به نام خدا

ساختمان داده‌ها و الگوریتم‌ها (۴۰۲۵۴) ۳ دی ۱۴۰۱

دانشگاه صنعتی شریف مدرس: مهدی صفرنژاد

آزمون میان‌ترم

زمان این امتحان ۱۵۰ دقیقه است. امتحان از ۱۱۰ نمره است که ۱۰ نمره آن امتیازی است. ترتیب سوالات به صورت تصادفی است بنابراین توصیه می‌شود روی تمام سوالات وقت بگذارید. لطفا هر سوال را داخل یک برگه جدا پاسخ دهید و نام و شماره دانشجویی خود را بر روی تمام برگه‌ها بنویسید. امیدوارم تا این‌جای کلاس با تفکر الگوریتمی آشنا شده باشید و با همین تفکر به سؤالات امتحان به خوبی پاسخ دهید.

سؤال ۱. (۲۰ نمره) کَلمُراد و حسین دوستان خوبی برای هم هستند.کَلمُراد k آرایه به طول n دارد، او آن‌ها را یک روز به حسین قرض داده و سپس هنگامی که حسین آن‌ها را به او پس می‌دهد متوجه می‌شود که تمام آرایه‌ها مرتب شده‌اند. یک شب در خواب حسین می‌بیند که شرک بالای سر کَلمُراد با یک شمشیر ایستاده و از حسین می‌خواهد تا میانه تمام nk عدد موجود در آرایه‌ها را به او بگوید وگرنه کَلمُراد را به قتل می رساند. حسین برای اینکه به موقع کَلمُراد را از دست غول سبز رنگ نجات دهد لازم است الگوریتمی با پیچیدگی زمانی $O(k\log(k)\log(n))$ طراحی کند که میانه‌ی تمام این اعداد را پیدا کند. به حسین در طراحی این الگوریتم کمک کنید.

اگر سوال را با پیچیدگی زمانی $O(k^۲\log^۲(n))$ حل کنید نیمی از نمره را دریافت خواهید کرد.

پاسخ:

مسئله را تعمیم می‌دهیم.

فرض کنید k آرایه مرتب شده (با طول‌های نه لزوما یکسان) داریم که طول آرایه i ام برابر l_i است و $s=\sum_{i=۱}^{k} l_i$ تعریف می‌کنیم و یک عدد t هم به ما داده شده است و می‌خواهیم t امین عدد در ترتیب مرتب شده‌ی تمام این s عدد را در زمان $O(\log k \times \sum_{i=۱}^{k}\log l_i)$ پیدا کنیم.

بدیهی است با حل این سوال مسئله اصلی هم حل می‌شود چون اگر داشته باشیم $t=\lceil \frac{nk}{۲} \rceil$ و $l_i=n$ سوال اصلی به این سوال تبدیل می‌شود.

پس به حل سوال تعمیم داده شده می‌پردازیم. با توجه به این که آرایه‌ها مرتب هستند می‌توانیم میانه هر آرایه را در $O(۱)$ محاسبه کنیم (عدد $\lceil \frac{l_i}{۲} \rceil$ میانه آرایه i ام است).

همچنین تعریف می‌کنیم $r=\sum_{i=۱}^{k} \lceil \frac{l_i}{۲} \rceil$.

حال بین تمام k میانه آرایه‌ها بزرگترین میانه را در نظر بگیرید ، با توجه که این عدد از تمامی میانه‌های دیگر بزرگتر است پس از حداقل r عدد بزرگتر است و اگر داشته باشیم $t<r$ بزرگترین میانه و اعداد سمت راست آن در آرایه‌ی متناظرش

جواب نخواهند بود چون از حداقل r عدد دیگر بزرگتر هستند و $t < r$ است در نتیجه می‌توانیم آرایه متناظر با بزرگترین میانه را با نصفه‌ی سمت چپ این آرایه جایگزین کنیم و اگر $t \geq r$ بود می‌توانیم حال کوچکترین میانه را در نظر بگیریم و همانند استدلال قبل اعداد سمت چپ کوچکترین میانه در جواب نخواهند بود و می‌توانیم آرایه متناظر با کوچکترین میانه را با نصفه‌ی سمت راست جایگزین و t را به مقدار $\lceil \frac{l_{min}}{۲} \rceil$ کم کنیم که l_{min} طول آرایه متناظر با کوچکترین میانه است. دقت کنید که با هر گام این الگوریتم طول یکی از آرایه‌ها نصف می‌شود (و یا آن آرایه از جواب حذف می‌شود).

