابع درجه ۲ از درجه ۱ رشد بیشتری دارد. در واقع با این تعریف ریاضی مشخص می‌شود که $f(n) \in O(g(n))$ یعنی رشد $f(n)$ از رشد $g(n)$ کمتر یا مساوی است.

نماد Ω : گوییم $f(n) \in \Omega(g(n))$ هرگاه:

$$\exists c, n_0 > 0, \forall n \geq n_0 : f(n) \geq c.g(n)$$

یعنی بتوان اعداد مثبت c و n_0 را طوری یافت که به ازای همه n های بزرگتر مساوی n_0 ، نامساوی $f(n) \geq c.g(n)$ برقرار باشد. مثلاً $4n^2 + 2n + 3 \in \Omega(n^2)$ زیرا نامساوی $4n^2 + 2n + 3 \geq cn^2$ به ازای مثلاً $c = 4$ و هر n مثبتی برای همه n های از n_0 بیشتر برقرار است.

نماد Θ : گوییم $f(n) \in \Theta(g(n))$ هرگاه:

$$\exists c_1, c_2, n_0 > 0, \forall n \geq n_0 : c_1.g(n) \leq f(n) \leq c_2.g(n)$$

مثالاً $(c_1 = 4, c_2 = 5)$ زیرا $4n^2 + 2n + 3 \leq c_1.n^2 \leq 4n^2$ به ازای مثلاً $n_0 = 3$ و $c_2 = 5$ به ازای همه n های از n_0 بیشتر برقرار است.

نماد Θ : گوییم $f(n) \in \Theta(g(n))$ هرگاه نامساوی $f(n) < c.g(n)$ به ازای همه c های مثبت از یک مقدار n_0 به بعد برقرار باشد:

$$\forall c > 0, \exists n_0 > 0, \forall n \geq n_0 : f(n) < c.g(n)$$

توجه کنید تفاوت تعریف ریاضی O با Θ این است که در O گفته می‌شود به ازای برخی c ها و در Θ گفته می‌شود به ازای همه c ها. مثلاً $n^2 + 2n \notin O(n^2)$ چون نامساوی $n^2 + 2n < cn^2$ به ازای $c = \frac{1}{2}$ برقرار نیست. لازم به ذکر است که $f(n) < c.g(n)$ را می‌توان به شکل $f(n) \leq c.g(n)$ نیز نوشت. مثلاً $c = \frac{1}{2}$ نیز به ازای $f(n) < c.g(n)$ برقرار نیست. در واقع در O تعریف ریاضی O و Θ می‌توان $f(n) \leq c.g(n)$ نوشت و مهم این است که O می‌گوید به ازای برخی c ها و Θ می‌گوید به ازای همه c ها.

مثال ۱۶ : آیا $n^3 + 2n \in o(n^3)$ درست است؟

پاسخ: بله. زیرا $n^3 + 2n < c.n^3$ یا $n^3 + 2n \leq c.n^3$ به ازای همه مقادیر $c > 0$ از یک n_0 به بعد برقرار است.

نماد ω : گوییم $f(n) \in \omega(g(n))$ هرگاه:

$$\forall c > 0, \exists n_0 > 0, \forall n \geq n_0 : f(n) > c.g(n) \text{ or } f(n) \geq c.g(n)$$

این تعریف نشان می‌دهد که رشد $f(n)$ از $g(n)$ بیشتر است.

مثال ۱۷ : آیا می‌توان ادعا کرد به ازای هر دو تابع $f(n)$ و $g(n)$ همواره $f(n) \in O(g(n))$ یا $f(n) \in \Omega(g(n))$ ؟

پاسخ: هر چند برای اکثر توابع این جمله درست است ولی برای برخی توابع مثلاً متناسب لزوماً درست نیست. مثلاً فرض کنید $f(n) = n^{1+\sin n}$ و $g(n) = n$. حال بررسی می‌کنیم $f(n) \in O(g(n))$ نادرست است زیرا نامساوی $n \leq c.n^{1+\sin n}$ به ازای موقوعی که $\sin n$ منفی می‌شود ($0 < \sin n < -1$) نادرست است. $f(n) \in \Omega(g(n))$ نادرست است زیرا نامساوی $n \geq c.n^{1+\sin n}$ به ازای موقوعی که $\sin n < 1$ مثبت می‌شود. ($1 < \sin n < 0$) نادرست است.

خلاصه فصل

۱- خواص سیگما را بررسی کردیم و مهم‌ترین نتیجه‌ای که می‌توان گرفت:

$$\sum_{k=1}^n k^p = 1^p + 2^p + \dots + n^p \sim \frac{1}{p+1} n^{p+1} = \theta(n^{p+1}) \quad (p \geq 0)$$

که برای درک درستی این رابطه می‌توانید بجای سیگما، انتگرال قرار دهید.

$$2- \text{سری همسازه (هارمونیک)} \sum_{k=1}^n \frac{1}{k} = \theta(\ln n) \quad \text{است که با } \ln n \text{ معادل است، پس}$$

لگاریتم‌ها با هر پایه ثابتی هم رشد هستند.

$$3- \text{سری هندسی} \quad 1 + x + x^2 + \dots + x^n = \frac{1 - x^{n+1}}{1 - x}$$

$$\text{گرفت} \quad 1 + x + x^2 + \dots = \frac{1}{1-x}$$

$$1 + 2x + 3x^2 + \dots = \frac{1}{(1-x)^2}$$

یافت.

$$4- \text{خواص لگاریتم بررسی شد. } \log_b^a = c \quad \text{یعنی} \quad a = b^c \quad \text{و قرارداد شد که} \quad \log_n^a = \lg n \quad \text{بنویسیم.}$$

۵- رشد توابع را بررسی کردیم و می‌توان ترتیب رشد توابع معروف را به شکل زیر نوشت: (a) ثابت (mثبت)

$$\dots < \frac{1}{n^m} < \frac{1}{n} < 1 < \lg^* n < \lg^a n < \lg^a n < n < n \cdot \lg n < n^2 < n^3 < \dots$$

$$\dots < 2^n < 3^n < \dots < n! < n^n$$

(تابع $\lg^* n$ برابر است با تعداد باری که باید از n لگاریتم پایه ۲ بگیریم که حاصل برابر یک یا کمتر شود).

۶- برای مقایسه رشد $f(n)$ با $g(n)$ می‌توان از حد استفاده کرد. اگر حد وقتی $\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)}$ وقته است. اگر ∞ شود رشد $f(n)$ بیشتر است در غیر این صورت هم رشد هستند.

۷- نمادهای مجانبی بررسی شدند و می‌توان گفت مفهوم نمادها به شکل زیر است:

$$f(n) \in O(g(n)) \leftrightarrow f \text{ رشد } g \leq$$

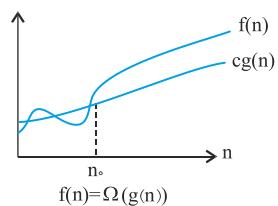
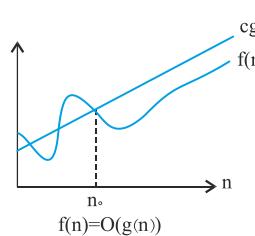
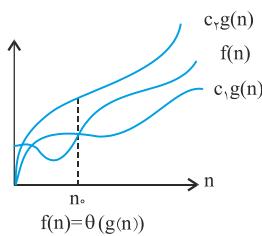
$$f(n) \in \Omega(g(n)) \leftrightarrow f \text{ رشد } g \geq$$

$$f(n) \in \Theta(g(n)) \leftrightarrow f \text{ رشد } g =$$

$$f(n) \in o(g(n)) \leftrightarrow f \text{ رشد } < g$$

$$f(n) \in \omega(g(n)) \leftrightarrow f \text{ رشد } > g$$

- برای نمادهای Θ, Ω, O می‌توان شکل زیر را کشید:



تمرین

۱- حاصل عبارات زیر را بیابید:

$$a) \sum_{k=1}^{n-1} \frac{1}{k(k+1)}$$

$$b) \sum_{k=1}^n (2k-1)$$

$$c) \sum_{k=1}^{\infty} (2k+1)x^{2k}$$

$$d) \sum_{k=0}^{\infty} \frac{(k-1)}{x^k}$$

$$e) \prod_{k=1}^n 2(4^k)$$

$$f) \prod_{k=1}^n \left(1 - \frac{1}{k^2}\right)$$

$$2- \text{نشان دهید } \sum_{k=0}^{\infty} k^r x^k = \frac{x(1+x)}{(1-x)^{r+1}}$$

$$3- \text{نشان دهید } \sum_{k=1}^n \frac{1}{(2k-1)} = \ln(\sqrt{n}) + c \text{ ثابت است.}$$

$$4- \text{نشان دهید } \sum_{k=1}^n \frac{1}{k^r} \text{ به یک ثابت محدود می‌شود.}$$

۵- برای مجموعهای زیر، بهترین کران را بیابید. $r \geq s \geq 0$

$$a) \sum_{k=1}^n k^r$$

$$b) \sum_{k=1}^n \lg^s k$$

$$c) \sum_{k=1}^n k^r \lg^s k$$

۶- ثابت کنید $\lg n! = \theta(n \lg n)$ یعنی نشان دهید $n \lg n$ با $\lg n!$ هم رشد است.

۷- درستی یا نادرستی گزاره‌های زیر را بررسی کنید و یا شرایط درستی را بیان کنید.

$$\text{اگر } f(n) \in O(g(n)) \text{ آنگاه } f(n) \in O(g(n)) \quad (a)$$

$$\text{اگر } f(n) \in o(g(n)) \text{ آنگاه } f(n) \in o(g(n)) \quad (b)$$

$$\text{اگر } \lg f(n) \in O(\lg g(n)) \text{ برای } n \text{ های به اندازه کافی بزرگ}$$

$$(f(n) \geq 1 \text{ و } \lg(g(n)) \geq 1)$$

$$f(n) \in O(f^r(n)) \quad (d)$$

$$f(n) \in \Theta\left(f\left(\frac{n}{r}\right)\right) \quad (e)$$

$$f(n) + O(f(n)) \in \Theta(f(n)) \quad (f)$$

$$O(f(n)) - \Omega(f(n)) \text{ با } o(f(n)) \text{ مساوی است.} \quad (g)$$

$$f(n) + g(n) = \Theta(\min\{f(n), g(n)\}) \quad (h)$$

$$\max\{f(n), g(n)\} = \Theta(f(n) + g(n)) \quad (i)$$

۸- آیا تابع $f(n) = \lceil \lg n \rceil!$ محدود به چند جمله‌ای است؟ یعنی آیا وجود دارد تابع n^a که رشد $f(n)$ از n^a کمتر یا مساوی باشد؟

-۹- آیا تابع $f(n) = \lceil \lg \lg n \rceil!$ محدود به چند جمله‌ای است؟

-۱۰- گویند $f(n) \in \tilde{\Omega}(g(n))$ را بخوانید امگا بی‌نهایت) اگر به ازای حداقل یک $c > 0$ نامساوی $f(n) \geq c.g(n)$ به ازای بی‌شمار n صحیح، درست باشد. آیا می‌توان ادعا کرد به ازای هر دو تابع $f(n)$ و $g(n)$ حداقل یکی از دو عبارت $f(n) \in O(g(n))$ یا $f(n) \in \tilde{\Omega}(g(n))$ صحیح است؟

-۱۱- آیا $\tilde{\Omega}$ متعدد است؟

-۱۲- نشان دهید اگر $k \cdot \ln k = \theta(n)$ آنگاه $.k = \theta\left(\frac{n}{\ln n}\right)$

-۱۳- ترتیب رشد زیر را بررسی کنید.

$$\begin{aligned} n^{\frac{1}{\lg n}} &< \lg(\lg^* n) < \lg^* n \sim \lg^*(\lg n) < \sqrt{\lg^* n} < \ln \ln n < \sqrt{\lg n} < \ln n < \lg^* n < \sqrt[3]{\lg n} \\ &< \sqrt[3]{\lg n} = \sqrt{n} < \sqrt{\lg n} = n < n \lg n \sim \lg n! < 4^{\lg n} = n^4 < n^3 < (\lg n)! < (\lg n)^{\lg n} = n^{\lg \lg n} \\ &< \left(\frac{3}{2}\right)^n < 2^n < n \times 2^n < e^n < n! < (n+1)! < 2^{n^2} < 2^{n^{n+1}} \end{aligned}$$

منظور از علامت \sim در این مثال، یعنی هم رشد هستند.

-۱۴- نشان دهید $f(n) = \begin{cases} 2^{n+2}, & \text{زوج } n \\ 0, & \text{فرد } n \end{cases}$ از مرتبه $O(f(n))$ هیچ یک تابع تمرین قبل نیست.

پاسخ تمرین

$$a) \sum_{k=1}^{n-1} \frac{1}{k(k+1)} = \sum_{k=1}^{n-1} \frac{(k+1)-k}{k(k+1)} = \sum_{k=1}^{n-1} \left(\frac{1}{k} - \frac{1}{k+1} \right) \quad -1$$

$$= \left(\frac{1}{1} - \frac{1}{2} \right) + \left(\frac{1}{2} - \frac{1}{3} \right) + \left(\frac{1}{3} - \frac{1}{4} \right) + \dots + \left(\frac{1}{n-1} - \frac{1}{n} \right) = 1 - \frac{1}{n}$$

توجه: $\sum_{k=0}^{n-1} (a_k - a_{k+1}) = a_0 - a_n$ به خاصیت تلسکوپی معروف است.

$$b) \sum_{k=1}^n (2k-1) = 2 \sum_{k=1}^n k - \sum_{k=1}^n 1 = 2 \frac{n(n+1)}{2} - n = n^2$$

توجه: $c \sum_{k=1}^n (ca_k + d.b_k) = c \sum_{k=1}^n a_k + d \sum_{k=1}^n b_k$

که c نسبت به k ثابت هستند. همچنین $\sum_{k=m}^n c = (n - m + 1)c$

$$c)x^r + x^{\Delta} + x^{\gamma} + \dots = \frac{x^r}{1-x^r} \xrightarrow{\text{مشتق}} rx^r + \Delta x^{\gamma} + \gamma x^{\delta} + \dots = \frac{rx^r(1-x^r) - (-rx)x^r}{(1-x^r)^2}$$

$$\rightarrow \sum_{k=1}^{\infty} (rk+1)x^{rk} = \frac{rx^r - x^{\gamma}}{(1-x^r)^2}$$

$$d) \sum_{k=0}^{\infty} \frac{k-1}{r^k} = \sum_{k=0}^{\infty} \frac{k}{r^k} - \sum_{k=0}^{\infty} \frac{1}{r^k} = r - \frac{1}{1-\frac{1}{r}} = r - r = 0$$

توجه: در مثال ۲ متن درس دیدیم که r شروع شود نیز همین میشود.

(e) علامت \prod به معنی ضرب است پس:

$$\prod_{k=1}^n r \times r^k = (r \times r^1)(r \times r^2)(r \times r^3) \dots (r \times r^n) = r^n r^{1+2+\dots+n}$$

$$= r^n \times r^{\frac{n(n+1)}{2}} = r^n \times r^{n(n+1)} = r^{n^2+n}$$

$$f) \prod_{k=r}^n \left(1 - \frac{1}{k^r}\right) = \prod_{k=r}^n \left(\frac{k-1}{k}\right) \left(\frac{k+1}{k}\right) = \left(\frac{1}{r} \cdot \frac{r}{2}\right) \left(\frac{2}{3} \cdot \frac{3}{r}\right) \left(\frac{3}{4} \cdot \frac{4}{3}\right) \dots \left(\frac{n-1}{n} \cdot \frac{n+1}{n}\right)$$

$$= \frac{1}{r} \cdot \frac{n+1}{n} = \frac{n+1}{rn}$$

$$1 + x + x^r + x^{r^2} + \dots = \frac{1}{1-x} \xrightarrow{\text{مشتق}} 1 + rx + r^2 x^r + r^3 x^{r^2} + \dots = \frac{1}{(1-x)^r} \quad -2$$

$$\xrightarrow{\times x} x + rx^r + r^2 x^{r^2} + r^3 x^{r^3} + \dots = \frac{x}{(1-x)^r}$$

$$\xrightarrow{\text{مشتق}} 1 + r^2 x + r^3 x^r + r^4 x^{r^2} + \dots = \frac{1+x}{(1-x)^r}$$

$$\xrightarrow{\times x} x + r^2 x^r + r^3 x^{r^2} + r^4 x^{r^3} + \dots = \frac{x(1+x)}{(1-x)^r}$$

$$\sum_{k=1}^n \frac{1}{(2k-1)} \simeq \int_1^n \frac{1}{2k-1} dk = \frac{1}{2} \ln(2n-1) = \ln \sqrt{2n-1} \simeq \ln(\sqrt{n}) + C \quad -3$$

۶- از تقریب استرلینگ برای $n!$ استفاده می‌کنیم و از آن لگاریتم می‌گیریم:

$$n! = \sqrt{2\pi n} \left(\frac{n}{e} \right)^n \left(1 + \theta \left(\frac{1}{n} \right) \right)$$

$$\rightarrow \lg n! = \lg \sqrt{2\pi n} + n \cdot \lg \left(\frac{n}{e} \right) + \lg \left(1 + \theta \left(\frac{1}{n} \right) \right) \simeq n \cdot \lg \left(\frac{n}{e} \right) = n \cdot \lg n - n \cdot \lg e \simeq n \cdot \lg n$$

پس $\lg n!$ با $n \cdot \lg n$ هم رشد می‌باشد.

(a) نادرست. مثلاً $f(n) = 2n$ و $g(n) = n$ را در نظر بگیرید.

(b) نادرست. مثلاً $g(n) = \frac{1}{n}$ و $f(n) = \frac{1}{n^2}$ را در نظر بگیرید.

(c) درست.

$$f(n) = O(g(n)) \Rightarrow f(n) \leq cg(n) \Rightarrow \lg(f(n)) \leq \lg c + \lg(g(n))$$

برای آنکه نشان دهیم این عبارت کمتر از $b \cdot \lg(g(n))$ است (برای مقدار ثابتی مثل b ، فرض می‌کنیم $\lg(g(n)) = b \cdot \lg(g(n))$) حال طرفین را به $\lg c + \lg(g(n))$ تقسیم می‌کنیم:

$$b = \frac{\lg(c) + \lg(g(n))}{\lg(g(n))} = \frac{\lg(c)}{\lg(g(n))} + 1 \leq \lg c + 1$$

(d) اگر برای n ‌های بزرگ $f(n) < o(n)$ آنگاه نادرست و اگر $f(n) > o(n)$ آنگاه درست. (مثلاً به ازای $f(n) = \frac{1}{n}$ ، نادرست است)

(e) نادرست. مثلاً $f(n) = 4^n$ را در نظر بگیرید.

(f) درست. $O(f(n))$ یعنی توابعی که از $f(n)$ رشد کمتر یا یکسان دارند)

$$g(n) \in O(n) - \Omega(n), \text{ می‌توان نشان داد} \quad (g)$$

ولی $o(f(n)) \subseteq O(f(n)) - \Omega(f(n))$. ولی $g(n) \notin o(f(n))$ یعنی $g(n) \in O(f(n)) - \Omega(f(n))$

(h) نادرست.

(i) درست. باید نشان دهیم اعداد مثبت C_1 و C_2 وجود دارند که

$$C_1(f(n) + g(n)) \leq \max(f(n), g(n)) \leq C_2(f(n) + g(n))$$

برای همه $n \geq n_0$. کافیست $C_1 = \frac{1}{2}$ و $C_2 = 1$ انتخاب شوند.

-۸- برای آنکه نشان دهیم $f(n)$ محدود به چندجمله‌ای است باید نشان دهیم $\lg(f(n))$ از $\lg(n)$ رشد بیشتری ندارد:

$$f(n) = \lceil \lg n \rceil! \rightarrow \lg(f(n)) = \lg \lceil \lg n \rceil! \simeq \lceil \lg n \rceil \cdot \lg \lceil \lg n \rceil > \lg n$$

پس $\lceil \lg n \rceil!$ محدود به چندجمله‌ای نیست یعنی رشدش از همه چندجمله‌ای‌ها بیشتر

است. (نشان دهد رشد این تابع از نمایی‌ها کمتر است)

$$f(n) = \lceil \lg \lg n \rceil! \rightarrow \lg(f(n)) = \lg \lceil \lg \lg n \rceil! \simeq \lceil \lg \lg n \rceil \cdot \lg \lceil \lg \lg n \rceil < \lg n \quad -۹$$

پس $\lceil \lg \lg n \rceil!$ محدود به چندجمله‌ای است.