برای دسترسی به بزرگترین و کوچکترین میانه هم می‌توانیم از داده‌ساختار heap استفاده کنیم و یک min heap برای نگه‌داری کوچکترین میانه و یک max heap برای نگه‌داری بزرگترین میانه استفاده کنیم.

در هر گام الگوریتم طول یکی از آرایه‌ها نصف می‌شود و میانه متناظر آن آرایه از heap حذف و میانه جدید به heap اضافه می‌شود در نتیجه هر گام الگوریتم از $O(\log k)$ است (طول heap همواره حداکثر k است) و به دلیل این که در هر مرحله طول یکی از آرایه‌ها نصف می‌شود به تعداد $O(\sum_{i=۱}^{k} \log l_i)$ مرحله داریم و پیچیدگی زمانی کلی $O(\log k \times \sum_{i=۱}^{k} \log l_i)$ خواهد بود و مسئله حل می‌شود.

سؤال ۲. (۱۵ نمره) شرک که به نظر می‌رسد حسین را گیر آورده است از او این بار می خواهد که داده ساختاری به نام شبه‌پشته بسازد که سه عملیات زیر را انجام دهد و هزینه زمانی هر عملیات از $O(۱)$ باشد.

- Push(x) : عنصر x را به بالای شبه‌پشته اضافه می‌کند.
- Pop() : عنصر بالای شبه‌پشته را حذف می کند.
- Reverse() : می‌دانیم c یک عدد طبیعی ثابت است. در این دستور اگر تعداد عناصر شبه‌پشته از c بیش‌تر باشد، c عنصر بالایی را برعکس می‌کند.

دقت کنید که عدد c ثابت است و تغییر نمی‌کند.

پاسخ:

مسئله را با استفاده از یک لیست پیوندی و یک پشته حل می‌کنیم.

برای push کردن یک عضو کافی است آن را به انتهای لیست پیوندی اضافه کنیم و اگر پس از اضافه کردن آن عضو، تعداد اعداد داخل لیست پیوندی از c بیشتر شد، عضو ابتدای لیست پیوندی را حذف و داخل پشته منتقل کنیم.

برای pop کردن کافی است عضو انتهای لیست پیوندی را حذف، و در صورتی که پشته خالی نبود، عضو بالای پشته را به ابتدای لیست پیوندی منتقل کنیم.

برای Reverse هم کافی است جای اشاره‌گر ابتدا و انتهای لیست پیوندی با هم جابه‌جا شود که با نگه‌داری یک متغیر مثل flag، که جهت شروع لیست پیوندی از چه سمتی است هم امکان‌پذیر است.

همچنین تمامی عملیات‌ها از $O(۱)$ خواهند بود.

سؤال ۳. (۱۴ نمره) حسین که به تازگی به سوالات شرک پاسخ داده و احساس اعتماد به نفس می‌کند به کامیار سری می‌زند. کامیار که از اعتماد به نفس کاذب حسین شاکی شده است، سوالی سخت برای او مطرح می کند تا مچ او را بخواباند، او از حسین می خواهد توابع زیر را بر اساس نرخ رشدشان از کم به زیاد مرتب کند. به حسین کمک کنید که اینبار نیز از این چالش سربلند بیرون بیاید.