-۱۰- بله. مثلاً $f(n) \in \Omega(g(n))$ و $g(n) = n$ را در نظر بگیرید آنگاه (زیرا $f(n) = n^{1+\sin n}$

نامساوی $n^{1+\sin n} \geq c \cdot n$ به ازای بی‌شمار n (نه همه n ‌ها) صحیح است.

-۱۱- خیر. Ω متعدد نیست. مثلاً $n \in \Omega(n^{1+\sin n})$ زیرا نامساوی $n \geq c \cdot n^{1+\sin n}$ به ازای

بی‌شمار n (در واقع به ازای همه n ‌هایی که $\sin n$ منفی است) برقرار است. و همچنین

$$n^{1+\sin n} \geq c \cdot n^{1/\Delta} \text{ زیرا نامساوی } n^{1+\sin n} \in \Omega(n^{1/\Delta})$$

$$n \notin \Omega(n^{1/\Delta})$$

-۱۲- در عبارت $k \cdot \ln k$ اگر بجای k قرار دهیم $\frac{n}{\ln n}$ داریم:

$$k \cdot \ln k \simeq \frac{n}{\ln n} \cdot \ln \left(\frac{n}{\ln n} \right) = \frac{n}{\ln n} (\ln n - \ln \ln n) = n - \frac{n \cdot \ln \ln n}{\ln n} \simeq n = \Theta(n)$$

سؤالهای چهارگزینه‌ای فصل اول

(علوم کامپیوتر ۸۹) -۱ کدام گزینه صحیح است؟ $(\exists x < n \text{ such that } f(x) = O(n^{1-x}))$

$$\frac{n}{\lg n} = O(n^{1-x}), \sqrt{n} = O((\lg n)^{\frac{1}{2}})$$

$$n(\log n)^{\frac{1}{2}} = O(n^{\frac{1}{2}}), \sqrt{n} = O((\lg n)^{\frac{1}{2}})$$

$$\frac{n}{\lg n} = O(n^{1-x}), \sqrt{n} = O(n^{\frac{1}{2}})$$

$$n(\log n)^{\frac{1}{2}} = O(n^{\frac{1}{2}}), \sqrt{n} = O((\lg n)^{\frac{1}{2}})$$

-۲ با فرض مثبت بودن توابع f و g ، تعداد گزاره‌های درست از میان گزاره‌های زیر چند است؟

(۸۸ IT) ۱ گزاره $f(n) = O(f(n))$ ■

۲ گزاره $f(n) + o(f(n)) = \theta(f(n))$ ■

۳ گزاره $f(n) = O(g(n)) \Rightarrow 2^{f(n)} = O(2^{g(n)})$ ■

۴ گزاره $f(n) = \theta\left(f\left(\frac{n}{2}\right)\right)$ ■

۱) یک گزاره صحیح است.

۲) دو گزاره صحیح است.

۳) سه گزاره صحیح نیست.

(علوم کامپیوتر ۸۸) -۳ برای $k > 1$ و $\epsilon < \frac{1}{k}$ کدام گزینه صحیح است؟

$$n^\epsilon = O(\sqrt{n}) = O(\lg n!) = O((\lg n)^k)$$

$$n^\epsilon = O(\sqrt{n}) = O((\lg n)^k) = O(\lg n!)$$

$$(\lg n)^k = O(n^\epsilon) = O(\lg n!) = O(\sqrt{n})$$

$$(\lg n)^k = O(n^\epsilon) = O(\sqrt{n}) = O(\lg n!)$$

(علوم کامپیوتر - ۸۵) -۴ در رشد توابع زیر کدام ترتیب صحیح است؟ (چپ به راست)

$$O((1+\epsilon)^n), O(n \log n), O\left(\frac{n^{\frac{1}{2}}}{\log n}\right) \quad O(n \log n), O((1+\epsilon)^n), O\left(\frac{n^{\frac{1}{2}}}{\log n}\right)$$

$$O(n \log n), O\left(\frac{n^{\frac{1}{2}}}{\log n}\right), O((1+\epsilon)^n) \quad O\left(\frac{n^{\frac{1}{2}}}{\log n}\right), O(n \log n), O((1+\epsilon)^n)$$

(۸۴ IT)

کدام یک از گزینه‌های زیر صحیح است؟ -۵

$$n^{\gamma} \sin n \in O(n) \quad (۲)$$

$$n^{\gamma} \sin n \in \Omega(n) \quad (۱)$$

$$n \in \Theta(n^{\gamma} \sin n) \quad (۴)$$

$$n \in \Theta(n^{\gamma} \sin n) \quad (۳)$$

(علوم کامپیوتر - ۸۲)

کدام عبارت صحیح است؟ -۶

$$(n+1)(n^{\gamma} - \gamma n + 1) \in \Theta(n) \quad (۲)$$

$$(n+1)(n^{\gamma} - \gamma n + 1) \in O(2^n) \quad (۱)$$

$$(n+1)(n^{\gamma} - \gamma n + 1) \in O(n^{\gamma} \log n) \quad (۴)$$

$$(n+1)(n^{\gamma} - \gamma n + 1) \in \Omega(n^{\gamma}) \quad (۳)$$

(علوم کامپیوتر - ۸۲)

کدام گزینه صحیح است؟ -۷

$$n^a \neq O((\log n)^b) \text{ if } a > b \text{ iff } (۲)$$

$$n^a = O((\log n)^b) \text{ if } a > b \text{ iff } (۱)$$

$$n^a \neq O((\log n)^b), \forall b, a > 0. \quad (۴)$$

$$n^a = O((\log n)^b), \forall b, a > 0. \quad (۳)$$

(علوم کامپیوتر - ۸۰)

گزینه صحیح انتخاب کنید؟ -۸

$$\log n! \in O(\log n) \quad (۲)$$

$$n^{\frac{1}{\log n}} \in \Omega(\log n) \quad (۱)$$

$$1^n + 2^n + \dots + n^n \in O(n^n) \quad (۴)$$

$$n^{\gamma} + 3n - 4 \in O(n \log n) \quad (۳)$$

(آزاد ۸۶)

کدام یک از موارد زیر نادرست است؟ -۹

$$6 \times 2^n + n^{\gamma} = O(2^n) \quad (۲)$$

$$6 \times 2^n + n^{\gamma} = \Theta(n^{\gamma}) \quad (۱)$$

$$6 \times 2^n + n^{\gamma} = \Omega(2^n) \quad (۴)$$

$$6 \times 2^n + n^{\gamma} = \Omega(2^n) \quad (۳)$$

(تخصصی نرم افزار مهندسی کامپیوتر - آزاد ۸۵)

کدام تساوی درست است؟ -۱۰

$$\frac{6n^{\gamma}}{(\log n + 1)} = O(n^{\gamma}) \quad (۲)$$

$$n^{\gamma} \log n = \Theta(n^{\gamma}) \quad (۱)$$

$$n^{\gamma} 2^n + 6n^{\gamma} 3^n = O(n^{\gamma} 2^n) \quad (۴)$$

$$\frac{n^{\gamma}}{\log n} = \Theta(n^{\gamma}) \quad (۳)$$

(آزاد ۸۳)

کدام یک از موارد زیر صحیح است؟ -۱۱

$$\frac{n^{\gamma}}{\log n} = \Theta(n^{\gamma}) \quad (ب)$$

$$n! = O(n^n) \quad (الف)$$

$$n^{\gamma} n + 6(2^n) = \Theta(n^{\gamma} n) \quad (د)$$

$$n^{\gamma} \log n = \Theta(n^{\gamma}) \quad (ج)$$

(۴) همه موارد

(۳) الف و د

(۲) ب و ج

(۱) الف و ج

(ساختمان داده‌ها – دولتی ۸۵)

-۱۲ کدام یک از عبارت زیر درست‌اند:

$$I. e^{c\sqrt{n}} = O(e^{\sqrt{n}}) \quad c \geq 1$$

$$II. n^{\gamma} = O(n \log n)$$

$$III. n = O(n \log n)$$

III و I (۴)

II و I (۳)

III (۲) فقط

I (۱) فقط

-۱۳ کدام یک ترتیب صحیح از کوچکترین نرخ رشد تا بزرگترین نرخ رشد است؟
(گسسته – دولتی ۸۴)

$$x \log x, 16x^{\gamma}, (3x)^{3/2}, x^{3/2} \log \sqrt{x} \quad (1)$$

$$x \log x, 16x^{\gamma}, x^{3/2} \log \sqrt{x}, (3x)^{3/2} \quad (2)$$

$$16x^{\gamma}, x \log x, (3x)^{3/2}, x^{3/2} \log \sqrt{x} \quad (3)$$

$$16x^{\gamma}, x \log x, x^{3/2} \log \sqrt{x}, (3x)^{3/2} \quad (4)$$

(مبانی نرم افزار – آزاد ۷۹)

-۱۴ کدام گزینه غلط است؟

if $f(n) = O(g(n))$ and $f(n) = \theta(g(n))$ then $f(n) = \Omega(g(n))$ (۱)

if $f(n) = \Omega(g(n))$ and $f(n) = \theta(g(n))$ then $f(n) = O(g(n))$ (۲)

if $f(n) = \Omega(g(n))$ and $f(n) = O(g(n))$ then $f(n) = \theta(g(n))$ (۳)

if $f(n) = O(g(n))$ and $g(n) = \Omega(f(n))$ then $f(n) = \theta(g(n))$ (۴)

-۱۵ فرض کنید $f(n) = O(g(n))$. کدام گزینه برای هر تابع با مقادیر مثبت مانند $f(n)$ و $g(n)$ برقرار است؟
(تخصص هوش مصنوعی دولتی ۸۴)

$$\gamma^{f(n)} = O(\gamma^{g(n)}) \quad (۲) \quad g(n) = \theta(f(n)) \quad (۱)$$

$$(\log(g(n))) \geq 1 \text{ با فرض: } \log(g(n)) = O(\log(f(n))) \quad (۴) \quad \gamma^{f(n)} = \Omega(\gamma^{g(n)}) \quad (۳)$$

-۱۶ با توجه به تعاریف علامت O و Ω و θ کدام گزینه غلط است؟
(مبانی نرم افزار آزاد ۷۸ و ۷۰)

$$I. O\left(\sum_{i=1}^m f_i(n)\right) = O(\max_{i=1}^m \{f_i\}) \quad (۱)$$

if $f(n) = O(g(n))$ and $g(n) = \Omega(f(n))$ then $f(n) = \theta(g(n))$ (۲)

I. $g(n) f(n) = \theta(f(n))$ به شرط اینکه $g(n) \leq c$ برای $n \geq n_0$. (۳)

II. $g(n) f(n) = \Omega(f(n))$ به شرط اینکه $g(n) \geq c$ برای $n \geq n_0$.

$$I. O(O(f(n))) = O(f(n)) \quad (۴)$$

$$II. O(f(n) + g(n)) = O(f(n)) + O(g(n))$$

-۱۷ آرایه‌ی A حاوی n بیت است. فرض کنید که A یکی از دو گونه‌ی زیر است: گونه‌ی ۱: نیمی از درایه‌های A حاوی بیت صفر و نیم دیگر حاوی بیت ۱ (بدون ترتیب خاصی) است. گونه‌ی ۲: در مجموع $\frac{2n}{3}$ صفر و $\frac{n}{3}$ یک دارد. (تخصیص هوش مصنوعی ۹۲)

به شما یک آرایه‌ی A داده شده است که با احتمال یکسان یا از گونه‌ی ۱ است یا ۲. در بدترین حالت دست کم چند تا از درایه‌های A را باید بررسی کنید تا گونه‌ی آن را با اطمینان تشخیص دهید؟

$$\frac{5n}{6} + 1 \quad (4)$$

$$\frac{5n}{6} \quad (3)$$

$$\frac{2n}{3} + 1 \quad (2)$$

$$\frac{2n}{3} \quad (1)$$

(علوم کامپیوتر - ۹۴)

-۱۸ کدام عبارت صحیح است؟

$$O(n) < O(\log n!) < O((\log n)!) \quad (2) \quad O((\log n)!) < O(n) < O(\log n!) \quad (1)$$

$$O(\log n!) < O(n) < O((\log n)!) \quad (4) \quad O(\log n!) < O((\log n)!) < O(n) \quad (3)$$

(۹۵ - IT)

-۱۹ چند تا از گزاره‌های زیر درست‌اند؟

$$f(n) = \Theta(n) \wedge g(n) = \Omega(n) \Rightarrow f(n)g(n) = \Omega(n^2) \quad •$$

$$f(n) = \Theta(1) \Rightarrow n^{f(n)} = O(n) \quad •$$

$$f(n) = \Omega(n) \wedge g(n) = O(n^2) \Rightarrow g(n) / f(n) = O(n) \quad •$$

$$f(n) = O(n^2) \wedge g(n) = O(n) \Rightarrow f(g(n)) = O(n^2) \quad •$$

$$f(n) = O(\log n) \Rightarrow n^{f(n)} = O(n) \quad •$$

$$4 \quad (4)$$

$$3 \quad (3)$$

$$2 \quad (2)$$

$$1 \quad (1)$$

-۲۰ اگر زمان اجرای یک الگوریتم با رابطه‌ی $T(n) = T(\log n) + O(n)$ تعریف شود، کدام

(۹۵ - IT) گزینه مرتبه‌ی زمانی این الگوریتم را نشان می‌دهد؟

(تعریف: $\{(log^{(i)} n - log(log^{(i-1)} n) \text{ و } log^* n - min\{i > 0 : log^{(i)} n \leq 1\}\}$)

$$O(\log n) \quad (4) \quad O(n) \quad (3) \quad O(n \log^* n) \quad (2) \quad O(n \log n) \quad (1)$$

(علوم کامپیوتر - ۹۵)

-۲۱ کدام یک از موارد زیر درست است؟

$$n \in O((\log n)^{\log n}) \quad (2)$$

$$n \in \Omega((\log n)^{\log n}) \quad (1)$$

$$n \in \Omega((\log n)!) \quad (4)$$

$$\log n! \in O(n) \quad (3)$$

[پاسخنامه سوال‌های چهارگزینه‌ای فصل اول]

- ۱ گزینه (۴). برای مقایسه $n^x > \lg n$ حال طرفین را به n^{x-1} تقسیم

کنید پس $n > n^{x-1} \lg n$ حال در n ضرب کنید پس

$$\frac{n}{\lg n} = \Omega(n^{x-1}) \quad \text{بنابراین } \frac{n}{\lg n} > n^{x-1}$$

برای مقایسه n^{3n}, n^{2n}, n^{2^n} می‌دانیم $n^{2^n} > n^{3n} > n^{2n}$ حال طرفین را در 2^n ضرب کنید پس

$$.3^n = \Omega(n.2^n)$$

برای مقایسه $n^{1/2}$ و $n^{log_3 n}$ می‌دانیم $n^{log_3 n} > (log_3 n)^5$ حال طرفین را در n ضرب

$$.n.(log_3 n)^5 = O(n^{1/2}) > n.(log_3 n)^5$$

برای مقایسه \sqrt{n} و $(\lg n)^5$ می‌دانیم n^a رشد بیشتری از $(\lg n)^b$ و a ثابت و $b < 0$ دارد، پس

$$\sqrt{n} = \Omega((\lg n)^5)$$

- ۲ گزینه (۱). گزاره ۱ برای تابعی که $f(n) < 1$ نادرست است. مثلاً اگر $f(n) = \frac{1}{n}$ آنگاه

$$\frac{1}{n} = O(\frac{1}{n^2}) \quad \text{غلط است زیرا } \frac{1}{n} \text{ رشد بیشتری از } \frac{1}{n^2} \text{ دارد. (با کمک حد می‌توان اثبات کرد.)}$$

گزاره ۲ صحیح است. $\theta(f(n)) = 0$ یعنی توابع با رشد کمتر از $f(n)$ پس اگر $f(n)$ را با

تابعی که رشد کمتری از $f(n)$ دارد جمع بزنیم، طبق خاصیت ماکزیمم‌گیری همان

$$.n^2 + n = \theta(n^2)$$

می‌شود. مثلاً $\theta(f(n))$

گزاره ۳ نادرست است. مثلاً $2n = O(n)$ و $f(n) = 2n$ فرض کنید.

$$2^n = O(2^n) \quad \text{غلط است زیرا } 2^n = 2^{2^n} = 4^n = 2^{2n}$$

است ولی $2^n = O(2^n)$ نادرست است. مثلاً $4^n = \theta(4^n)$ فرض کنید،

$$4^n = 2^{2^n} = 2^{\frac{n}{2}} = 2^n \quad \text{دارد.}$$

- ۳ گزینه (۴). می‌دانیم $\sqrt{n} = n^{\frac{1}{2}}$ اگر $a < b$ پس $n^a < n^b$ رشد کمتری از n^{ϵ} دارد، بنابراین

$$n.lg n! = O(\sqrt{n}).n^{\epsilon} \quad \text{می‌دانیم } n.lg n! \text{ هم رشد است. و می‌دانیم } n^{\epsilon} \text{ از}$$

$(\lg n)^k$ ، که k ثابت است، رشد بیشتری دارد. پس ترتیب رشد توابع در این تست به شکل

است. لازم به ذکر است که طریقه نوشتن گزینه‌ها نادرست

است. یعنی مثلاً $O(n^\varepsilon) = O(\sqrt{n})$ نادرست است. زیرا به این معنی است که مجموعه $O(n^\varepsilon)$ با مجموعه $O(\sqrt{n})$ برابر است که چنین نیست! ولی از بی دقتی طراح، چشم پوشی می کنیم.

-۴ گزینه (۴). باید فرض کنیم $\varepsilon > 0$. تابع نمایی است که رشد بیشتری از سایر توابع در این تست دارد. می دانیم $n^{1+\varepsilon} > n \lg n$ حال طرفین را در n ضرب کنید و به $n \lg n$ تقسیم کنید پس $n \lg n < \frac{n^{1+\varepsilon}}{\lg n} < n \cdot n^\varepsilon$ بنابراین ترتیب رشد توابع در این تست $n \lg n < n^{1+\varepsilon}$ است.

-۵ گزینه (۱). با توجه به اینکه توابع در این درس باید مثبت باشند پس در گزینه ها $\sin n$ باید داخل قدر مطلق باشد. که از این بی دقتی طراح چشم پوشی می کنیم. یادمان هست که دامنه n در این درس اعداد طبیعی $\{1, 2, 3, \dots\}$ فرض می شود. با این دامنه، $\sin n$ هیچگاه صفر نمی شود و $|\sin n|$ همانند یک عدد ثابت با مقدار کمتر از ۱ می باشد. پس $|\sin n|$ رشد بیشتری از n دارد. اگر فرض کنیم n عدد حقیقی است (در برخی منابع چنین است) آنگاه $\sin n = k\pi$ در نقاط صفر می شود پس $n \in \Omega(n)$ درست $n = k\pi$ نخواهد بود چون نامساوی $n \geq c \cdot n$ بیشمار بار نقض می شود. (در نقاط $\sin n = 0$ صفر می شود).

-۶ گزینه (۱). تابع $(n+1)(n^3 - 2n + 1)$ با n^3 هم رشد است که رشد کمتری از 2^n و n^4 دارد و رشد بیشتری از n و $\lg n$ دارد. پس گزینه ۱ صحیح است.

-۷ گزینه (۴). به ازای هر a و b ثابت که $a > b$ تابع n^a رشد بیشتری از $(\lg n)^b$ دارد. پس $(\lg n)^b = O((\lg n)^a)$ یا $n^a = \Omega((\lg n)^b)$ به ازای هر a, b ثابت که $a > b$ صحیح است. در گزینه ۲ گفته شده که $n^a \neq O((\lg n)^b)$ اگر و فقط اگر ($a > b$) (iff)، که غلط است چون $n^a \neq O((\lg n)^b)$ به ازای هر $a, b > 0$. بنابراین گزینه ۴ صحیح است.

-۸ گزینه (۱). رشد $n^{\frac{1}{n}}$ از $\lg n$ بیشتر است پس گزینه ۱ صحیح است. رشد $\lg n$ با $n \lg n$ یکسان و بیشتر از $\lg n$ است پس گزینه ۲ غلط است. رشد $n^{\frac{1}{n}} = \sqrt[n]{n} = \sqrt[n]{n^{n+1}} = \sqrt[n]{n^n + n^{n-1} + \dots + n^2 + n + 1} < \sqrt[n]{8n^3 + 3n^2 + 4n} = \sqrt[n]{8n^3} = n^{\frac{3}{n}}$ یکسان و بیشتر از $n^{\frac{1}{n}}$ است که گزینه ۴ غلط است.

-۹ گزینه (۱). تابع $2^n + n^2 + 6 \cdot 3^n$ هم رشد با 2^n است که از n^2 بیشتر است. پس گزینه های ۲ و ۳ و ۴ صحیح هستند.

۱۰ - گزینه (۲). $n^{\sqrt{3}} \lg n$ رشد بیشتری از $n^{\sqrt{2}}$ دارد پس گزینه ۱ غلط است. با توجه به اینکه

اگر طرفین را به $\lg n > \frac{n^{\sqrt{2}}}{\lg n}$ تقسیم کنید $n^{\sqrt{3}} > n^{\sqrt{2}}$ پس گزینه ۳ غلط است. تابع

$n^{\sqrt{3}} \cdot n^{\sqrt{2}}$ با $n^{\sqrt{3} + \sqrt{2}}$ هم رشد است و رشد $n^{\sqrt{3} + \sqrt{2}}$ از $n^{\sqrt{3}}$ بیشتر است، پس

گزینه ۴ غلط است. رشد $\frac{6n^{\sqrt{3}}}{\lg n+1}$ از $n^{\sqrt{3}}$ کمتر است پس گزینه ۲ صحیح است.

۱۱ - گزینه (۳). الف) رشد $n!$ از $n^{\sqrt{3}}$ کمتر است پس $O(n^n) = O(n!)$ درست است.

ب) رشد $\frac{n^{\sqrt{2}}}{\lg n}$ از رشد $n^{\sqrt{2}}$ کمتر است پس $\theta(n^{\sqrt{2}}) = \frac{n^{\sqrt{2}}}{\lg n}$ غلط است.

ج) رشد $n^{\sqrt{3}} \lg n$ از رشد $n^{\sqrt{3}}$ بیشتر است پس $\theta(n^{\sqrt{3}}) = \theta(n^{\sqrt{3}} \lg n)$ غلط است.

د) رشد $(n^{\sqrt{3}})^n + 6(2^n)$ با $n^{\sqrt{3}n} + 6(2^n) = \theta(n^{\sqrt{3}n})$ یکسان است پس $n^{\sqrt{3}n} + 6(2^n)$ درست است.

۱۲ - گزینه (۲). I. رشد $e^{c\sqrt{n}}$ از $e^{c\sqrt{n}}$ بیشتر است چون $(e^{\sqrt{n}})^c = e^{c\sqrt{n}}$ که از $e^{c\sqrt{n}}$ بیشتر است. پس $e^{c\sqrt{n}} = O(e^{c\sqrt{n}})$ غلط است.

II. رشد $n^{\sqrt{3}} \lg n$ از $n^{\sqrt{3}}$ بیشتر است پس $O(n \lg n) = O(n^{\sqrt{3}} \lg n)$ غلط است.

III. رشد $n \lg n$ از $n^{\sqrt{3}}$ کمتر است پس $O(n \lg n) = O(n^{\sqrt{3}})$ درست است.

۱۳ - گزینه (۱). توجه کنید $\lg \sqrt{x} = \frac{1}{2} \lg x$ که با $\lg x$ هم رشد است. با توجه به اینکه

$\lg x < x^{\sqrt{3}/2}$ طرفین را در x ضرب کنید پس $x \lg x < x^{\sqrt{3}/2} \cdot x^{\sqrt{3}/2} = x^{\sqrt{3}}$ واضح است که $x^{\sqrt{3}/2} < x^{\sqrt{3}}$ و

$x^{\sqrt{3}/2} < x^{\sqrt{3}/2} \lg \sqrt{x} < x^{\sqrt{3}/2} (3x^{\sqrt{3}/2}) < 16x^{\sqrt{3}}$ پس ترتیب رشد تابع $x \lg x$ است.

۱۴ - گزینه (۴). در گزینه ۱، از $f(n) \leq g(n)$ و $f(n) \simeq g(n)$ میتوان نتیجه گرفت $f(n) \geq g(n)$ پس درست است. از $f(n) \simeq g(n)$ به تنها ی میتوان نتیجه گرفت $f(n) \geq g(n)$

$$(f(n) \geq g(n))$$

در گزینه ۲، از $f(n) \simeq g(n)$ و $f(n) \geq g(n)$ میتوان نتیجه گرفت $f(n) \leq g(n)$ پس درست است.

در گزینه ۳، از $f(n) \simeq g(n)$ و $f(n) \leq g(n)$ نتیجه میشود $f(n) \geq g(n)$ پس درست است.

در گزینه ۴، از $f(n) \simeq g(n)$ و $f(n) \geq g(n)$ نتیجه نمیشود $f(n) \leq g(n)$ پس غلط است.

توجه کنید منظور از $f(n) \leq g(n)$ یعنی رشد $f(n)$ کمتر یا مساوی رشد $g(n)$ است.

- ۱۵ گزینه (۴). هرچهار گزینه اشتباه هستند و تست غلط است. مatasفانه هر سال تست‌های غلط در کنکور طرح می‌شوند که برخی حذف می‌شوند. با توجه به اینکه $f(n) = O(g(n))$ پس $f(n) \leq g(n)$ (منظور رشد است) که نتیجه نمی‌شود $g(n) \approx f(n)$ و پس گزینه ۱ غلط است. برای نقض گزینه ۲ فرض کنید $f(n) = 2n$ و $g(n) = n$. برای نقض گزینه ۳ فرض کنید $f(n) = n$ و $g(n) = 2^n$. برای نقض گزینه ۴ فرض کنید $f(n) = n$ و $g(n) = \lg n$. درست است ولی $f(n) = O(g(n))$ و $\log(f(n)) = \lg n$ که $\log(f(n)) = \lg n$ درست است ولی $f(n) = O(g(n))$ و $n \neq O(\lg n)$.

گزینه ۴ اگر به صورت $\log(f(n)) = O(\log(g(n)))$ می‌بود، درست بود.

- ۱۶ گزینه (۱). گزینه ۱ در صورتی درست است که m ثابت باشد ولی اگر مثلاً فرض کنیم $f_i(n) = i$ آنگاه $m = n$ و تابع $\max_{i=1}^m f_i(n) = \max_{i=1}^n i = O(n)$

$$O\left(\sum_{i=1}^m f_i(n)\right) = O\left(\sum_{i=1}^n i\right) = O\left(\frac{n(n+1)}{2}\right) = O(n^2)$$

$$O(\max_{i=1}^m \{f_i\}) = O(\max_{i=1}^n \{i\}) = O(\max \{1, 2, \dots, n\}) = O(n)$$

پس گزینه ۱ ممکن است غلط باشد.

در گزینه ۲، از $f(n) \leq g(n)$ و $g(n) \geq f(n)$ نتیجه نمی‌شود $f(n) \approx g(n)$ پس گزینه ۲ غلط است.

در گزینه ۳، قسمت I. گفته شده $c, g(n) \leq f(n)$ ، اگر $g(n) \leq c$ را تابع ثابت فرض کنیم آنگاه درست است ولی می‌توان مثلاً $g(n) = \frac{1}{n}$ فرض کرد آنگاه $g(n).f(n) = \theta(f(n))$ غلط است مثلاً اگر $g(n).f(n) = \frac{1}{n}.n^2 = n$ باشد $f(n) = n^2$ که از مرتبه $\theta(n^2)$ نیست. در قسمت II، $g(n).f(n) \geq f(n)$ پس $g(n) \geq c$ درست است.

در گزینه ۴ هر دو جمله I و II صحیح هستند.

کلید مورد نظر طراح در این تست گزینه ۲ می‌باشد.

- ۱۷ گزینه (۴). بدترین حالت وقتی است که خوانده می‌شود صفر باشد، $\frac{n}{3}$ درایه اول که خوانده می‌شود صفر باشد، $\frac{n}{3}$ درایه بعدی ۱ باشد، تاکنون نوع آرایه مشخص نشده است، ولی اگر یک بیت دیگر خوانده شود نوع آرایه معلوم می‌شود. پس حداقل باید $1 + \frac{n}{3} + \frac{n}{3} = \frac{5n}{3}$ درایه خوانده شود.

- ۱۸ گزینه (۲). می‌دانیم $\lg n! \sim n \cdot \lg n$ و همچنین رشد $(\lg n)!$ از رشد چند جمله‌ای‌ها بیشتر و از رشد نمایی‌ها کمتر است.

- ۱۹ گزینه (۲). فقط گزاره‌های اول و سوم صحیح هستند.

۲۰ - گزینه (۳) .

$$T(n) = T(\lg n) + O(n) = \Omega(n)$$

$$T(n) \leq T\left(\frac{n}{2}\right) + O(n) = \theta(n) \Rightarrow T(n) = O(n)$$

$$\therefore T(n) = \theta(n) \text{ پس}$$

۲۱ - گزینه (۲). رشد $(\lg n)^{\lg n}$ از $n \cdot \lg n$ بیشتر است. رشد $(\lg n)!$ از $n \cdot \lg n$ یکی است که از n بیشتر است. رشد $(\lg n)!$ از همه چندجمله‌ای‌ها بیشتر است.

تحلیل الگوریتم‌های غیربازگشتی – آنالیز استهلاکی

فصل ۲

تحلیل الگوریتم‌های غیربازگشتی

در این بخش می‌خواهیم نحوه محاسبه زمان اجرای الگوریتم‌های غیر بازگشتی را بررسی کنیم. زمان اجرا به عوامل مختلفی بستگی دارد از جمله: سخت افزاری که استفاده می‌کنیم، زبان برنامه‌نویسی که الگوریتم را با آن پیاده‌سازی می‌کنیم، اندازه ورودی، ترکیب داده‌های ورودی و ... هدف ما این است که زمان اجرا را بر حسب تعداد باری که دستورات اجرا می‌شوند محاسبه کنیم. به طور پیش فرض همه دستورات را از نظر زمان اجرا مثل هم فرض می‌کنیم (مگر اینکه خلاف آن گفته شود) و زمان هر بار اجرای یک دستور را ۱ فرض می‌کنیم بنابراین زمان اجرا را برابر با تعداد اجرای همه دستورات تعریف می‌کنیم. مثلاً اگر دستورات یک الگوریتم جمعاً ۱۰۰۰ بار اجرا شوند، زمان آن الگوریتم برابر ۱۰۰۰ است.

لازم به ذکر است که ما برای نوشتن کد الگوریتم‌ها از زبان خاصی استفاده نمی‌کنیم و شبه کد می‌نویسیم. گاهی ممکن است از زبان C استفاده کنیم. از علامت = یا ← یا := برای انتساب استفاده می‌کنیم از علامت == یا = برای مقایسه تساوی کمک می‌گیریم. برای نوشتن توضیحات (comment) همانند زبان C از علامت // استفاده می‌کنیم. برای مشخص کردن دستورات یک بلاک مثلاً دستورات یک حلقه از {} استفاده می‌کنیم و یا ممکن است در برخی تست‌ها فقط از فروفتگی استفاده شود و علامت خاصی بکار نرود.

مثال ۱: در تکه برنامه‌های زیر تعداد اجرای همه خطوط را (یعنی زمان اجرا را) بیابید.

- | | |
|---------------------------|---------------------------|
| a) ۱. for($i=1$ to n) | b) ۱. for($i=m$ to n) |
| ۲. write(*) | ۲. write(*) |
| c) ۱. for($i=1$ to n) | d) ۱. for($i=1$ to n) |
| ۲. for($j=1$ to n) | ۲. for($j=1$ to n) |
| ۳. write(*) | ۳. for($k=1$ to n) |
| | ۴. write(*) |

پاسخ: **(a)** دستور $2n$ بار اجرا می‌شود و دستور ۱ یعنی حلقه for خودش $n+1$ بار اجرا می‌شود. چون حلقه for، متغیر i را از ۱ تا n مقدار می‌دهد و write اجرا می‌شود. (تاکنون هر یک n بار اجرا شده‌اند) و در نهایت مقدار n را یک واحد دیگر افزایش می‌دهد و خارج می‌شود. پس خود حلقه از جمله داخلش، یک بار بیشتر اجرا شده است. پس زمان این کد $2n+1$ است.

(b) دستور $n-m+1$ بار و خود دستور ۱، یک بار بیشتر یعنی $n-m+2$ بار اجرا می‌شود. پس این دو دستور جمماً $(n-m)+3$ بار اجرا می‌شوند.

(c) دستور ۱، $n+1$ بار، دستور ۲، $(n+1)n$ بار و دستور ۳، n^3 بار اجرا می‌شوند پس جمماً $2n^3+2n+1$ بار اجرا می‌شوند.

(d) دستور ۱ و ۲ و ۳ و ۴ به ترتیب $n(n+1)$ ، $n(n+1)^2$ و n^3 بار اجرا می‌شوند. پس جمع تعداد اجراهای برابر $2n^3+2n+1$ است.

توجه: در بسیاری از تست‌ها، زمان اجرا به طور کامل مورد نظر نیست و فقط مرتبه زمان اجرا مطلوب است. مثلاً در قسمت (a) مثال ۱، زمان اجرا $2n+1$ است که از مرتبه $\Theta(n)$ می‌باشد. یا قسمت (d) از مرتبه $\Theta(n^3)$ است. برای یافتن مرتبه زمان اجرا، لازم نیست تعداد اجرای همه دستورات محاسبه شود بلکه فقط تعداد اجرای جمله اصلی که در این مثال write است، کافی است.

مثال ۲: در تکه برنامه‌های زیر، دستور write چند بار اجرا می‌شود؟

- | | |
|---------------------------|---------------------------|
| a) ۱. for($i=1$ to n) | b) ۱. for($i=1$ to n) |
| ۲. for($j=1$ to i) | ۲. for($j=1$ to i) |
| ۳. write(*) | ۳. for($k=1$ to j) |
| | ۴. write(*) |

پاسخ: **(a)** حلقه دوم به حلقه اول وابسته است. بار اول، وقتی $i=1$ ، در حلقه دوم j از ۱ تا ۱ تغییر می‌کند و write، یک بار اجرا می‌شود. در دور بعد مقدار $i=2$ و در حلقه دوم، j از ۱ تا ۲ تغییر

می‌کند و `write`, دو بار اجرا می‌شود. به همین ترتیب در نهایت $i = n$ و مقدار j از ۱ تا n تغییر می‌کند و `write`, n بار اجرا می‌شود پس تعداد بار اجرای دستور `write` برابر $\sum_{i=1}^n i = \frac{n(n+1)}{2}$ است. البته می‌توان با استفاده از سیگما نیز تعداد اجرای دستور `write` را یافت. کافی است بجای هر حلقه یک سیگما قرار دهید و بجای `write`, عدد ۱ قرار دهید:

$$\sum_{i=1}^n \sum_{j=1}^i 1 = \sum_{i=1}^n i = \frac{n(n+1)}{2}$$

(b) با کمک سیگما تعداد اجرای `write` را محاسبه می‌کنیم:

$$\begin{aligned} \sum_{i=1}^n \sum_{j=1}^i \sum_{k=1}^j 1 &= \sum_{i=1}^n \sum_{j=1}^i j = \sum_{i=1}^n \frac{i(i+1)}{2} = \frac{1}{2} \left[\sum_{i=1}^n i^2 + \sum_{i=1}^n i \right] \\ &= \frac{1}{2} \left[\frac{n(n+1)(2n+1)}{6} + \frac{n(n+1)}{2} \right] = \frac{n(n+1)(n+2)}{6} = \binom{n+2}{3} = \theta(n^3) \end{aligned}$$

جالب است بدانید که تعداد اجرای دستور `write` در قسمت (b) را با روش‌های شمارشی که در درس ساحتمنان گیسته می‌خوانید نیز می‌توان یافت. هر بار اجرای دستور `write` متناظر است با یک دسته (i, j, k) صحیح که $1 \leq k \leq j \leq i \leq n$ که این نامساوی‌ها معادلند با نامساوی‌های $n+2 \leq k < j+1 < i+2 \leq n+2$ و در واقع می‌خواهیم سه شی متمایز از ۱ تا $n+2$ (یعنی ۳ شی) انتخاب کنیم که $\binom{n+2}{3}$ حالت دارد.

نتیجه: تعداد اجرای دستور `write` در k حلقه وابسته مقابل برابر است با $\binom{n+k-1}{k}$ که از مرتبه $\theta(n^k)$ است.

```
for(i₁ = 1 to n)
  for(i₂ = 1 to i₁)
    for(i₃ = 1 to i₂)
      :
      for(iₖ = 1 to iₖ₋₁)
        write(*)
```

تست ۱: در تکه برنامه زیر دستور write چند بار اجرا می‌شود؟

```
i = 1
while(i < n)
{
    write(*)
    i = i * 2
}
    ⌈lg n⌉ + 1 (۴)      ⌊lg n⌋ + 1 (۳)      ⌈lg n⌉ (۲)      ⌋lg n⌋ (۱)
```

پاسخ: دستور write به ازای مقادیر i برابر $1, 2, 4, \dots, 2^k$ که $n \leq 2^k < 2^{k+1}$ و n می‌شود پس تعداد اجرای دستور write، حدوداً برابر $\lg n$ است. برای یافتن تعداد بار دقیق اجرا، می‌توان به ازای n های مختلف بررسی کرد. مثلاً به ازای $n = 8$ تعداد اجرا ۳ بار است که یا گزینه ۱ یا ۲ صحیح هستند. به ازای $n = 9$ تعداد اجرا ۴ بار است که گزینه ۲ صحیح است.

تست ۲: در تکه برنامه زیر دستور write چند بار اجرا می‌شود؟

```
i = n
while(i > 1)
{
    write(*)
    i = ⌊i / 2⌋
}
    ⌋lg n⌋ (۱)      ⌈lg n⌉ (۲)
    ⌈lg n⌉ + 1 (۳)      ⌋lg n⌋ + 1 (۴)
```

پاسخ: همانند تست قبل تعداد اجرا لگاریتمی است چون i هر بار نصف می‌شود. به ازای $n = 8$ دستور write، ۳ بار و به ازای $n = 9$ نیز ۳ بار اجرا می‌شود. پس گزینه ۱ صحیح است. توجه: در دو تست قبل اگر بجای $i * 2$ یا $\frac{i}{2}$ ، قرار دهیم $k * i$ یا $\frac{i}{k}$ که k ثابت است، آنگاه تعداد اجرا $\log_k n$ می‌باشد که برای یافتن مقدار دقیق می‌توان برای n مقادیر مختلفی قرار داد.

تست ۳: زمان اجرای قطعه کد مقابل، از چه مرتبه‌ای است؟

```
i = 2
while(i ≤ n)
{
    write(*)
    i = i * i
}
    Θ(lgn) (۱)      Θ(lg2n) (۲)
    Θ(√lgn) (۳)      Θ(lg lg n) (۴)
```

پاسخ: دستور write به ازای مقادیر $i = 2, 2^2, 2^4, 2^8, \dots$ اجرا می‌شود در واقع مقادیر i که باعث اجرای دستور write می‌شوند به شکل $(2^0, 2^1, 2^2, \dots, 2^k)$ هستند و تا وقتی write اجرا شود

که $n \leq 2^k$ پس $\lg n \leq k$ در نتیجه $k \leq \lg n$. دقت کنید تعداد اجرای دستور `write`، $k+1$ بار است که k حدوداً $\lg n$ است. پس گزینه ۴ صحیح است.