$$f_۱(n)=n^{\pi} \quad f_۲(n)=\pi^n \quad f_۳(n)=\binom{n}{۵} \quad f_۴(n)=\sqrt{۲^{\sqrt{n}}}$$
$$f_۵(n)=\binom{n}{n-۴} \quad f_۶(n)=۲^{\log^۴ n} \quad f_۷(n)=n^{۵(\log n)^۲} \quad f_۸(n)=n^۴\binom{n}{۴}$$

پاسخ:

در ابتدا می‌توانیم مقادیر $\binom{n}{k}$ را با n^k جایگزین کنیم چون پیچیدگی زمانی یکسانی دارند. همچنین با توجه به این که $\binom{n}{k}=\binom{n}{n-k}$ در نتیجه $f_۳(n)=n^۵$، $f_۵(n)=n^۴$، $f_۸(n)=n^۴\times n^۴=n^۸$. درنتیجه ترتیب توابع غیرنمایی به صورت $f_۱(n), f_۵(n), f_۳(n), f_۸(n)$ خواهد بود.

حال توابع $f_۲(n), f_۴(n), f_۶(n), f_۷(n)$ باقی‌مانده‌اند که توابع نمایی هستند. توابع $f_۴(n)=\sqrt{۲^{\sqrt{n}}}=۲^{n^{\frac{۱}{۴}}}$ ، $f_۷(n)=n^{۵(\log n)^۲}=۲^{۵(\log n)^۳}$ و $f_۲(n)=\pi^n=۲^{\log(\pi)n}$ را میتوانیم ساده کنیم و درنتیجه با مقایسه‌ی نماهای این توابع ، ترتیب به صورت $f_۷(n), f_۶(n), f_۴(n), f_۲(n)$ خواهد بود. درنتیجه ترتیب کلی به صورت

$$f_۱(n), f_۵(n), f_۳(n), f_۸(n), f_۷(n), f_۶(n), f_۴(n), f_۲(n)$$

است.

سؤال ۴. (۱۵ نمره) فیونا عاشق پاشاجوک(جوک هایی که پاشا می‌گوید) است. شرک برای اینکه بتواند مانند پاشا جوک بگوید تا دل فیونا را به دست بیاورد ، n فاکتور کمّی را برای پاشاجوک بودن یک جوک در نظر می‌گیرد و برای هر جوک آن‌ها را در یک آرایه n تایی به نام A می ریزد. شرک که در ساخت پاشاجوک ناتوان است سعی در انتقام از پاشا دارد، یک آرایه به او می‌دهد و از او می‌خواهد بیشترین مقدار $j-i$ را پیدا کند به طوری که $A[i]<A[j]$ دقت داشته باشید که مقادیر میان $A[i]$ و $A[j]$ مهم نیستند. به عنوان مثال آرایه A زیر را در نظر بگیرید. $A=\{۱۴,۶,۸,۱,۱۲,۷,۵\}$ الگوریتم شما باید اندازه ۴ را به عنوان خروجی چاپ کند زیرا بیشترین فاصله میان $A[۲]=۶$ و $A[۶]=۷$ است که برابر با ۴ می باشد. در حل این سوال به پاشا کمک کنید.

آ. (۵ نمره) یک الگوریتم با پیچیدگی زمانی $O(n)$ ارائه دهید که مینیمم آرایه پیشوندی $A[۱],\cdots A[k]$. را برای هر k پیدا کرده و به شکل زیر ذخیره کند.

$$MA[k]=min_{i=۱}^{k}A[i]$$

ب. (۱۰ نمره) با استفاده از آرایه $MA[i]$ که در بالا تولید کردید الگوریتمی با پیچیدگی زمانی $O(nlogn)$ ارائه کنید که بیشترین مقدار $j - i$ هنگامی که $A[i] < A[j]$ است را پیدا کند.