مثال ۳: زمان اجرای تکه برنامه‌های زیر، ار چه مرتبه‌ای است؟

a) ۱. `for(i = 1; i ≤ n; i++)`
۲. `for(j = 1; j ≤ n; j = j * 2)`
۳. `write(*)`

b) ۱. `for(i = 1; i ≤ n; i++)`
۲. `for(j = 1; j ≤ i; j = j * 2)`
۳. `write(*)`

c) ۱. `for(i = 1; i ≤ n; i = i * 2)`
۲. `for(j = 1; j ≤ n; j++)`
۳. `write(*)`

d) ۱. `for(i = 1; i ≤ n; i = i * 2)`
۲. `for(j = 1; j ≤ i; j++)`
۳. `write(*)`

پاسخ: a) حلقه اول خطی و حلقه دوم لگاریتمی است و حلقه‌ها به هم وابسته نیستند. پس تعداد اجرای دستور `write`, $n \lg n$ است. پس مرتبه $\Theta(n \lg n)$ است.

b) حلقه دوم به حلقه اول وابسته است ($i \leq j$). حلقه دوم به ازای مقادیر j از ۱ تا n که j هر بار در ۲ ضرب می‌شود، اجرا می‌شود. پس تعداد اجرای این حلقه $\lg n$ است. و چون از ۱ تا n و به صورت خطی تغییر می‌کند، پس تعداد اجرای دستور `write` برابر $\sum_{i=1}^n \lg i$ است که برابر است با $\lg 1 + \lg 2 + \dots + \lg n = \lg n!$ پس مرتبه زمان اجرا $\Theta(n \lg n)$ است.

c) همانند قسمت a، چون حلقه‌ها به هم وابسته نیستند، مرتبه $\Theta(n \lg n)$ است.
d) حلقه دوم به حلقه اول وابسته است پس نیاز به بررسی دارد. به ازای هر i ، حلقه دوم دقیقاً i بار اجرا می‌شود. مقادیر i عبارتند از $1, 2, 1, 2^2, 2^3, \dots, 2^k$. پس تعداد اجرای دستور `write` برابر است با:

$$1 + 2 + 2^2 + \dots + 2^k = 2^{k+1} - 1 = \Theta(2^k) = \Theta(n)$$

یعنی مرتبه اجرا بر خلاف ظاهر حلقه‌ها، $\Theta(n)$ یعنی خطی است!

توجه: در برخی الگوریتم‌ها زمان اجرا در حالات مختلف، متفاوت است. مثلاً در مرتبسازی n عنصر اگر آرایه از قبل مرتب باشد، ممکن است زمان کمتری نسبت به حالتی که آرایه مثلاً بر عکس مرتب است، صرف شود. در تحلیل الگوریتم‌ها، سه حالت مطرح می‌شود که عبارتند از: بهترین حالت (Best case)، حالت متوسط (Average case) و بدترین حالت (Worst case). مثلاً فرض کنید می‌خواهیم x را در آرایه n عنصری $[1 \dots n]$ جستجوی خطی کنیم، ابتدا x را با $[1 \dots n]$ مقایسه می‌کنیم اگر مساوی نبود با $[2 \dots n]$ و اگر مساوی نبودند با $[3 \dots n]$ و به همین ترتیب. واضح

است که حداقل تعداد مقایسه ۱ است. پس بهترین حالت زمان اجرا ۱ است ($B(n) = 1$). حداکثر تعداد مقایسه برابر n است. پس بدترین حالت زمان اجرا n است ($W(n) = n$). تحلیل حالت متوسط معمولاً دشوار است و باید بین همه حالات، میانگین گرفته شود (امید ریاضی). در جستجوی خطی اگر فرض کنیم عدد x حتماً در آرایه وجود دارد آنگاه با احتمال $\frac{1}{n}$ عدد x در $c[i] \leq i \leq n$ است بنابراین زمان اجرای حالت متوسط جستجوی خطی عبارت است از:

$$A(n) = 1 \times \frac{1}{n} + 2 \times \frac{1}{n} + \dots + n \times \frac{1}{n} = (1+2+\dots+n) \times \frac{1}{n} = \frac{n+1}{2}$$

دقت کنید برای محاسبه $A(n)$ ، همه حالات x را در نظر گرفته‌ایم: اگر x در $c[i]$ باشد، ۱ مقایسه نیاز است و احتمال آن $\frac{1}{n}$ است. اگر x در $c[2..n]$ باشد، ۲ مقایسه نیاز است و احتمال آن $\frac{1}{n}$ است، و به همین ترتیب اگر x در $c[n]$ باشد، n مقایسه نیاز است و احتمال آن $\frac{1}{n}$ است که با محاسبه $A(n) = \frac{n+1}{2}$ بدست آمد. ما $A(n)$ را با این فرض که x حتماً در آرایه هست یافتیم. حال فرض کنید x با احتمال P در آرایه $c[1..n]$ است پس با احتمال $\frac{P}{n}$ در مکان i است و با احتمال $1 - P$ در آرایه نیست، بنابراین زمان اجرای حالت متوسط برابر است با:

$$\begin{aligned} A(n) &= 1 \times \frac{P}{n} + 2 \times \frac{P}{n} + \dots + n \times \frac{P}{n} + n \times (1-P) \\ &= (1+2+\dots+n) \times \frac{P}{n} + n(1-P) = \frac{(n+1)P}{2} + n(1-P) \end{aligned}$$

توجه کنید برای محاسبه $A(n)$ همه حالت‌ها را در نظر گرفتیم، احتمال وجود x در $c[i..n]$ است و تعداد i تا مقایسه ($1 \leq i \leq n$) برای یافتن x نیاز است و همچنین احتمال نبود x در $c[1..n]$ است و با n مقایسه مشخص می‌شود که x در آرایه نیست.

تست ۴: فرض کنید با احتمال 50% عنصر x در $c[1..n]$ باشد، به طور متوسط چه تعداد عنصر آرایه c برای یافتن x بررسی می‌شوند؟

$$\frac{1}{4} (4) \quad \frac{3}{4} (3) \quad \frac{2}{3} (2) \quad \frac{1}{2} (1)$$

پاسخ: $P = 0.5$ است پس متوسط تعداد مقایسه برای یافتن x برابر است.

$$\frac{(n+1)P}{2} + n(1-P) = \frac{(n+1)}{4} + \frac{n}{2} = \frac{3n+1}{4}$$

یعنی به طور متوسط $\frac{3}{4}$ عناصر باید بررسی شوند که گزینه ۳ صحیح است.

مثال ۴: الگوریتم زیر مرتب‌سازی درجی (insertion sort) است که در فصل مرتب‌سازی کتاب ساختمان داده‌ها با آن آشنا شدید. فرض کنید هر بار که خط آم اجرا می‌شود زمان c_i صرف می‌شود (برخلاف فرضیاتی که تاکنون داشتیم و زمان هر بار اجرای هر دستوری را ۱ واحد فرض می‌کردیم، در این مثال زمان اجراهای را متفاوت فرض کردیم) و c_i ثابت است. همچنین فرض کنید به ازای هر $n, 2, 3, \dots, n$ تعداد اجرای دستور ۴ (while) برابر t_j باشد. در ستون هزینه (cost) زمان هر بار اجرای دستورات نوشته شده است. در ستون تعداد بار (times) تعداد اجرای هر خط نوشته شده است حاصلضرب cost در زمان کل هر خط را نشان می‌دهد. می‌خواهیم زمان اجرا را بیابیم:

تعداد بار (times)	هزینه (cost)	تعداد بار (times)
۱. for ($j = 2$ to n)	C_1	n
۲. {key = $A[j]$ }	C_2	$n - 1$
۳. $i = j - 1$	C_3	$n - 1$
۴. while ($i > 0$) and ($A[i] > key$)	C_4	$\sum_{j=1}^n t_j$
۵. $\{A[i + 1] = A[i]\}$	C_5	$\sum_{j=1}^n (t_j - 1)$
۶. $i = i - 1\}$	C_6	$\sum_{j=1}^n (t_j - 1)$
۷. $A[i + 1] = key\}$	C_7	$n - 1$

توجه کنید دستورات داخل حلقه while (دستورات ۵ و ۶) همیشه ۱ بار کمتر از خود حلقه while (دستور ۴) اجرا می‌شوند. برای محاسبه زمان اجرا یعنی $T(n)$ باید هزینه‌ها را در تعداد بار ضرب کرده و مقادیر بدست آمده را جمع کنیم:

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

حال می‌خواهیم زمان اجرای $T(n)$ را در حالات مختلف بررسی کنیم. بهترین حالت وقتی است که آرایه A از قبیل مرتب صعودی باشد آنگاه شرط $A[i] > key$ در حلقه while همواره غلط است. و دستور while هر دور فقط یک بار اجرا می‌شود و دستورات داخل حلقه while یعنی دستورات ۵ و ۶ اصلاً اجرا نمی‌شوند، یعنی $t_j = 1$ پس زمان اجرای بهترین حالت برابر است با:

$$\begin{aligned}
 B(n) &= c_1n + c_2(n-1) + c_3(n-1) + c_4 \sum_{j=2}^n 1 + \dots + c_7(n-1) \\
 &= c_1n + c_2(n-1) + c_3(n-1) + c_4(n-1) + c_7(n-1) \\
 &= (c_1 + c_2 + c_3 + c_4 + c_7)n - (c_2 + c_3 + c_4 + c_7) = \theta(n)
 \end{aligned}$$

بدترین حالت وقتی است که آرایه نزولی مرتب باشد، در این صورت هر بار $A[j]$ با همه عناصر قبل خودش معنی با عناصر $[1\dots j-1]$ مقایسه می‌شود پس $T_j = \frac{j}{2}$ بنابراین زمان اجرای بدترین حالت برابر است با:

$$\begin{aligned}
 W(n) &= c_1n + c_2(n-1) + c_3(n-1) + c_4 \left(\frac{n(n+1)}{2} - 1 \right) + c_5 \frac{n(n-1)}{2} + c_6 \frac{n(n-1)}{2} \\
 &\quad + c_7(n-1) = \theta(n^2)
 \end{aligned}$$

دقت کنید $W(n)$ تابعی درجه ۲ بر حسب n است پس $\theta(n^2)$ است. در حالت متوسط

می‌توان فرض کرد $A[j]$ از نصف عناصر قبل خودش کوچکتر است که در این صورت $T_j = \frac{j}{2}$ می‌شود و زمان اجرای حالت متوسط برابر است با:

$$\begin{aligned}
 A(n) &= c_1n + c_2(n-1) + c_3(n-1) + \frac{1}{2}c_4 \left(\frac{n(n+1)}{2} - 1 \right) \\
 &\quad + \frac{1}{2}c_5 \left(\frac{n(n-1)}{2} \right) + \frac{1}{2}c_6 \left(\frac{n(n-1)}{2} \right) + c_7(n-1) = \theta(n^2)
 \end{aligned}$$

آنالیز استهلاکی (Amortized Analysis)

در این نوع آنالیز، n عمل روی یک ساختمان داده انجام می‌شود و هزینه زمانی این عملیات روی n عمل سرشکن می‌شود. در واقع ممکن است عملیات انجام شده هزینه‌های مختلفی داشته باشند و برخی هزینه زیادی داشته باشند ولی هزینه متوسط هر عمل کوچک باشد. لازم به ذکر است که آنالیز استهلاکی با آنالیز در حالت میانگین متفاوت است و روش‌های آماری را شامل نمی‌شود. در واقع آنالیز استهلاکی زمان اجرای متوسط هر عمل در بدترین حالت را نشان می‌دهد.

برای این نوع آنالیز ۳ روش مطرح است:

- آنالیز تجمعی (Aggregate Analysis)

- روش حسابرسی (Accounting method)

- روش پتانسیل (Potential Method)

آنالیز تجمعی

در این روش مجموع هزینه n عمل را به دست می‌آوریم سپس به n تقسیم می‌کنیم. مثلاً n عمل push و pop روی پشته S که در ابتداء خالی است انجام داده‌ایم k عنصر از پشته pop می‌کند، البته اگر کمتر از k عنصر در پشته باشند، همه عناصر pop می‌شوند) درست است که هزینه (k) multipop ممکن است $O(k)$ باشد ولی n عمل از عملیات فوق روی هم هزینه $O(n)$ دارند پس هزینه هر عمل به صورت سرشکن شده برابر $O(1)$ است.

مثال ۵: فرض کنید آرایه $A[0..k-1]$ شامل یک عدد باینری است که $A[0]$ بیت کم ارزش است

$$\text{بنابراین اگر } X \text{ مقدار عدد باشد: } X = \sum_{i=0}^{k-1} A[i] \cdot 2^i$$

ابتدا $X = 0$ یعنی عناصر A همگی صفر هستند، می‌خواهیم X را با عدد یک جمع کنیم الگوریتم به صورت مقابل است:

$\text{INC}(A)$ $\{$ $i = 0;$ $\text{while } (i < k \text{ and } A[i] == 1)$ $\quad \{A[i] = 0; i = i + 1;\}$ $\text{if } (i < k) \quad A[i] = 1;$	می‌خواهیم هزینه زمانی n عمل INC را به دست آوریم. هزینه هر تغییر بیت را یک فرض کنید. با اولین عمل INC، یک تغییر بیت داریم ($A[0]$) با دومین عمل، دو تغییر بیت ($A[0], A[1]$) با سومین عمل، یک تغییر بیت ($A[0]$) و ...
---	---

ولی می‌توان بهتر به مسئله نگاه کرد. بیت $A[0]$ با هر بار اجرای عمل INC، تغییر می‌کند $\left\lfloor \frac{n}{2} \right\rfloor$ بار تغییر می‌کند. $\left\lfloor \frac{n}{4} \right\rfloor$ بار و به طور کلی $\left\lfloor \frac{n}{2^i} \right\rfloor$ بار تغییر می‌کند که با اولین عمل INC، یک تغییر بیت داریم ($A[0]$) با دومین عمل، دو تغییر بیت ($A[0], A[1]$) با سومین عمل، یک تغییر بیت ($A[0]$) و ... تغییر بیت‌ها:

$$\sum_{i=0}^{\lfloor \lg n \rfloor} \left\lfloor \frac{n}{2^i} \right\rfloor < n \sum_{i=0}^{\infty} \frac{1}{2^i} = 2n$$

پس در بدترین حالت هزینه n عمل INC $2n$ است. پس متوسط هزینه هر عمل یعنی هزینه مستهلك شده به ازای هر عمل $O(1)$ است.

تمرین: نشان دهید اگر عمل DEC نیز وجود داشته باشد، آنگاه n عمل می‌توانند هزینه‌ای معادل $\theta(nk)$ داشته باشند.

مثال ۶: دنباله‌ای از n عمل روی یک ساختمان داده اجرا شده است. هزینه عمل i ام برابر i است. اگر i توانی از ۲ باشد و در غیر این صورت هزینه عمل برابر ۱ است. هزینه استهلاکی به ازای هر عمل را بیابید.

پاسخ:

	هزینه عمل i ام										...
operation	۱	۲	۳	۴	۵	۶	۷	۸	۹	۱۰	...
cost	۱	۲	۱	۴	۱	۱	۱	۸	۱	۱	...
$\sum_{i=1}^n C_i \leq n + \sum_{j=0}^{\lg n} 2^j = n + (2n - 1) < 3n$											
$\frac{\text{هزینه کل}}{\text{تعداد عملیات}} = \frac{3n}{n} = 3$											

روش حسابرسی

در این روش به عملیات مختلف، شارژهای مختلفی نسبت می‌دهیم که ممکن است شارژی که نسبت می‌دهیم، از هزینه واقعی عمل کمتر یا بیشتر باشد. مقداری که یک عمل شارژ می‌شود را «هزینه استهلاک» آن عمل گویند.

اگر «هزینه استهلاک» یک عمل از «هزینه واقعی» آن عمل بیشتر باشد، تفاوت این دو به عنوان «اعتبار» به عناصر خاصی از ساختمان داده تخصیص می‌یابد. این اعتبار را می‌توان بعداً برای عملیاتی که هزینه استهلاکشان کمتر از هزینه واقعی‌شان است، صرف کرد.

به عنوان مثال عملیات پشته را در نظر بگیرید. می‌دانیم هزینه واقعی عملیات عبارت است از:

$\text{push} = 1$, $\text{pop} = 1$, $\text{Multipop}(k) = \min(k, s)$

که s تعداد عناصر موجود در پشته هنگام فرآخوانی Multipop است.

ولی ما هزینه استهلاکی زیر را به عملیات نسبت می‌دهیم:

$\text{push} = 2$, $\text{pop} = 0$, $\text{Multipop} = 0$

با این مقادیر، می‌توان هر دنباله از عملیات را پشتیبانی کرد. هر push که انجام می‌شود، هزینه واقعی‌اش ۱ است، پس از شارژ ۲ مقداری آن، ۱ مقدار برای هزینه‌اش صرف می‌شود و ۱ مقدار در عنصر push شده به عنوان اعتبار ذخیره می‌شود، پس تمام عناصری که در پشته هستند، اعتبار ۱ مقداری دارند. حال اگر pop انجام شود، چون هزینه واقعی pop، ۱ است، این ۱ را از اعتبار عنصری که pop می‌شود، دریافت می‌کند. پس از آنجایی که هزینه استهلاکی push و pop ثابت ((O(1)) است پس هزینه n عمل، O(n) است.

به عنوان مثالی دیگر، عمل INC (جمع عدد k بیتی با ۱) را در نظر بگیرید. هزینه واقعی INC برابر تعداد بیت‌هایی است که عوض می‌شود.

حال فرض کنید هزینه استهلاکی تغییر صفر به یک را ۲ واحد در نظر بگیریم. و برای تغییر یک به صفر، هزینه استهلاکی صفر. وقتی بیتی ۱ می‌شود، ۱ واحد از شارژش را صرف هزینه واقعی اش می‌کنیم و ۱ واحد را در بیت ذخیره می‌کنیم، بنابراین هر ۱ در عدد دارای یک اعتبار ۱ واحدی است که می‌توان آن را وقتی صرف کرد که بیت مربوطه صفر شود. از آنجایی که هر عمل INC حداقل یک بیت را ۱ می‌کند پس هزینه استهلاکی هر عمل INC برابر ۲ واحد است، پس هر عمل INC دارای هزینه استهلاکی $O(1)$ است پس n عمل INC دارای هزینه استهلاکی $O(n)$ است.

توجه: تخصیص هزینه استهلاکی به هر عمل، دقت می‌خواهد. اگر C_i هزینه واقعی عمل i ام و

$$\sum_{i=1}^n \hat{C}_i \geq \sum_{i=1}^n C_i$$

روش پتانسیل

روش قبلی، اعتبار را به عنصر تخصیص می‌داد ولی این روش، اعتبار را به صورت پتانسیل به کل ساختمان داده تخصیص می‌دهد.

فرض کنید D_i وضعیت اولیه ساختمان داده‌ای است که می‌خواهیم n عمل روی آن انجام دهیم. C_i هزینه واقعی عمل i ام، D_i وضعیت ساختمان داده است پس از انجام عمل i ام روی D_{i-1} . تابع پتانسیلی تعریف می‌کنیم به اسم ϕ که به هر وضعیت ساختمان داده، یک عدد حقیقی نسبت می‌دهد:

$\phi(D_i)$ یک عدد حقیقی

بنابراین هزینه استهلاکی (\hat{C}_i) عمل i ام برابر است با:

$$\hat{C}_i = C_i + \phi(D_i) - \phi(D_{i-1})$$

پس کل هزینه استهلاکی برابر است با:

$$\sum_{i=1}^n \hat{C}_i = \sum_{i=1}^n (C_i + \phi(D_i) - \phi(D_{i-1})) = \sum_{i=1}^n C_i + \phi(D_n) - \phi(D_0)$$

مثلاً فرض کنید در عملیات پشته (Multipop , pop , push)، تابع پتانسیل ϕ را برابر عناصر موجود در پشته فرض کنیم. در حالت پشته خالی D_0 ، که حالت شروع است $\phi(D_0) = 0$. از آنجایی که تعداد عناصر موجود در پشته عددی نامنفی است پس همیشه $\phi(D_i) \geq 0 = \phi(D_0)$. پس هزینه کل مستهلك شده برای n عمل با توجه به تابع ϕ ، یک کران بالا برای هزینه واقعی است. حال هزینه مستهلك شده را برای push و pop به دست می‌آوریم.

اگر عمل i ام روی پشته push باشد و در حال حاضر تعداد s عنصر در پشته باشد آنگاه:

$$\phi(D_i) - \phi(D_{i-1}) = (s+1) - s = 1$$

بنابراین هزینه مستهلک شده push برابر است با:

$$\hat{c}_i = c_i + \phi(D_i) - \phi(D_{i-1}) = 1 + 1 = 2$$

به همین ترتیب نشان دهید که هزینه مستهلک شده pop و Multipop برابر صفر است. پس هزینه مستهلک شده هر عمل $O()$ است. بنابراین از آنجایی که هزینه مستهلک شده کران بالای هزینه واقعی است پس هزینه واقعی n عمل برابر $O(n)$ است.

خلاصه فصل

- ۱- هدف ما، محاسبه زمان اجرای الگوریتم‌هاست. زمان اجرای یک کد را برابر تعداد اجرای همه دستورات آن کد تعریف می‌کنیم. و از آنجایی که معمولاً مرتبه زمان اجرا برای ما اهمیت دارد، باید تعداد اجرای جمله اصلی را در یک کد محاسبه کنیم.
- ۲- در برخی الگوریتم‌ها زمان اجرا در حالات مختلف، عوض می‌شود. سه حالت بهترین، متوسط و بدترین مطرح می‌شود. مثلاً در جستجوی خطی در $A[1..n]$ بهترین حالت این است که عنصر مورد جستجو با یک مقایسه پیدا شود. بدترین حالت این است که عنصر مورد جستجو، عنصر آخر آرایه باشد یا اصلاً در آرایه نباشد که نیاز به n مقایسه دارد. برای محاسبه حالت متوسط نیز باید امید ریاضی محاسبه کنیم.
- ۳- در آنالیز استهلاکی (سرشکنی)، دنباله‌ای از عملیات را روی یک ساختمان داده انجام می‌دهیم و می‌خواهیم بررسی کنیم متوسط یا میانگین زمان اجرای این عملیات در بدترین حالت چقدر است. برای این منظور سه تکنیک وجود دارد که ساده‌ترین تکنیک، آنالیز تجمعی است که زمان اجرای همه عملیات را در بدترین حالت، محاسبه کرده و جمع می‌کنیم و نتیجه را به تعداد عملیات انجام شده تقسیم می‌کنیم.

تمرین

۱- کوچکترین مقدار n که به ازای آن الگوریتمی که زمانش $100n^3$ میباشد از الگوریتمی که زمانش 2^n میباشد، سریعتر است را بیابید.

۲- زمان اجرای تکه برنامه‌های زیر از چه مرتبه‌ای است؟

a) `for(i = 1 to n)`

```
    for(j = 1 to i)
        if(j mod i == 0)
            for(k = 1 to j)
                write(*)
```

b) `for(i = 1 to n)`

```
    if(odd(i))   for(j = i to n)  x = x + 1
    else         for(j = 1 to i)  y = y + 1
```

c) `i = 3`

```
    if(n == 2 or n == 3) return true
    if(n mod 2 == 0) return false
    while (i <= n)
        if(n mod i == 0) return false else i = i + 2
    return true
```

〔 سؤال‌های چهارگزینه‌ای فصل دوم 〕

-۱ در یک الگوریتم، چند مرحله اول از پیچیدگی $O(n)$ ، چند مرحله بعدی از پیچیدگی $O(n^4)$ و چند مرحله آخر از پیچیدگی $O(n^3)$ است. پیچیدگی کل الگوریتم چقدر است؟
 (علوم کامپیوتر - ۸۶)

$$O(n^7) \quad (4)$$

$$O(n^4) \quad (3)$$

$$O(n^3) \quad (2)$$

$$O(n) \quad (1)$$

-۲ کدام گزینه تعداد مراحل برنامه زیر را به درستی بیان می‌کند؟ (ساختمان دادها - دولتی ۸۳)

```
void sum (int m,int n,float S[ ][ ]) {  
    int i,j;  
    for(j=0;j<m;j++) {  
        S[n-1][j]=0;  
        for (i=0;i<n-1;i++)  
            S[n-1][j]+=S[i][j];  
    }  
}
```

(مبانی نرم‌افزار - آزاد ۷۷)

-۳ پیچیدگی زمانی قطعه برنامه زیر چیست؟

```
for i:=1 to n do  
    for j:=1 to i do  
        for k:=1 to n do  
            x:=x+1;
```

$$O(n^2) \quad (1)$$

$$O(n^3) \quad (2)$$

$$O(2^n) \quad (3)$$

$$O(n^2 \lg n) \quad (4)$$

-۴ مرتبه بزرگی (Order of magnitude) قطعه کد زیر چیست؟ (تخصصی نرم‌افزار آزاد - ۸۰)

```
k:=0;  
for i:=1 to n do  
begin  
    for j:=1 to m do  
        k:=k+1;  
    j:=1;  
    while j< n do  
    begin  
        k:=k+1;  
        j:=2j;  
    end  
end;
```

$$\Theta(n^2) \quad (1)$$

$$\Theta(nm) \quad (2)$$

$$\Theta(nm + n^2) \quad (3)$$

$$\Theta(nm + n \log n) \quad (4)$$

(تخصصی هوش مصنوعی - آزاد ۸۲)

-۵ درجه الگوریتم زیر چیست؟

```
for i ← 0 to n do
{j ← i ; while j ≠ 0 do j ← j div 2}
```

$$n^{\frac{1}{4}} \quad n + \log n \quad n \log n \quad n^{\frac{1}{2}}$$

(ساختمان گسسته مهندسی کامپیوتر - آزاد ۸۵)

-۶ مرتبه زمانی قطعه کد زیر چیست؟

```
i := 2
while i <= n do
begin
    i := i^2
    x := x + 1
end
```

$$\theta(\log n) \quad \theta(\log(\log n)) \quad \theta(n) \quad \theta(n \log n)$$

(مهندسی فناوری اطلاعات - آزاد ۸۳)

-۷ مرتبه زمان قطعه کد زیر چیست؟

```
j := n
while j ≥ 1 do
begin
    for i := 1 to n do
        x := x + 1
    j := ⌊ j / 2 ⌋
end
```

$$\theta(n \log n) \quad \theta(\log n) \quad \theta(n) \quad \theta(n^{\frac{1}{2}})$$

(تخصصی نرم افزار - آزاد ۸۰)

```
for i := 1 to n do
    for j := i to n do
begin
    k := j; r := 1;
    while (r < k) do
begin
    x := x + 1;
    r := r + 1
end
end;
```

$$\frac{n^{\frac{1}{4}} - 9n^{\frac{1}{2}} - 8n}{6} \quad \frac{n^{\frac{1}{4}} - 2n + 1}{4} \quad \frac{n^{\frac{1}{3}} - n}{3} \quad \frac{n^{\frac{1}{2}} - 11n^{\frac{1}{4}} + 9n}{6}$$

(علوم کامپیوتر - ۷۹)

- ۹ مقدار محاسبه شده برای TOTAL را تعیین کنید.

```

TOTAL := 0;
for i := 1 to N do
begin
k := N;
while(k <> 1) do
begin
k := k DIV 2;
TOTAL := TOTAL + 1
end;
end;

```

(علوم کامپیوتر - ۸۰)

- ۱ for(k = 0; k <= n - 1; k++)
- ۲ for(i = 1; i <= n - k; i++)
- ۳ a[i][i + k] = k;

$$N(\log_2^N - 1) \quad (1)$$

$$N \log_2^N \quad (2)$$

$$N(\log_2^N + 1) \quad (3)$$

$$\log_2^N + 1 \quad (4)$$

- ۱۰ در برنامه زیر تعداد دفعات تکرار دستور العمل شماره ۳ کدام است؟

$$\frac{n(n-1)}{2} \quad (4)$$

$$\frac{n^2}{2} \quad (3)$$

$$\frac{n(n+1)}{2} \quad (2)$$

$$n^2 \quad (1)$$

(علوم کامپیوتر - ۸۱)

- ۱۱ پیچیدگی زمان الگوریتم زیر کدام است؟

```

sum = 0
for(i = 0; i < n ; i++)
for(j = 0; j < i; j++)
for(k = 0; k < 3; k++)
sum++;

```

$$O(n^3) \quad (1)$$

$$O(n) \quad (2)$$

$$O(n \log n) \quad (3)$$

$$O(n^2) \quad (4)$$

(علوم کامپیوتر - ۸۵)

- ۱۲ مرتبه زمانی الگوریتم زیر چیست؟

```

for(i = 1; i <= n; i = i + 1)
for(j = 1; j <= n ; j = j + i)
x = x + 1;

```

$$\theta(n \log n) \quad (4)$$

$$\theta(n^3) \quad (3)$$

$$\theta(n^2) \quad (2)$$

$$\theta(n) \quad (1)$$

(۸۳ IT)

- ۱۳ میزان زمان لازم برای اجرای قطعه برنامه زیر چگونه برآورد می‌شود؟

```

for i ← 1 to n
for j ← n downto i
for k ← 1 to n
sum ← sum + A

```

$$O(n^3) \quad (1)$$

$$\theta(n^3) \quad (2)$$

$$\omega(n^4) \quad (3)$$

$$\theta(n^4) \quad (4)$$

۱۴ - مرتبه زمانی شبه کد زیر چیست؟ (۸۶ IT)

```
for(i=۱;i <= n;i++)
  for(j=۱;j <= n;j++)
  {
    x++;
    n--;
  }
```

n (۱)
 n^2 (۲)
 $\log n$ (۳)
 $n \log n$ (۴)

۱۵ - مرتبه زمانی شبه کد زیر چیست؟ (۸۴ IT)

```
for(i=۱;i <= n;i++)
{
  for(j=۱;j <= n;j++)
  {
    x++;
    n--;
  }
}
```

n (۱)
 n^2 (۲)
 $\log n$ (۳)
 $n \log n$ (۴)

۱۶ - پس از اجرای قطعه کد زیر، مقدار نهایی x چه مقدار خواهد بود؟ (۸۵ IT)

```
x = ۰;
for(i=۱;i <= n;i++)
{
  for(j=۱;j <= n;j++) x++;
  j = ۱;
  while(j < n)
  {
    x++; j = j * ۲;
  }
}
```

$n \lceil \log_2 n \rceil$ (۱)
 $n^2 + \lceil \log_2 n \rceil$ (۲)
 $n^2 + n \lceil \log_2 n \rceil$ (۳)
 $n(1 + \lfloor \log_2 n \rfloor)$ (۴)

۱۷ - زمانی مصرفی قطعه برنامه زیر در بهترین و بدترین حالت چه مقداری است و مربوط به چه

مواردی میباشد؟

```
for i ← ۲ to n do
  if k mod i = ۰ then
    for j ← i to n do
      l ← l * k
```

۱) در بهترین حالت زمان خطی است و این مربوط به ورودی است که k به n بخشیدن باشد و

بدترین حالت زمانی است که k به کلیه اعداد ۱ تا n بخشیدن باشد در اینصورت زمان $2n$ است.

۲) در بهترین حالت زمان ضریب ثابتی از n است و این حالت مربوط به مواردی است که k

مضربی از n باشد و بدترین زمان n^2 است که مربوط به باقی حالات میشود.

۳) در هر حال و در کلیه موارد زمان مصرفی n^2 است.

۴) اگر k ضریبی از $n!$ باشد در بدترین حالت قرار میگیریم که زمان مصرفی مربوطه n^2

است و بهترین حالت، زمانی است که k به هیچکدام از اعداد ۲ تا n بخشیدن باشد در

این صورت زمان مصرفی خطی است.

فصل ۲ . تحلیل الگوریتم‌های غیربازگشتی – آنالیز استهلاکی

۵۳

۱۸- پیچیدگی زمان اجرای حلقه زیر چه می‌باشد؟ (علوم کامپیوتر - ۸۴)

$$\text{while } (n > 0) \left\{ n = \frac{n}{10}; \right.$$

$O(\log n)$

$O(n)$

$O\left(\frac{n}{10}\right)$

$O(1)$

۱۹- شمار فراوانی کد زیر در بدترین حالت چیست؟ (مهندسی فناوری اطلاعات - آزاد ۸۵)

$i := 1; j := n;$

repeat

$$k := \frac{(i + j)}{2}$$

if $a[k] \leq x$ then $i := k + 1$
else $j := k - 1$

until $i > j$

$\lceil \log(n+1) \rceil$

$2 + 3 \lceil \log(n+1) \rceil$

$\log n$

$2 + 4 \lceil \log(n+1) \rceil$

۲۰- در تکه برنامه زیر processa چند بار اجرا می‌شود؟ (علوم کامپیوتر - ۸۴)

```
for (int i = 0; i < N - i; ++i)
    for (int i = 0; i < N - i; ++i)
        /* Process a */
```

$\frac{(N+1)^2}{4}$

$\frac{N+1}{2}$

$\frac{N^2}{4}$

$\frac{n}{2}$

۲۱- یک آرایه از اعداد صحیح به صورت $A[1..m]$ مفروض است، بطوری که $\sum_{i=1}^m A[i] = S$ می‌باشد. در این صورت درجه اجرای الگوریتم زیر کدامیک از گزینه‌ها است؟ (تخصصی هوش مصنوعی آزاد ۸۱)

در این صورت درجه اجرای الگوریتم زیر کدامیک از گزینه‌ها است؟

T := 0;

$O(s^2)$

for i := 1 to m do

$O(m + s)$

 for j := 1 to A[i] do

$O(m^2)$

 T := T + 1;

$O(ms)$

۲۲- کامپیوتري در واحد زمان مساله‌ای به اندازه ۱۶ را که الگوریتم آن از مرتبه زمانی n^{2^n} است حل می‌کند. اگر سرعت کامپیوت ۱۳۱۰۷۲ برابر گردد این کامپیوت همان مساله را با چه اندازه‌ای در واحد زمان حل خواهد کرد؟ (ساختمن داده‌ها - مهندسی کامپیوت ۸۶)

$16 * 17 * \log 17$

$16 + 17 + \log 17$

$16 + \log 131072$

32

(۳)

- ۲۳- فرض کنید که میانگین زمان اجرای یک الگوریتم تصادفی A بروی ورودی‌های به اندازه‌ی n برابر $\Theta(n^{\gamma})$ است. چند تا از گزاره‌های زیر درست هستند؟ (نرم‌افزار ۹۲)

ممکن است داده‌ی ورودی‌ای باشد که A آن را در زمان $\Omega(n^{3\gamma})$ حل کند.

ممکن است داده‌ی ورودی‌ای باشد که A آن را در زمان $\Theta(n^{\gamma})$ حل کند.

ممکن است داده‌ی ورودی‌ای باشد که A آن را در زمان $O(1)$ حل کند.

(۱) \circ (۲) \square (۳) \times (۴) \times

- ۲۴- دنباله‌ای از 2^n عمل بروی داده ساختاری انجام می‌شود. هزینه‌ی عمل آم برابر ۱ است. اگر ۱ توانی از ۲ باشد، و گرنه برابر ۱ است. میانگین هزینه یک عمل دلخواه (یعنی مجموع هزینه‌ها تقسیم بر تعدادشان) به کدام گزینه نزدیک‌تر است؟ (تخصصی نرم‌افزار ۸۹)

$2n$ (۱) \times (۲) \times (۳) \times (۴) \circ

- ۲۵- آرایه‌ی A به طول n داده شده است. درایه‌های A در ابتدا صفرند. عمل درج روبرو را n بار بر روی A انجام می‌دهیم. هزینه‌ی سرشکنی هر عمل درج (یعنی مجموع هزینه‌ی n عمل درج تقسیم بر n) کدام است؟ بهترین گزینه را انتخاب کنید. (نرم‌افزار ۹۱)

INSERT(x)	$O(1)$
۱ $n \leftarrow n + 1$	$O(\lg n)$
۲ $t \leftarrow x$	$O(n)$
۳ for $i \leftarrow 0$ to $\lfloor \lg n \rfloor$	
۴ do if $A[i] \neq 0$	$O(\lg^{\gamma} n)$
۵ then $t \leftarrow t + A[i]$	
۶ $A[i] \leftarrow 0$	
۷ else $A[i] \leftarrow t$	
۸ stop and return	

- ۲۶- هزینه‌ی زمانی تکه برنامه‌ی زیر کدام است؟ (۹۱ IT)

int $i = n;$	$O(\lg^{\gamma} n)$	$O(\lg n)$
while ($i > 1$) {		
$i /= 2;$	$O(n^{\gamma})$	$O(n)$
$j = i;$		
while ($j > 1$)		
$j /= 2;$		
}		

(تخصصی نرم افزار دولتی ۸۳)

- ۲۷ - رویه زیر را در نظر بگیرید:

این الگوریتم قرار است مقدار x^k را محاسبه کند. مقدار مستقل از حلقه (Loop invariant)

این الگوریتم چیست؟ (یعنی چه عبارتی همواره در ابتدای حلقه درست است؟)

function power (x , k)

$$y^i a = x^k \quad (1)$$

$y \leftarrow x$

$$y^i a^i = x^k \quad (2)$$

$i \leftarrow k$

$$a(ay)^i = x^k \quad (3)$$

$a \leftarrow 1$

$$y(ay)^i = x^k \quad (4)$$

while $i > 0$

do if i is odd

then $a \leftarrow a \times y$

$y \leftarrow y \times y$

$$i \leftarrow \left\lfloor \frac{i}{2} \right\rfloor$$

return a

- ۲۸ - یک شمارنده k بیتی در ابتدای مقدار صفر دارد. این شمارنده را 2^k بار و هر بار به مقدار ۱

واحد افزایش می‌دهیم. مجموع تعداد تغییرات بیت‌های این شمارنده (از ۰ به ۱ و برعکس

(تخصصی هوش مصنوعی دولتی ۸۵)

چند تاست؟)

$$\theta(n^2) \quad (2)$$

$$\sum_{i=1}^k \frac{n}{2^i} \quad (1)$$

$$\theta(n \log n) \quad (4)$$

$$\sum_{i=1}^k \frac{n}{2^{i-1}} \quad (3)$$

- ۲۹ - الگوریتم زیر را در نظر بگیرید:

$y \leftarrow 0$

$i \leftarrow n$

while $i \geq 0$ do

$y \leftarrow a_i + x \cdot y$

$i \leftarrow i-1$

(علوم کامپیوتر – ۸۲)

مقدار y در شروع هر عبارت while برابر است با:

$$y = \sum_{k=0}^i a_k x^k \quad (2)$$

$$y = \sum_{k=0}^{n-i} a_k x^k \quad (1)$$

$$y = \sum_{k=0}^n a_k x^k \quad (4)$$

$$y = \sum_{k=0}^{n-(i+1)} a_{k+i+1} x^k \quad (3)$$

۳۰ - n نفر با ترتیب تصادفی به صف وارد یک باجه می‌شوند. مسئول باجه قرار است اندازه‌ی قد بلندقدترین این افراد را به دست آورد. مسئول باجه یک برگه دارد که در ابتدا بر روی آن صفر نوشته است. او قد هر فرد که وارد باجه می‌شود را اندازه‌گیری می‌کند و اگر این قد از عدد نوشته شده بر روی برگه بزرگ‌تر باشد، عدد برگه را خط می‌زند و به جای قد فرد ورودی را می‌نویسد. به این کار او عمل «جایه‌جایی» می‌گوییم. اگر هر نفر با احتمال یکسان بتواند بلندقدترین فرد باشد، در انتها تعداد میانگین عمل جایه‌جایی چند تاست؟ بهترین پاسخ را برای n بزرگ انتخاب کنید.

(۹۵ - IT)

$$\ln n$$

$$(n-1)/2$$

$$n/2$$

$$\log_2 n$$

[پاسخنامه سوال‌های چهارگزینه‌ای فصل دوم]

- ۱ گزینه (۳). اگر چندین مرحله پشت سرهم و نه تو باشند، زمان اجرای آنها با هم جمع می‌شود. و طبق قضیه ماکزیمم‌گیری زمان اجرا، متناظر با زمان اجرای کندترین مرحله است:

$$O(n + n^4 + n^2) = O(n^4)$$

- ۲ گزینه (۳). تعریف متغیر، دستور اجرایی نیست به شرطی که مقدار دهی اولیه نشده باشد. جدول زیر تعداد اجرای هر دستور را نشان می‌دهد.