پاسخ:

آ)

در این قسمت کافی است MA را به صورت زیر محاسبه کنیم.

$$MA[k] = \begin{cases} A[۱] & k = ۱ \\ \min(MA[k-۱], A[k]) & ۱ < k \leq n \end{cases}$$

ب)

با توجه به تعریف آرایه MA ، این آرایه غیرصعودی است. حال اگر بخواهیم برای یک j ثابت مسئله را حل کنیم، می‌توانیم از جستجوی دودویی برای یافتن کوچکتر i که $a_i < a_j$ است استفاده کنیم. برای این منظور اگر یک m خاص را در نظر بگیریم اگر $MA[m] < A[j]$ باشد کوچکترین i در بازه‌ی $[۱, m]$ و اگر $MA[m] \geq A[j]$ باشد کوچکترین i در بازه‌ی $[m + ۱, j]$ است درنتیجه با استفاده از جستجوی دودویی می‌توانیم برای یک j خاص در زمان $O(\log n)$ کوچکترین i را پیدا کنیم و اگر این عملیات را برای تمام j ها انجام دهیم جواب مسئله ماکسیمم تمامی این مقادیر است و درنتیجه مسئله در $O(n \log n)$ حل می‌شود.

سؤال ۵. (۳۲ نمره) سهیل عاشق مسائل مرتب‌سازی است. کامیار برای کادوی تولد او تصمیم گرفته است به او یک مسئله زیبای مرتب‌سازی بدهد. فرض کنید به شما یک آرایه داده شده است و از شما می‌خواهند اعداد آن را به صورت صعودی مرتب کنید. تنها عملیاتی که می‌توانید روی این آرایه انجام دهید به صورت $Reverse(A, i, j)$ می‌باشد که به این صورت است که در آرایه A بازه‌ی $[i, j]$ را برعکس می کند. به طور مثال در آرایه $A = \{۱, ۵, ۴, ۳, ۲\}$ با انجام دستور $Reverse(A, ۲, ۵)$ آرایه مرتب می‌شود.

آ. (۵ نمره) نشان دهید هر آرایه‌ای را می توان با $O(n)$ بار استفاده از تابع $Reverse$ مرتب کرد.

ب. (۷ نمره) اثبات کنید آرایه‌ای وجود دارد که $\Omega(n)$ عملیات $Reverse$ برای مرتب کردن آن لازم است. (اثبات وجودی کافی است و لازم نیست مثال بزنید)

پ. (۱۰ نمره) در صورتی که هزینه انجام عملیات $Reverse(A, i, j)$ برابر $|i - j| + ۱$ باشد. الگوریتمی برای مرتب‌سازی آرایه‌ای که اعضای آن ۰ یا ۱ هستند ارائه دهید که جمع هزینه عملیات‌های استفاده شده از $O(n \log n)$ باشد.

ت. (۱۰ نمره) با توجه به هزینه عملیات در قسمت قبل الگوریتمی برای مرتب‌سازی هر آرایه دلخواهی ارائه دهید که جمع هزینه عملیات‌های استفاده شده از $O(n\log^{۲} n)$ باشد.

پاسخ:

آ)

اگر c_i را مکان عضو i ام در ترتیب مرتب شده آرایه تعریف کنیم ، آنگاه می‌توانیم در n عملیات آرایه را مرتب کنیم به این صورت که در گام i ام بازه‌ی $[i, c_i]$ را برعکس می‌کنیم، بدیهی است که پس از انجام عملیات i ام اعداد بازه‌ی $[۱, i]$ شامل i عدد مینیمم به صورت مرتب شده خواهد بود و در نتیجه پس از n مرحله آرایه مرتب خواهد شد.

ب)

می‌دانیم هر عملیات $Reverse(A, i, j)$ ، $\binom{n}{۲} = O(n^{۲})$ حالت دارد که کدام بازه را برعکس کند، حال اگر فرض کنیم با تعدادی $Reverse$ از یک آرایه به آرایه‌ی مرتب شده‌اش رسیدیم می‌توانیم با انجام آن عملیات‌ها از انتها به ابتدا از آرایه مرتب شده به آرایه اولیه برسیم.