دقت کنید حلقه‌ای که از m تا n شامل m و n باشد، $n - m + 2$ و داخل حلقه $n - m + 1$ بار اجرا می‌شوند.

دستور	تعداد بار اجرا
for($j = \dots$)	$m + 1$
$s[n - 1][j] = \dots$	m
for($i = \dots$)	$m(n)$
$s[n - 1][j] += s[i][j]$	$m(n - 1)$
مجموع هم دستورات	
	$m(2n + 1) + 1$

- ۳ گزینه (۳). تعداد اجرای دستور $x := x + 1$ برابر است با:

$$\sum_{i=1}^n \sum_{j=1}^i \sum_{k=1}^j 1 = \sum_{i=1}^n i = n \sum_{i=1}^n i = n \sum_{i=1}^n i = n \frac{n(n+1)}{2} = \Theta(n^3)$$

البته در این تست از نماد O استفاده شده و گزینه ۳ هم غلط نیست ولی بهترین گزینه ۲ است.

۱. for $i := 1$ to n do begin	$\rightarrow \Theta(n)$
۲. for $j := 1$ to m do $k := k + 1$	$\rightarrow \Theta(m)$
$j := 1$	
while $j < n$ do begin	
$k := k + 1$	$\rightarrow \Theta(\lg n)$
$j := 2j$	
end	
end	

- ۴ گزینه (۴). حلقه‌های ۲ و ۳ پشت سر هم هستند پس زمان آنها با هم جمع می‌شوند که $m + \lg n$ می‌شود. هر دوی این حلقه‌ها داخل حلقه ۱ هستند که زمان حلقه ۱ یعنی n باید در $m + \lg n$ ضرب شود. پس جواب $\Theta(n(m + \lg n))$ است.

- ۵ گزینه (۲). حلقه for خطی است یعنی از مرتبه n است. حلقه لگاریتمی است یعنی از مرتبه $\lg n$ است و چون تودرتو هستند در هم ضرب می‌شوند و جواب $n \cdot \lg n$ است.
- ۶ گزینه (۳). دستورات داخل حلقه به ازای مقادیر i برابر $2^8, 2^4, 2^3, \dots$ اجرا می‌شوند. فرم مقادیر i که باعث اجرا می‌شود به صورت 2^k است که k تعداد اجرای دستورات داخل حلقه است و باید $n \simeq 2^k$ که حلقه تمام شود پس از طرفین $n \simeq 2^k$ لگاریتم بگیرید
- ۷ گزینه (۱). حلقه while لگاریتمی و حلقه for خطی است و چون تودرتو هستند ضرب می‌شوند پس مرتبه زمان $(n \cdot \lg n)^{\theta}$ است.
- ۸ گزینه (۳). به ازای $n = 1$ دستور $x := x + 1$ اصلاً اجرا نمی‌شود که اگر گزینه‌ها را با بررسی کنید فقط گزینه ۳ به ازای $n = 1$ برابر صفر می‌شود. برای یافتن تعداد اجرای $x := x + 1$ از سیگما استفاده می‌کنیم. توجه کنید حلقه while مقدار i را از ۱ تا $j - 1$ تغییر می‌دهد که k همان j است. پس این حلقه معادل $\sum_{r=1}^{j-1} i$ می‌باشد. پس تعداد اجرای $x := x + 1$ برابر است با:

$$\begin{aligned} \sum_{i=1}^n \sum_{j=i}^n \sum_{r=1}^{j-1} i &= \sum_{i=1}^n \sum_{j=i}^n (j-i) = \sum_{i=1}^n \frac{(i-1+n-1)(n-i+1)}{2} \quad (*) \\ &= \sum_{i=1}^n \frac{(n+i-2)(n-i+1)}{2} = \sum_{i=1}^n \frac{n^2 - ni + n + ni - i^2 + i - 2n + 2i - 2}{2} \\ &= \frac{1}{2} \sum_{i=1}^n (n^2 - i^2 - n + 3i - 2) = \frac{1}{2} \left[\sum_{i=1}^n (n^2 - n - 2) - \sum_{i=1}^n i^2 + 3 \sum_{i=1}^n i \right] \\ &= \frac{1}{2} \left[(n^2 - n - 1) - \frac{n(n+1)(2n+1)}{6} + \frac{3n(n+1)}{2} \right] \\ &= \frac{1}{2} \left[\frac{n(6n^2 - 6n - 12 - (n+1)(2n+1) + 9(n+1))}{6} \right] = \frac{n^3 - n}{3} \end{aligned}$$

(*) در تصاعد حسابی هر جمله برابر است با جمله قبلی بعلاوه یک مقدار ثابت به نام قدر نسبت. مجموع جملات تصاعد حسابی برابر است با: $\frac{(\text{تعداد جملات})(\text{جمله آخر} + \text{جمله اول})}{2}$

$$\text{پس } \sum_{j=i}^n (j-i) \text{ برابر است با } \frac{((i-1)+(n-1))(n-i+1)}{2}$$

[آزمون ارشد مهندسی کامپیوتر سال ۱۴۰۱]

- ۱ گراف جهت دار و وزن دار $G = (E, V)$ با رأس n و $O(n)$ یال و زیرمجموعه $S \subseteq V$ از رأس‌های گراف با اندازه حداقل $\frac{n}{2}$ داده شده است. فرض کنید وزن تمام یال‌های گراف مثبت است. به ازای هر رأس u از گراف، فاصله رأس u از مجموعه S که آن را با $d(u, S)$ نمایش می‌دهیم عبارت است از:

$$d(u, S) = \min_{v \in S} \delta(u, v)$$

که در آن $\delta(u, v)$ برابر با طول کوتاه‌ترین مسیر جهت دار از u به v در گراف G است. در چه مرتبه زمانی می‌توان مقادیر $d(u, S)$ را به ازای تمام رأس‌های u از گراف محاسبه کرد؟ دقت کنید که خروجی شامل n مقدار $d(u, S)$ به ازای تمام رأس‌های u از گراف است. (بهترین گزینه را انتخاب کنید).

- (۱) $O(n^3)$ (۲) $O(n^2)$ (۳) $O(n \log n)$ (۴) $O(n \log^2 n)$

- ۲ فرض کنید گراف G یک گراف وزن دار و مسطح با n رأس است. درخت پوشای کمینه G را در چه زمانی می‌توان محاسبه کرد؟ توجه کنید لزومی ندارد از الگوریتم‌های معروف برای محاسبه درخت پوشای کمینه استفاده شود. (بهترین گزینه را انتخاب کنید).

- (۱) $\Theta(n)$ (۲) $\Theta(n \log n)$ (۳) $\Theta(n \log \log n)$ (۴) $\Theta(n^2)$

- ۳ آرایه A شامل n عدد داده شده است. هدف پیدا کردن تعداد جفت اندیس‌های i و j است، به گونه‌ای که $j < i$ و $A[i] \times A[j] > 1$. این کار در چه زمانی قابل انجام است؟ (بهترین گزینه را انتخاب کنید).

- (۱) $O(n^3 \log n)$ (۲) $O(n^2)$ (۳) $O(n \log n)$ (۴) $O(n \log^2 n)$

- ۴ اعداد یک تا ۱۲۷ در یک هرم بیشینه که به صورت یک درخت دودویی کامل با ارتفاع ۶ پیاده‌سازی شده، قرار گرفته‌اند. حداقل تعداد برگ با مقدار بیشتر از ۱۰۰ در این درخت چقدر می‌تواند باشد؟

- (۱) ۲۰ (۲) ۱۲ (۳) ۱۱ (۴) ۲

- ۵ فرض کنید سه آرایه A و B و C هر کدام شامل n عدد، داده شده است. عناصر داخل آرایه‌ها متمایز هستند. آرایه A و C به صورت صعودی و آرایه B به صورت نزولی مرتب است. اگر بخواهیم آرایه D را بازسازیم که شامل عناصر $(A \cup B) \cap C$ باشد و به صورت

صعودی مرتب شده باشد و عضو تکراری نیز نداشته باشد، بهترین پیچیدگی زمانی ممکن برای این کار کدام مورد است؟ (توجه: ممکن است عناصری، در دو یا سه آرایه باشند).

$$O(n) \quad (4) \quad O(\log n) \quad (3) \quad O(n \log n) \quad (2) \quad O(n^3) \quad (1)$$

-۶ در کدام مورد، توابع به ترتیب صعودی (و نه اکیداً صعودی) بر مبنای رشد تابع از سمت چپ به راست، مرتب شده‌اند؟

$$\log n, \log^{\gamma} n, \log n^{\gamma}, n \log^{\gamma} n, \log n!, \log n^n \quad (1)$$

$$\log n, \log n^{\gamma}, \log^{\gamma} n, n \log^{\gamma} n, \log n^n, \log n^{\gamma}! \quad (2)$$

$$\log n, \log^{\gamma} n, \log n^{\gamma}, \log n!, \log n^n, n \log^{\gamma} n \quad (3)$$

$$\log n, \log n^{\gamma}, \log^{\gamma} n, \log n^n, \log n!, n \log^{\gamma} n \quad (4)$$

-۷ فرض کنید یک درخت دودویی جستجو بر روی n عدد حقیقی متمایز با ارتفاع $O(\log n)$ در اختیار داریم. چه تعداد از پرسمان‌های زیر را بدون پیش‌پردازش و اطلاعات اضافی می‌توان در $O(\log n)$ پاسخ داد؟ (در هر گره صرفاً یک کلید و دو اشاره‌گر به فرزندان نگه داشته شده است).

• محاسبه کوچکترین عدد

• محاسبه میانه

• تعیین آنکه آیا عدد داده شده X در درخت وجود دارد.

• محاسبه موتبه عدد X داده شده در بین n عدد ذخیره شده در درخت

$$(1) \text{ صفر} \quad (2) \quad (3) \quad (4)$$

-۸ مسئله k - مجموع بدین شکل تعریف می‌شود: مجموعه A از n عدد حقیقی و عدد k داده شده است. آیا k عضو از مجموعه A وجود دارند که جمع آنها صفر شود. چه تعداد از گزاره‌های زیر درست است؟

• مسئله ۱ - مجموع در زمان (1) قابل حل است.

• مسئله ۲ - مجموع در زمان (n) قابل حل است.

• مسئله ۳ - مجموع در زمان (n^3) قابل حل است.

$$(1) \text{ صفر} \quad (2) \quad (3) \quad (4)$$

-۹ مسئله جستجوی عنصر x در آرایه A شامل n عنصر را در نظر بگیرید. فرض کنید اطلاع داریم که توزیع ورودی به این صورت است که احتمال حضور عنصر x در نیمه دوم آرایه سه برابر احتمال حضور آن در نیمه اول است. همچنین برای هر نیمه، احتمال حضور در هر

خانه یکسان است. تعداد مقایسه‌های الگوریتم جستجوی خطی برای یافتن عنصر x در آرایه به طور متوسط چقدر است؟ (فرض کنید طول آرایه A زوج است و عدد x در آرایه وجود دارد. ضمناً جستجوی خطی از ابتدای آرایه شروع می‌شود.)

$$\frac{n}{2} \quad (4) \quad \frac{3}{4}n \quad (3) \quad \frac{5}{8}n \quad (2) \quad n \quad (1)$$

-۱۰ آرایه A شامل n عنصر داده شده است. عنصری از آرایه که حداقل $\frac{n}{3}$ بار تکرار شده باشد را یک عضو پر تکرار می‌نامیم. می‌خواهیم در آرایه A یک عضو پر تکرار را در صورت وجود پیدا کنیم. برای این کار از یک روش تقسیم و غلبه به این صورت استفاده می‌کنیم: ابتدا آرایه را به سه قسمت با اندازه برابر تقسیم می‌کنیم و در هر کدام از قسمت‌ها به طور بازگشتی در صورت وجود یک عضو پر تکرار را پیدا می‌کنیم. سپس میزان تکرار هر کدام از این سه عضو پر تکرار را در آرایه اصلی جستجو می‌کنیم و در صورتی که میزان تکرار هر عضو حداقل $\frac{n}{3}$ بود آن عضو را به عنوان عضو پر تکرار برمی‌گردانیم. کدام گزاره در خصوص این الگوریتم درست است؟ (فرض کنید n توانی از ۳ است).

- (۱) زمان اجرا $O(n \log n)$ است، اما الگوریتم لزوماً درست کار نمی‌کند.
- (۲) زمان اجرا $O(n \log n)$ است و الگوریتم درست کار می‌کند.
- (۳) زمان اجرا $O(n)$ است، اما الگوریتم لزوماً درست کار نمی‌کند.
- (۴) زمان اجرا $O(n)$ است و الگوریتم درست کار می‌کند.

-۱۱ آرایه نامتناهی A را در نظر بگیرید. فرض کنید در n خانه اول این آرایه n عدد صحیح متناهی به صورت مرتبه شده (صعودی) قرار گرفته‌اند و بقیه خانه‌های آرایه با ∞ پر شده است. به ازای عدد x داده شده می‌خواهیم برسی کنیم آیا عدد x در آرایه وجود دارد یا خیر. با چه مرتبه زمانی می‌توان به این پرسش پاسخ داد؟ (با فرض آن که مقدار n را از قبل نمی‌دانیم).

$$O(1) \quad (4) \quad O(\log n) \quad (3) \quad O(\log^2 n) \quad (2) \quad O(n) \quad (1)$$

-۱۲ فرض کنید G یک گراف بدون جهت و بدون وزن با n رأس و m یال باشد. مسئله زیر را در نظر بگیرید:

به ازای دو رأس u و v داده شده و پارامتر ورودی k ، آیا تعداد کوتاه‌ترین مسیرها بین u و v حداقل k است؟ کدام گزینه در مورد این مسئله درست است؟

- (۱) می‌توان الگوریتمی با زمان اجرای چندجمله‌ای برای این مسئله ارائه داد، اما زمان اجرای این الگوریتم نمی‌تواند بر حسب n و m خطی باشد.

۲) می‌توان الگوریتمی با زمان اجرای $O(m+n)$ برای این مسئله ارائه داد.

۳) این یک مسئله انپی – تمام است.

۴) این یک مسئله انپی – سخت است.

آزمون ارشد IT سال ۱۴۰۱

۱۳ - می‌خواهیم تابعی داشته باشیم که برای عدد طبیعی داده شده n ، در صورت اول بودن آن، مقدار ۱ و در صورت اول نبودن آن مقدار صفر را برگرداند. در تابع زیر برای این منظور، کمترین مقدار A که الگوریتم همواره درست جواب دهد، کدام است؟

```
Is_Prime(n){  
    for i = ۲ to A {  
        if (n mod i == ۰)  
            return ۰  
    }  
    return ۱  
}
```

(۱) $\lfloor \sqrt{n} \rfloor$
 (۲) $\left\lfloor \frac{n}{5} \right\rfloor$
 (۳) $n - 1$
 (۴) $\left\lfloor \frac{n}{2} \right\rfloor$

۱۴ - مرتبه زمانی الگوریتم زیر، کدام است؟

```
sum = ۰  
i = ۱  
while (i < n) {  
    j = ۰  
    while (j < i) {  
        sum = sum + ۱  
        j = j + ۱  
    }  
    i = i * ۲  
}
```

(۱) $\Theta(n^3)$
 (۲) $\Theta(n)$
 (۳) $\Theta(n^{\frac{3}{2}})$
 (۴) $\Theta(n \log n)$

۱۵ - آرایه‌ای به طول n از اعداد صحیح متمایز داده است. می‌دانیم به ازای یک اندیس i ($n \geq i \geq ۱$) عناصر آرایه از خانه ۱ تا k به صورت صعودی و از خانه k تا n به صورت نزولی هستند. اگر بخواهیم بزرگ‌ترین عدد ذخیره شده در آرایه را بیابیم، بهترین پیچیدگی زمانی الگوریتم کدام یک از گزینه‌های زیر است؟ (فرض کنید k برای الگوریتم از قبل مشخص نیست).

$O(\log^3 n)$ (۴)

$O(i)$ (۳)

$O(\sqrt{n})$ (۲)

$O(\log n)$ (۱)

۱۶- فرض کنید یک پشته با اعمال اساسی push و pop داریم. تابع mypush را به صورت زیر به پشته اضافه می‌کنیم:

```
mypush(S,x) :  
while stack S is not empty :  
    y = S.pop()  
    if x < y :  
        exit the while loop  
    end while  
S.push(x)
```

اگر با شروع از یک پشته خالی، دنباله‌ای از n تابع push و pop و mypush را با ترتیب

دلخواه روی پشته اجرا کنیم، هزینه اجرای این توابع به صورت سرکشن کدام است؟

(بهترین گزینه را انتخاب کنید.)

Θ(۱) (۴) Θ(log log n) (۳) Θ(n) (۲) Θ(log n) (۱)

۱۷- فرض کنید a و b دو عدد ثابت بزرگ‌تر از یک و n یک عدد طبیعی دلخواه باشد. چه تعداد از گزاره‌های زیر درست است؟

$$\begin{array}{lll} n^a = O(b^n) & \bullet & \log n = O(\log n^a) & \bullet & \log_a n = O(\log_b n) & \bullet \\ 3 & (4) & 1 & (3) & 2 & (1) \end{array}$$

۱۸- به ازای دو عدد طبیعی x و n داده شده، همه مقادیر x, x^2, \dots, x^n را با چند عمل ضرب می‌توان محاسبه کرد؟ (توجه کنید تنها عمل مجاز، ضرب دو عدد است).

Θ(n log n) (۴) Θ(n) (۳) Θ(n^۲) (۲) Θ(log n) (۱)

۱۹- به چند ترتیب مختلف می‌توان اعداد ۱ تا ۷ را در یک درخت دودویی جستجو درج کرد، به گونه‌ای که درخت نهایی مشابه درختی شود که از درجه‌های زیر (از چپ به راست) به دست می‌آید؟
۱, ۵, ۶, ۷, ۲, ۴, ۳

۱۰ (۴) ۱ (۳) ۱۵ (۲) ۲ (۱)

۲۰- دو لیست مرتب شده در اختیار داریم که هر یک شامل ۱۴۰۱ عدد است. برای مرتب کردن این دو لیست در بدترین حالت، حداقل چند مقایسه مورد نیاز است؟

۱۴۰۰ (۴) ۱۴۰۱ (۳) ۲۸۰۱ (۲) ۲۸۰۲ (۱)

۲۱- فرض کنید n تومان پول را می‌خواهیم با کمترین تعداد سکه‌های ۱، ۷ و ۸ تومانی خرد کنیم. اگر الگوریتم حریصانه متعارف را اجرا کنیم، به ازای چند عدد طبیعی مختلف n جواب بهینه توسط الگوریتم به دست نمی‌آید؟

۴) صفر ۲) نامتناهی ۱) ۷

- ۲۲ - گراف بدون وزن G شامل n رأس و m یال داده شده است. هر یک از یال‌های گراف با یکی از سه رنگ سبز، آبی و قرمز رنگ‌آمیزی شده است. می‌خواهیم کوتاه‌ترین مسیر از رأس ۱ به رأس n را پیدا کنیم که رنگ هر دو یال مجاور در مسیر متفاوت باشد. در چه زمانی می‌توان این کار را انجام داد؟ (بهترین گزینه را انتخاب کنید.)

$$O(m+n) \quad (1)$$

۲) برای این مسئله نمی‌توان راه حل چندجمله‌ای ارائه داد، مگر آنکه $P = NP$ باشد.

$$O(n^2) \quad (3)$$

$$O(n \log n + m) \quad (4)$$

- ۲۳ - فرض کنید n کار در اختیار داریم. زمان شروع و خاتمه کار i به ترتیب s_i و f_i است. یک پردازنده در اختیار داریم. می‌خواهیم بیشترین تعداد کاری که می‌توان به وسیله این پردازنده اجرا کرد را محاسبه کنیم. طبیعی است دو کاری که اشتراک زمانی داشته باشند نمی‌توانند بوسیله یک پردازنده همزمان اجرا شوند. برای حل این مسئله الگوریتم حریصانه متعارف بدین شکل است. کارها براساس یک پارامتر به صورت صعودی مرتب می‌شوند. براساس ترتیب فوق، کارها پردازش شده و اگر هر کار با کارهای قبلی که در خروجی قرار گرفته، هم‌پوشانی زمانی نداشته باشد در خروجی قرار می‌گیرد. به ازای چه تعداد از پارامترهای زیر الگوریتم فوق درست کار می‌کند؟

• f_i	• $s_i + f_i$
• $f_i s_i$	• s_i
۳ (۴)	۴ (۲)
۱ (۳)	۲ (۱)

- ۲۴ - فرض کنید G یک گراف وزن‌دار، همبند و بدون جهت با n رأس و m یال باشد. اگر وزن‌ها در G متمایز باشد، چه تعداد از گزاره‌های زیر درست است؟

• درخت پوشای کمینه G یکتاست.

• درخت پوشای کمینه G در زمان چندجمله‌ای بر حسب n و m قابل محاسبه است.

• درخت پوشای بیشینه G در زمان چندجمله‌ای بر حسب n و m قابل محاسبه نیست، مگر $.P = NP$.

• اگر H یک زیرگراف القایی G باشد، یال‌های درخت پوشای کمینه H (در صورت وجود) زیرمجموعه یال‌های درخت پوشای کمینه G است.

• ماکزیمم درجه درخت پوشای کمینه G شش است.

۳ (۴)	۱ (۳)	۴ (۲)	۲ (۱)
-------	-------	-------	-------

[آزمون ارشد علوم کامپیوتر سال ۱۴۰۱]

۲۵ - خروجی الگوریتم زیر از چه مرتبه‌ای است؟

```
j = 0
i = 1
while(i ≤ n)
    j = j + 2 * i
    i = 4 * i
print j
```

n (۱)
 \sqrt{n} (۲)
 $n\sqrt{n}$ (۳)
 $\sqrt{n \lg n}$ (۴)

۲۶ - با توجه به رابطه بازگشته‌ی زیر، $T(n)$ از چه مرتبه‌ای است؟

$$\begin{cases} T(2^n) = T(2^{n-1}) + \theta(n^2) \\ T(1) = \theta(1) \end{cases}$$

$(\lg n)^3$ (۱)
 $n^{\lg 2^3}$ (۲)
 n^3 (۳)
 $\lg n$ (۴)

۲۷ - هنگامی که می‌خواهیم نشان دهیم زمان یا حافظه یک الگوریتم، حداقل از مرتبه تابعی

مانند $f: \mathbb{N} \rightarrow \mathbb{R}^+$ است، از کدام نماد باید استفاده کنیم؟

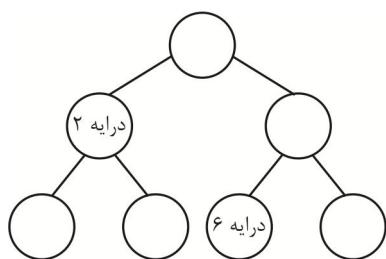
O (۱) ω (۲) Ω (۳) o (۴)

۲۸ - کدام گزاره یا گزاره‌ها درست است؟

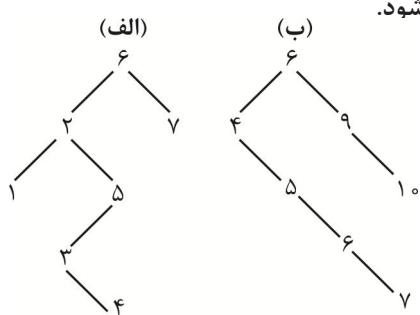
الف) سومین بزرگترین کلید در یک MAX heap با کلیدهای متمایز می‌توان در درایه با اندیس ۲ باشد.

ب) سومین بزرگترین کلید در یک MAX heap با کلیدهای متمایز می‌توان در درایه با اندیس ۶ باشد.

- (۱) فقط الف
- (۲) فقط ب
- (۳) الف و ب
- (۴) هیچ‌کدام



- ۲۹- کدام درخت یا درخت‌ها را می‌توان با یک چرخش (rotation) به درخت AVL تبدیل کرد؟
درخت AVL، درختی است که در آن برای هر رأس، قدر مطلق اختلاف ارتفاع زیر درخت‌های راست و چپ، حداقل یک است. همچنین چرخش (rotation) در درخت AVL مانند چرخش در درخت قرمز-سیاه تعریف می‌شود.



- ۱) فقط الف
- ۲) فقط ب
- ۳) هر دو
- ۴) هیچ‌کدام

- ۳۰- الگوریتم مرتب‌سازی سریع (Quick sort) را روی آرایه‌ای شامل ۴ عدد اجرا می‌کنیم.
تعداد مقایسه‌ها چه عددی نمی‌تواند باشد؟

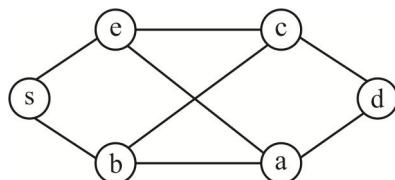
- ۱) ۳
- ۲) ۴
- ۳) ۵
- ۴) ۶

- ۳۱- حداقل حافظه کمکی لازم برای دو عمل زیر، به ترتیب از راست به چپ از چه مرتبه‌ای است؟

- بر عکس کردن یک لیست پیوندی (linked list) یک طرفه n عضوی
- بر عکس کردن یک لیست پیوندی (linked list) دو طرفه n عضوی

- ۱) ۱ و n
- ۲) n و ۱
- ۳) ۱ و n
- ۴) ۱ و ۱

- ۳۲- دو الگوریتم BFS (جستجوی اول سطح) و DFS (جستجوی اول عمق) را روی گراف زیر و با شروع از رأس S اجرا می‌کنیم. در هر گام از BFS یا DFS اگر چند رأس برای انتخاب وجود داشت، رأسی را انتخاب می‌کنیم که از نظر ترتیب الفبای انگلیسی زودتر باشد. درخت‌های حاصل از BFS و DFS را به ترتیب T_B و T_D مینامیم. فاصله دو رأس a و e از همدیگر (و نه از S) در T_B و T_D به ترتیب از راست به چپ چند است؟



- ۱) ۱ و ۱
- ۲) ۱ و ۳
- ۳) ۱ و ۱
- ۴) ۳ و ۳

۳۳- پیاده‌سازی معمولی کدام مورد با داده ساختار پشته (stack) انجام می‌شود؟

Insertion sort (۲)

Radix sort (۱)

Bubble sort (۴)

Quick sort (۳)

۳۴- کدام گزاره یا گزاره‌ها درست است؟

(الف) ارتفاع یک درخت جستجوی دودویی (binary search tree) با ۰ کلید می‌تواند ۹ باشد.

(ب) اگر داده‌ها کلیدها اعداد ۱, ۲, ..., ۷ باشند فقط یک درخت جستجوی دودویی به ارتفاع ۲ وجود دارد.

۴) هیچ‌کدام

۳) الف و ب

۲) فقط ب

۱) فقط الف

۳۵- G یک گراف همبند با n رأس و e یال است. همچنین می‌دانیم که وزن همه یال‌های G یکسان است. بهترین زمان برای یافتن یک زیردرخت فراگیر (پوشای) کمینه G از چه مرتبه‌ای است؟

e lg n (۴)

n lg n (۳)

n (۲)

e (۱)

۳۶- قصد داریم الگوریتمی طراحی کنیم که با ورودی گرفته یک درخت دودویی T با n رأس (گره) تعیین کند T درخت جستجوی دودویی است یا خیر. بهترین زمان برای این کار از چه مرتبه‌ای است؟

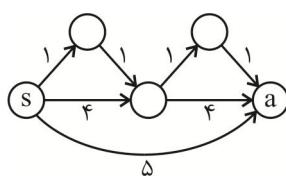
$n^2 \lg n$ (۴)

n^2 (۳)

n lg n (۲)

n (۱)

۳۷- الگوریتم دایکسترا را روی گراف زیر و با شروع از رأس s اجرا می‌کنیم. فاصله a از s در ابتدا $+\infty$ در نظر گرفته می‌شود. هنگام اجرای الگوریتم، فاصله a از s چند بار تغییر می‌کند؟



۲ (۲)

۱ (۱)

۴ (۴)

۳ (۳)

۳۸- کدام مسئله یا مسئله از ۳ مسئله زیر دارای الگوریتم حریصانه (greedy) با زمان چندجمله‌ای است؟ کوله‌پشتی ۰ و ۱ / کوله‌پشتی کسری / زیردرخت فراگیر (پوشای) کمینه (MST)

۱) فقط کوله‌پشتی ۰ و ۱

۲) فقط کوله‌پشتی ۰ و ۱ و زیردرخت فراگیر (پوشای) کمینه

۳) فقط کوله‌پشتی کسری و زیردرخت فراگیر (پوشای) کمینه

۴) فقط کوله‌پشتی کسری

- ۳۹ بهترین زمان برای تعیین دو بخشی بودن یا نبودن یک گراف با n رأس و e یال از چه مرتبه زمانی است؟ (گراف دو بخشی یعنی گرافی که بتوان رأس‌های آن را به دو بخش A و B افراز کرد به طوری که هر یال G یک سر در A و یک سر در B داشته باشد).

$$\Theta(ne) \quad (1)$$

۴۰ مسئله $NP-hard$ است.

$$\Theta(ne) \quad (3)$$

- ۴۰ کدام گزاره یا گزاره‌ها در مورد مسئله جریان (شار) در شبکه‌ها (Network Flow) درست است؟

الف) اگر ظرفیت هر یال مضرب $\frac{1}{3}$ باشد، مقدار جریان (شار) بیشینه نیز مضرب $\frac{1}{3}$ است.

ب) دقیقاً یک برش (cut) وجود دارد که ظرفیت آن برابر با مقدار جریان بیشینه است.

$$(1) \text{ فقط الف} \quad (2) \text{ فقط ب} \quad (3) \text{ هر دو} \quad (4) \text{ هیچ کدام}$$

- ۴۱ کدام گزاره یا گزاره‌ها در مورد کلاس P درست است؟

الف) $L \in P$ اگر و تنها اگر الگوریتم A موجود باشد که برای هر ورودی به طول n بیت،

عدد ثابت $c > 0$ موجود باشد که A در زمان $O(n^c)$ پاسخ درست را بدهد.

ب) $L \in P$ اگر و تنها اگر برای هر ورودی به طول n بیت، الگوریتم A و عدد ثابت $c > 0$ موجود باشد که در زمان $O(n^c)$ پاسخ درست را بدهد.

$$(1) \text{ فقط الف} \quad (2) \text{ فقط ب} \quad (3) \text{ هر دو} \quad (4) \text{ هیچ کدام}$$

- ۴۲ G یک گراف وزن‌دار با وزن‌های مثبت و با n رأس است؛ به طوری که بیشترین درجه رأس‌ها $\lg n$ است. کدام الگوریتم برای محاسبه فاصله هر دو رأس از هم دیگر مرتبه زمانی

بهتری دارد؟ (فرض کنید n به اندازه کافی بزرگ است).

$$(1) \text{ فلوید - وارشاو} \quad (2) \text{ بار بلمن - فورد (با شروع از هر رأس)}$$

$$(3) \text{ بار دایکسترا (با شروع از هر رأس)} \quad (4) \text{ بار BFS (با شروع از هر رأس)}$$

- ۴۳ فرض کنید $f: \mathbb{N} \rightarrow \mathbb{N}$ یک تابع باشد که برای هر i ، محاسبه $f(i)$ در زمان $(\sqrt{i})^\theta$

ممکن باشد. همچنین برای هر i و j که $1 \leq i \leq n$ و $1 \leq j \leq n$ تعریف می‌کنیم:

$$g(i, j) = \begin{cases} f(i) : i = j & \text{اگر} \\ \max \{g(i, k-1) + g(k, j)\} : i < j & \text{اگر} \\ & i < k \leq j \end{cases}$$

محاسبه $g(1, n)$ با الگوریتم پویا (دینامیکی) ساده‌ای که از رابطه بالا حاصل می‌شود، از چه

مرتبه زمانی است؟

$$n^{\sqrt{n}} \quad (4) \quad n^{\sqrt{n}} \quad (3) \quad n^{\sqrt{n}} \quad (2) \quad n^{\sqrt{n}} \quad (1)$$

۴۴- فرض کنید $b = b_1 b_2 \dots b_n$ یک رشته n حرفی با یک الفبای Σ باشد. منظور از یک زیردنباله b ، یک رشته حاصل از حذف برخی حروف‌های b است (بنابراین b دارای 2^n زیردنباله است) به عنوان مثال $xywy$ یک زیردنباله yx است اما $\gamma\alpha\alpha\beta\alpha\alpha\gamma$ زیردنباله آن نیست. همچنین منظور از یک رشته متقارن، رشته‌ای است که قرائت آن از چپ برابر با قرائت آن از راست باشد؛ به عنوان مثال $b_i b_{i+1} \dots b_j$ را با $B[i, j]$ نشان می‌دهیم. همچنین تعریف می‌کنیم:

$$x = \max \{B[i, j-1], B[i+1, j]\}$$

$$y = B[i+1, j-1]$$

کدام گزینه برای i و j ‌هایی که $j < i$ درست است؟

$$B[i, j] = \begin{cases} x & : b_i \neq b_j \\ y+1 & : b_i = b_j \end{cases} \quad (1)$$

$$B[i, j] = \begin{cases} x & : b_i \neq b_j \\ y+2 & : b_i = b_j \end{cases} \quad (2)$$

$$B[i, j] = \begin{cases} x+1 & : b_i \neq b_j \\ y+1 & : b_i = b_j \end{cases} \quad (3)$$

$$B[i, j] = \begin{cases} x+1 & : b_i \neq b_j \\ y+2 & : b_i = b_j \end{cases} \quad (4)$$

پاسخنامه آزمون ارشد مهندسی کامپیوتر سال ۱۴۰۱

-۱ گزینه (۴). زیرمجموعه S را کلاً تبدیل به یک سوپر نود می‌کنیم و جهت یال‌ها را بر عکس کرده و از این سوپر نود S دایجسترا می‌زنیم. مرتبه دایجسترا با هیپ فیبوناچی $O(n \lg n)$ است که چون گراف خلوت است ($e \in O(n)$) پس مرتبه $O(e + n \lg n)$ می‌باشد.

-۲ گزینه (۱). فرض کن G^* دوگان G است که با مرتبه خطی پیدا می‌شود.

Let T and $T^* := \text{null}$, $G_1 = G$, $G_1^* = G^*$

while true

if G_1 and G_1^* are empty then ret [T, T^*]

$v :=$ vertex in G_1 or G_1^*

if v contains a self loop

$f :=$ edge with the selfloop

$G_1 := G_1 \setminus f$, $G_1^* := G_1^* \setminus f$

if v belongs to G_1 then $T^* := T^* \cup \{f\}$

if v belongs to G_1^* then $T := T \cup \{f\}$

if v contains no selfloop and belongs to G_1 ,

$E :=$ set of edges incident on v

$e :=$ edge in E with min weight

$G_1 := G_1 / e$

$G_1^* := G_1^* \setminus e$

$T := T \cup \{e\}$

if v contains no self loop and belongs to G_1^*

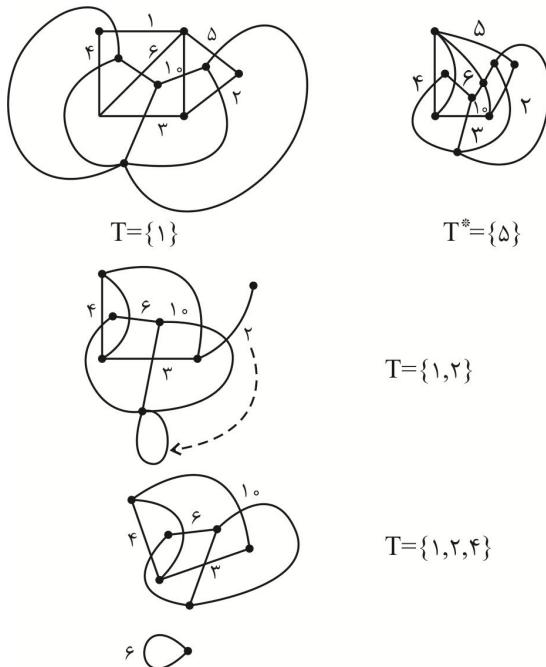
$E :=$ set of edges incident on v

$e :=$ edge in E with max weight

$G_1 := G_1 \setminus e$

$G_1^* := G_1^* / e$

$T^* := T^* \cup \{e\}$



-۳ گرینه (۳). نامساوی داده شده را به شکل $\frac{A[i]}{i} > \frac{j}{A[j]}$ بنویسد. فرض کنیم آرایه

$A[1..n]$ است و عنصر صفر ندارد (اگر صفر داشت حذف می‌کنیم و بعد مقایسه می‌کنیم)

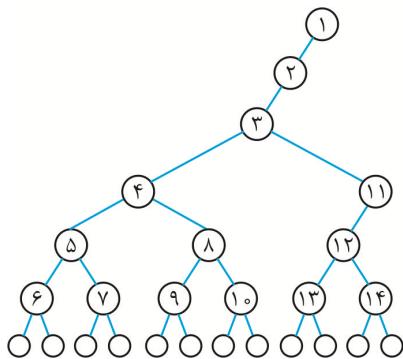
سپس دو آرایه B و C را می‌سازیم اولی شامل عناصر $\frac{A[i]}{i}$ و دومی شامل $\frac{j}{A[j]}$ که زمان

$\theta(n)$ صرف می‌شود سپس آرایه‌های B و C را صعودی مرتب می‌کنیم. سپس تک تک

عناصر B را در C جستجوی دودویی می‌کنیم با این هدف که $B[i] > C[j]$ بزرگتر پیدا

شود که این مراحل زمان $O(n \lg n)$ می‌خواهد.

-۴ گرینه (۲). حداقل ۱۲ عدد بیشتر از ۱۰۰ می‌توانند هم‌زمان در سطح آخر باشند.



-۵ گزینه (۴). آرایه B را برعکس کرده و با آرایه A ادغام می‌کنیم و هنگام ادغام تکراری‌ها را حذف می‌کنیم و سپس $A \cup B$ را با C ادغام می‌کنیم و غیرتکراری‌ها را حذف می‌کنیم و از هر تکرار فقط یک نمونه نگه می‌داریم.

-۶ گزینه (۴).

-۷ گزینه (۳). موارد اول و سوم با زمان $O(\lg n)$ قابل پاسخ است. موارد دوم و چهارم زمان $O(n)$ نیاز دارد.

-۸ گزینه (۴). مسئله اول در زمان $O(n)$ و مسئله دوم در زمان $O(n \lg n)$ قابل حل است.

مسئله سوم را می‌توان در زمان $(\lg n)^3$ حل کرد. در واقع باید ۳ عنصر A را پیدا کنیم که جمعشان صفر شود یعنی $A[i] + A[j] + A[k] = 0$.