حال درختی را در نظر بگیرید که رئوس آن متناظر حالات قرارگیری اعداد در آرایه باشند و فرزندان هر راس حالات متفاوتی که با انجام عملیات $Reverse$ روی آن حالت به دست می‌آیند باشند. می‌دانیم اگر اعداد متفاوت باشند $n!$ جایگشت مختلف از قرارگیری این اعداد وجود دارد و اگر ریشه این درخت را متناظر با آرایه‌ی مرتب شده در نظر بگیریم هر راس حداکثر $n^{۲}$ فرزند دارد درنتیجه به میزان $\log_{n^{۲}}(n!)$ ارتفاع برای پوشاندن تمام حالات احتیاج داریم و درنتیجه $\log_{n^{۲}}(n!) = \frac{n\log(n)}{۲\log(n)} = \frac{n}{۲} = \Omega(n)$ عملیات برای این که تمام حالات را بتوان مرتب کرد نیاز است بنابراین آرایه‌ای وجود دارد که $\Omega(n)$ عملیات $Reverse$ برای مرتب کردن آن لازم است.

پ)

از ایده‌ی الگوریتم مرج‌سورت استفاده می‌کنیم ، آرایه را به دو قسمت با طول برابر تقسیم می‌کنیم و هر کدام را به صورت بازگشتی مرتب می‌کنیم ، سپس هر دو آرایه به صورت تعدادی ۰ و سپس تعدادی ۱ خواهد بود. در نهایت می‌توانیم با برعکس کردن بازه‌ای شامل ۱ های نیمه‌ی اول آرایه و ۰ های نیمه‌ی دوم تمام آرایه را مرتب کنیم.

اگر هزینه عملیات‌های انجام شده را با $T(n)$ نشان دهیم، خواهیم داشت $T(n) = ۲T(\frac{n}{۲}) + O(n)$

چون تنها یک عملیات $Reverse$ پس از مرتب کردن دو نیمه انجام خواهیم داد که حداکثر هزینه‌ی n خواهند داشت.

در نهایت تابع $T(n)$ از $O(n\log(n))$ خواهد بود (تحلیل اردر همانند الگوریتم مرج‌سورت است).

ت)

ایده حل این سوال همانند کوییک‌سورت است. ابتدا میانه آرایه را محاسبه می‌کنیم (با استفاده از هر الگوریتم $O(n\log n)$) ، سپس هر عدد را اگر از میانه کمتر بود با ۰ جایگزین می‌کنیم و اگر نه با ۱ جایگزین می‌کنیم حال از الگوریتم قسمت

قبل استفاده می‌کنیم تا آرایه‌ای که شامل اعداد ۰ و ۱ است را مرتب کند. پس از انجام این عملیات نیمه‌ی اول آرایه شامل عدد ۰ و نیمه‌ی دوم آرایه شامل عدد ۱ خواهد بود. سپس دو نیمه را به صورت بازگشتی مرتب می‌کنیم. در نهایت هزینه‌ی عملیات‌های انجام شده به صورت $T(n) = ۲T(\frac{n}{۲}) + O(n \log n)$ خواهد بود که از $O(n \log^۲(n))$ است.

سؤال ۶. (۱۴ نمره) شرک که در ساخت پاشاجوک ها موفق نبوده، برای اینکه بتواند دل فیونا را در اخرین فرصتش به دست بیاورد نیاز دارد یک درخت AVL که درخت مورد علاقه فیونا است را تولید کند. شرک شناختی از درخت های AVL ندارد به همین خاطر از شما می‌خواهد که اعداد زیر را به ترتیب از چپ به راست در یک درخت AVL که در ابتدا خالی است وارد کرده و در هر مرحله شکل درخت را بکشید و هر چرخاندن و جابه‌جایی در درخت را به شکل واضح برای شرک توضیح دهید تا به مراد دلش برسد.

$$\{۹, ۲۷, ۵۰, ۱۵, ۲, ۲۱, ۳۶\}$$

پاسخ:

با استفاده از این لینک می‌توانید جواب را بررسی کنید =)

موفق باشید