برای این منظور A را با زمان $O(n \lg n)$ مرتب کرده و تساوی را به شکل $A[i] + A[j] = -A[k]$ می‌نویسیم. می‌دانیم باقتن دو عنصر در آرایه مرتب A که جمعشان x شود یعنی $A[i] + A[j] = x$ است پس به ازای هر عنصر A این بررسی را انجام می‌دهیم که مرتبه $O(n^3)$ می‌شود یعنی:

for ($k = 1$ to n)

is there i and j that $A[i] + A[j] = -A[k]$

-۹ گزینه (۲). اگر احتمال حضور x در $\frac{n}{2}$ خانه اول p باشد احتمال حضور x در $\frac{n}{2}$ خانه دوم

است. پس $1 = \frac{1}{\sqrt{n}} (\frac{3n}{2} + \frac{n}{2})p$ یعنی $\frac{n}{2}p + \frac{n}{2}3p = p$. متوسط تعداد مقایسه x برابر است:

$$\left(1 \times \frac{1}{\sqrt{n}} + 2 \times \frac{1}{\sqrt{n}} + \dots + \frac{n}{2} \times \frac{1}{\sqrt{n}}\right) + \left(\frac{n}{2} + 1\right) \times \frac{3}{\sqrt{n}} + \left(\frac{n}{2} + 2\right) \times \frac{3}{\sqrt{n}} + \dots + n \times \frac{3}{\sqrt{n}}$$

$$= \frac{1}{\sqrt{n}} \left[\left(1 + 2 + \dots + \frac{n}{2}\right) + 3 \left(\left(\frac{n}{2} + 1\right) + \left(\frac{n}{2} + 2\right) + \dots + n\right) \right]$$

$$= \frac{1}{\sqrt{n}} \left[\frac{\left(\frac{n}{2}\right)\left(\frac{n}{2} + 1\right)}{2} + \frac{3 \left[\left(\frac{n}{2} + 1\right) + n\right] \frac{n}{2}}{2} \right]$$

$$= \frac{1}{\sqrt{n}} \left[\frac{\frac{n^2}{4} + \frac{n}{2} + \frac{9n^2}{4} + \frac{3n}{2}}{2} \right] = \frac{1}{\sqrt{n}} \left[\frac{n^2 + 2n + 9n^2 + 6n}{8} \right]$$

$$= \frac{1}{\sqrt{n}} \left(\frac{5n^2 + 8n}{4} \right) = \frac{5n^2 + 8n}{4\sqrt{n}} = \frac{5n}{4} + 2 \approx \frac{5}{4}n$$

۱۰ - گزینه (۱). زمان اجرای این الگوریتم حداکثر $T(n) = 3T\left(\frac{n}{3}\right) + \theta(n) = \theta(n \lg n)$ است.

ولی لزوماً درست کار نمی‌کند مثلاً برای آرایه زیر

۲ ۲ ۱ | ۳ ۱ ۴ | ۳ ۱ ۴

عنصر پر تکرار وجود دارد و ۱ است ولی الگوریتم لزوماً آن را پیدا نمی‌کند.

۱۱ - گزینه (۳). یک نوع جستجوی دودویی به این صورت که x را با $A[۱]$ و $A[۲]$ و $A[۴]$ و $A[۸]$ و ... $A[i]$ و $A[۲i]$ مقایسه می‌کنیم و اگر $x < A[i]$ و $x > A[۲i]$ آنگاه آرایه $A[i..2i]$ را جستجوی دودویی معمولی می‌کنیم.

۱۲ - گزینه (۲). در گراف بدون وزن غیرجهتدار می‌توان با تغییراتی در BFS مسئله را حل کرد. در BFS برای هر نود دو مقدار ذخیره می‌کنیم: $distance$ که معرف طول کوتاهترین مسیر از مبدأ به نود جاری است. $paths$ معرف تعداد این کوتاهترین مسیرهاست. در ابتدا مقدار $distance$ برای مبدأ صفر و برای بقیه رئوس ∞ است و مقدار $paths$ برای مبدأ ۱ و برای بقیه صفر است. حال با BFS پیمایش می‌کنیم، هر زمان که می‌خواهیم یک فرزند از نود فعلی را به صف اضافه کنیم، دو انتخاب داریم:

اگر $distance[child] > distance[current]$ به این معنی است که مسیری از مبدأ به نود فرزند با طول کمتر از مسیر قبلی وجود دارد. قرار می‌دهیم $paths[child] = paths[current]$ و $distance[child] = distance[current] + 1$ قرار می‌دهیم.

ولی اگر $dist[child] == dist[current] + 1$ به این معنی است که تمام کوتاهترین مسیرها که از رأس مبدأ به رأس جاری می‌رود به تعداد کوتاهترین مسیرهای نود فرزند اضافه خواهد شد. علت این است که آنها اندازه یکسان با کوتاهترین مسیر نود فرزند بعد از اضافه کردن یال که نود جاری را به فرزند وصل می‌کند دارند. پس ما مقدار فاصله نود فرزند را تغییر نمی‌دهیم و لی مقدار $paths$ نود فرزند را با مقدار $paths$ نود جاری افزایش می‌دهیم.

[پاسخنامه آزمون ارشد IT سال ۱۴۰۱]

- گزینه (۱). برای برسی اول بودن عدد n باید برسی کنیم که آیا n به هیچ یک از اعداد اول $2 \leq \sqrt{n}$ بخش‌پذیر است یا نه. درواقع بزرگترین عددی که ممکن است n بر آن تقسیم کنیم پس کافی است عدد n را به اعداد $2 \leq \sqrt{n}$ بخش‌پذیر باشد.

به هر یک بخش‌پذیر بود اول نیست و در غیر این صورت اول است.

- گزینه (۲). حلقه داخلی به ازای هر i به تعداد i بار اجرا می‌شود و مقدار $\theta(n)$ نیز $sum = sum + 2^k$ است که $n = 2^k + 2^{k-1} + \dots + 2^1 + 2^0$ پس تعداد اجرای دستور اصلی یعنی i برابر 2^k است که از مرتبه $\theta(2^k) = \theta(n)$ است.

- گزینه (۳). چنین آرایه‌ای با اینویک نام دارد و برای یافتن ماکریم کافی است شبیه جستجوی دودویی عمل کنیم به این صورت که عنصر وسط را با قبل و بعدش مقایسه کنیم و با این مقایسه می‌فهمیم که نیمی از عناصر باید حذف شوند یا ماکریم پیدا می‌شود:

$A[mid - 1] < A[mid] < A[mid + 1] \Rightarrow goto A[mid + 1.. high]$

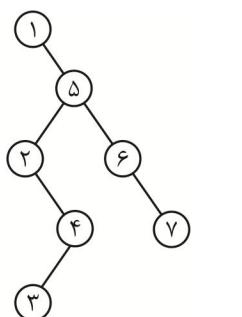
$A[mid - 1] > A[mid] > A[mid + 1] \Rightarrow goto A[low .. mid - 1]$

$A[mid - 1] < A[mid] > A[mid + 1] \Rightarrow A[mid] \text{ is max}$

- گزینه (۴). کافی است هزینه استهلاکی $2 + 2^0$ را به ترتیب به push و mypush و pop نسبت دهیم در این صوری وقتی عنصری push می‌شود هزینه واقعی یک مصرف می‌شود و یک واحد پول در عنصری پوش شده ذخیره می‌شود که این یک واحد هنگام pop یا هنگام خارج شدن توسط mypush مصرف خواهد شد. هنگام mypush نیز عناصری که از پشته خارج می‌شوند خود ذخیره یک واحد دارند و آن را مصرف کرده خارج می‌شوند و یک واحد پولی که ما پرداخت می‌کنیم، یک واحدش صرف قرارگیری عنصر در پشته می‌شود و یک واحد در عنصر ذخیره می‌شود پس هزینه استهلاکی هر عمل $\theta(1)$ است.

- گزینه (۵). هر سه درست هستند. $a, b > 1$ ثابت هم‌رشد هستند. و $\log_b n$ برابر $\log_a n$ است که با $\log_a \log_b n$ هم‌رشد است. تابع n^a چندجمله‌ای است که از تابع نمایی b^n رشد کمتری دارد.

- گزینه (۶). هر یک از این x^i ‌ها را فقط با یک عمل ضرب می‌توان محاسبه کرد. به این صورت که برای یافتن x^i کافی است x^{-1} را در x ضرب کنیم پس محاسبه تمامی این x^i ‌ها زمان $\theta(n)$ می‌خواهد.



- ۱۹ - گزینه (۴). شکل درخت به شکل مقابل است و تعداد حالات ممکن برابر است با:

$$\frac{5!}{3!2!} = 10$$

علت این را بررسی کنید!

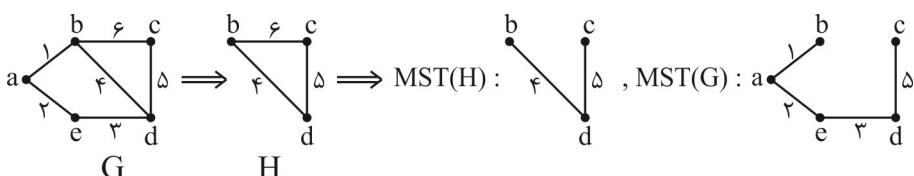
- ۲۰ - گزینه (۲). حداقل مقایسه ادغام دو لیست مرتب m و n عنصری $m+n-1$ می‌باشد پس $1401+1401-1=2801$.

- ۲۱ - گزینه (۲). مبلغ ۱۴ با حریصانه نیاز به ۷ سکه دارد ($8+1+1+1+1+1$) ولی جواب بهینه ۲ سکه است ($7+7$). وقتی به ازای حداقل یک حالت، حریصانه برای مسئله سکه جواب بهینه ندهد. بی‌شمار حالت وجود دارد که جواب نمی‌دهد. درواقع جواب چنین سؤالی یا صفر یا بی‌نهایت است.

- ۲۲ - گزینه (۱). ایده جالبی برای حل این مسئله وجود دارد. از گراف G گراف G' را می‌سازیم به این صورت که به ازای هر رأس در G سه رأس در G' به سه رنگ مختلف قرار می‌دهیم. و اگر یالی با رنگ x بین i و j در G وجود داشت از رأس با رنگ x معادل i در G' به رئوس با رنگ متفاوت x معادل یال‌هایی با رنگ x وصل می‌کنیم. پس از ایجاد G' , سه بار BFS با شروع از رئوس متناظر رأس ۱ می‌زنیم.

- ۲۳ - گزینه (۳). برای این مسئله دو الگوریتم حریصانه وجود دارد یکی براساس ترتیب صعودی i ها و دیگری براساس ترتیب نزولی i ها که در سؤال گفته صعودی پس فقط یک مورد درست است.

- ۲۴ - گزینه (۱). وزن‌ها متمایز است پس MST یکتاست و زمان محاسبه MST بر حسب n و m چندجمله‌ای است و با روش‌های کراسکال و پریم می‌توان به آن رسید. درخت پوشای بیشینه همان زمان درخت پوشای کمینه را دارد و مثلاً کافی است وزن یال‌ها قرینه شوند و سپس MST یافت. برای نقض گزاره چهارم به مثال صفحه بعد توجه کنید:



یال‌های $MST(H)$ زیرمجموعه یال‌های $MST(G)$ نیستند، یال bd در $MST(H)$ هست ولی در $MST(G)$ نیست.

گزاره آخر نیز نادرست است و برای نقض آن مثلاً یک درخت ستاره که درجه رأس مرکزی ۷ است را در نظر بگیرید. MST این درخت خود درخت است که رأس درجه ۷ دارد. پس فقط گزاره‌های اول و دوم صحیح هستند.

[پاسخنامه آزمون ارشد علوم کامپیوتر سال ۱۴۰۱]

- گزینه (۱).

$$\begin{array}{c|ccccccccc} i & 1 & 4 & 4^2 & 4^3 & \dots & 4^k \\ \hline j & 0 & 2 & 2+2\times 4 & 2+2\times 4+2\times 4^2 & \dots & 2+2\times 4+2\times 4^2+\dots+2\times 4^{k-1} \end{array}$$

خروجی الگوریتم j می‌باشد. الگوریتم وقتی خاتمه می‌یابد که $n = 4^k$ یعنی حال j را محاسبه می‌کنیم:

$$j = 2 + 2 \times 4 + 2 \times 4^2 + \dots + 2 \times 4^{k-1} = 2(1 + 4 + \dots + 4^{k-1})$$

$$= 2 \frac{4^k - 1}{4 - 1} = \frac{2}{3}(4^k - 1) = \frac{2}{3}(n - 1) = \theta(n)$$

- گزینه (۴).

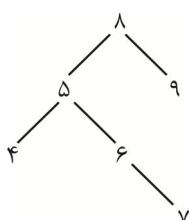
$$\begin{aligned} T(4^n) &= T(4^{n-1}) + cn \stackrel{4^n \approx m}{\Rightarrow} T(m) = T\left(\frac{m}{2}\right) + c \lg 4 m \\ &= \theta(\lg 4 m) \Rightarrow T(n) = \theta(\lg 4 n) \end{aligned}$$

- گزینه (۲). وقتی می‌گوییم زمان یا حافظه یک الگوریتم $\Omega(f)$ است یعنی زمان یا حافظه آن الگوریتم حداقل هم مرتبه با f است یعنی از مرتبه f بیشتر یا مساوی است.

- گزینه (۳). در یک هرم بیشینه با بی شمار کلید، k امین بزرگترین کلید که $1 \neq k$ می‌تواند

در انديس ۲ تا $2^k - 1$ باشد پس هر دو گزاره صحیح هستند.

- گزینه (۲). در «الف» هر نودی را بررسی کنید با یک دوران درخت متوازن نمی‌شود. در «ب» اگر ۴ را به چه دوران دهیم درخت متوازن حاصل می‌شود:



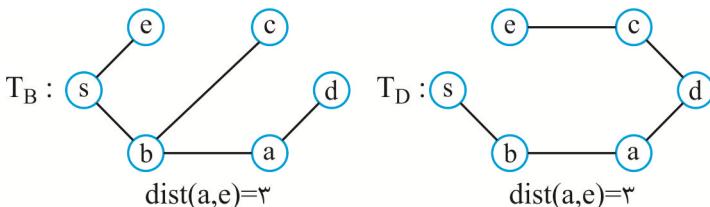
- ۳۰ - گزینه (۱). حداقل تعداد مقایسه‌ها در روش سریع از مرتبه $\theta(n \lg n)$ است. در روش‌های مثل حبابی و درجی حداقل تعداد مقایسه ۱ است.

حالات‌های مختلف (فرض کنید $a < b < c < d$) :

- ۱) $a \ b \ c \ d \xrightarrow[\text{مقایسه ۳}]{\text{محور } a} a \boxed{b \ c \ d} \xrightarrow[\text{مقایسه ۲}]{\text{محور } b} a \ b \boxed{c \ d} \xrightarrow[\text{مقایسه ۱}]{\text{محور } c} a \ b \ c \ d$
- ۲) $b \ a \ c \ d \xrightarrow[\text{مقایسه ۳}]{\text{محور } b} a \ b \boxed{c \ d} \xrightarrow[\text{مقایسه ۱}]{\text{محور } c} a \ b \ c \ d$
- ۳) $a \ c \ b \ d \xrightarrow[\text{مقایسه ۳}]{\text{محور } a} a \boxed{c \ b \ d} \xrightarrow[\text{مقایسه ۲}]{\text{محور } c} a \ b \ c \ d$

- ۳۱ - گزینه (۴). با ۳ متغیر می‌توان یک لیست یکطرفه را معکوس کرد که حافظه کمکی (۱) head را برعکس کردن لیست دوطرفه نیز بی‌معنی است و در واقع کافی است اشاره‌گر tail را جابجا کرد که اصلاً حافظه کمکی نیاز ندارد.

- ۳۲ - گزینه (۴).



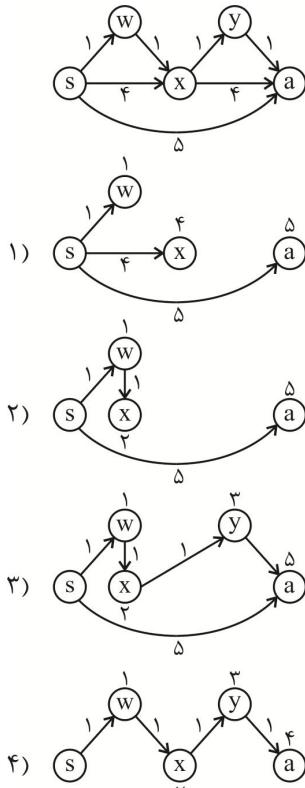
- ۳۳ - گزینه (۳). مرتب‌سازی سریع الگوریتم بازگشته دارد و از پشته استفاده می‌کند. البته می‌توان روش‌های دیگر را نیز بازگشته پیاده‌سازی کرد ولی پیش‌فرض آنها غیربازگشته است.

- ۳۴ - گزینه (۳). «الف» درست است. BST می‌تواند هر شکلی داشته باشد. با ۱۰ کلید ارتفاع ۳ حداقل و حداً کثر ۹ می‌باشد. «ب» درست است. با ۷ کلید تنها BST که ارتفاع ۲ دارد درخت پر می‌باشد.

- ۳۵ - گزینه (۱). در واقع هدف یافتن درخت پوشاست که الگوریتم BFS یا DFS با مرتبه $O(n + e)$ انجام می‌شود چون گراف همبند است مرتبه $O(e)$ می‌شود. کلید اولیه سؤال گزینه ۲ است که به نظر اشتباه است.

- ۳۶ - گزینه (۱). کافی است یک پیمایش میان ترتیب انجام شود و لیست نهایی بررسی شود که آیا مرتب صعودی است یا خیر که مرتبه این عملیات $O(n)$ است.

- ۳۷- گزینه (۲).



S	W	X	y	a
۰	∞	∞	∞	∞
۱	۴	∞	۵	
۲	∞	۵		
۳	۵			
۴				

فاصله s تا a دو بار تغییر می‌کند. یکبار ۵ و یکبار ۴ می‌شود.

- ۳۸- گزینه (۳). کوله‌پشتی صفر و یک با روش حریصانه لزوماً جواب بهینه نمی‌دهد ولی کوله‌پشتی کسری با روش حریصانه جواب بهینه می‌دهد کافی است اشیاء را براساس مقادیر نزولی ارزش به وزن انتخاب کنیم. یافتن MST نیز با روش‌هایی مثل کراسکال و پریم حریصانه است.

- ۳۹- گزینه (۱). برای تعیین دوبخشی بودن گراف کافی است یک پیمایش مثل BFS یا DFS انجام دهیم و به رئوس در حین پیمایش برچسب ۱ و ۲ بدهیم یعنی رأس با برچسب ۱ اگر رأسی را ملاقات کرد به آن ۲ بدهیم و بر عکس و در نهایت بررسی کنیم آیا هیچ دو رأس مجاوری هم‌شماره هستند که اگر باشند گراف دوبخشی نیست.

- ۴۰- گزینه (۱). «الف» درست است. در هر مرحله از روش فورد فالکرسون یال با کمترین ظرفیت انتخاب شده و به همان اندازه شار در برخی مسیرها به روز می‌شود یعنی در هر مرحله مضربی از $\frac{1}{2}$ شار به روزرسانی می‌شود و در نهایت شار بیشینه مضرب $\frac{1}{2}$ خواهد بود.

«ب» نادرست است. برش کمینه منحصر به فرد نیست ولی عدد آن یکتاست. مثلاً در شکل زیر هر برشی کمینه است.



- ۴۱ - گزینه (۴).

- ۴۲ - گزینه (۴). تعداد یال در این گراف از مرتبه $O(n \lg n)$ است. مرتبه زمانی یکبار دایکسترا با هیپ فیبوناچی $O(e + n \lg n)$ است که در این گراف $O(n \lg n)$ می‌شود. پس n بار دایکسترا مرتبه اش $O(n^3 \lg n)$ می‌شود. مرتبه فلويد وارشال $(O(n^3))$ است. مرتبه n بار بلمن فورد $(O(n^3 e))$ می‌شود. n بار BFS هم اصلاً نمی‌تواند این مسئله را حل کند.

- ۴۳ - گزینه (۳). درایه‌های قطر اصلی هر یک با مرتبه $O(\sqrt{n})$ محاسبه می‌شوند ($(1, 1) g(1, 1)$) با مرتبه $(1, \theta(1), \theta(2, 2), \dots, \theta(\sqrt{2}))$. سایر درایه‌ها که تعدادشان (n^3) تاست هر یک با مرتبه (n) محاسبه می‌شوند زیرا در بین (n) تا عبارت باید ماکزیمم‌گیری کنند. پس مرتبه محاسبه این درایه‌ها $(O(n^3))$ می‌باشد.

- ۴۴ - گزینه (۲). اگر $b_j = b_i$ آنگاه هر دو باید انتخاب شوند که ۲ واحد به جواب اضافه می‌شود و سپس باید از $i+1$ تا $j-1$ بازگشت کنیم یعنی $y+2$ و این حالت می‌شود $y+2$ اگر $b_i \neq b_j$ چیزی انتخاب نمی‌شود و فقط دو بار بازگشت می‌کنیم یکبار با حذف $i+1$ تا j و یکبار با حذف j از $i+1$ تا j که این یعنی x .