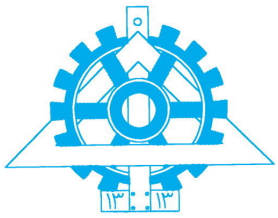
تمرین شماره ۲

درخت ها و داده ساختار های پایه



دانشکده مهندسی برق و کامپیوتر

مهلت تحویل: ۲۸ آبان

ساختمان های داده و الگوریتم - پاییز ۱۴۰۱

استاد: دکتر هشام فیلی

طراح تمرین: **شایان شیبهی**[1]

---

۱. با استفاده از پشته و عملیات مرتبط، عبارت پسوندی زیر را به عبارت میانوندی معادل تبدیل کنید و وضعیت پشته را در نقاط مشخص شده نمایش دهید.

$$acd * g * e/ + be/ c *- fh/i/l * j *- i -$$

پاسخ:

برای تبدیل عبارت پسوندی به میانوندی با استفاده از استک، عبارت داده شده را پیمایش میکنیم و در هر مرحله در صورتی که operand ای را خواندیم آن را در استک پوش میکنیم و در صورتی که operator ای را خواندیم دو عنصر به ترتیب a و b را از بالای استک پاپ کرده، و عبارت $b \ < opt > \ a$ را در استک پوش میکنیم. در اینجا منظور از $< opt >$ همان اپراتور خوانده شده است.

پس از اجرای کامل این الگوریتم، عبارت میانوندی معادل بصورت زیر خواهد بود. دقت کنیدکه در اینجا لزومی به پرانتز گذاری در عبارت خروجی نیست.

$$a + c * d * g / e - b / e * c - f / h / i * l * j - i$$

---

[1] shabihish@gmail.com

در این حالت وضعیت استک در نقاط خواسته شده از چپ به راست بصورت زیر خواهد بود:

1. [a | c*d | g | 2. [a+c*d*g/e | b/e | 3. [a+c*d*g/e-b/e*c |

۲. با استفاده از تنها یک پشته استاندارد، ساختار پشته ای طراحی کنید که علاوه بر عملیات اصلی push و pop، عملیات جدید min را نیز برای گرفتن کوچکترین عنصر موجود در پشته پشتیبانی کند. همچنین، نیاز است تمامی عملیات مطرح شده با پیچیدگی زمانی و حافظه جانبی $O(1)$ توسط ساختار پیاده سازی شده قابل محاسبه باشند.

راهنمایی: میتوانید عناصری از پشته را پیش از push و یا pop کردن تغییر دهید، به شکلی که پشته حاصل برای اجرای عملیات min مناسب باشد.

پاسخ:

در اینجا جهت فراهم کردن امکان پاسخ به کوئری مورد نظر از اردر زمانی $O(1)$، مقادیر پوش و یا پاپ شده به استک را همواره بصورتی تغییر میدهیم که همواره بتوان یا از حافظه مقدار مینیمم قبلی را خواند و یا اینکه از روی عنصر پوش یا پاپ شده مقدار آن مینیمم را آپدیت کرد. در این حالت هر یک از operation های پوش و پاپ را به شکل زیر تعریف میکنیم:

- **Push:** در صورتی که استک خالی است، بصورت استاندارد پوش انجام شده و مقداری با عنوان min در حافظه اصلی برابر با مقدار پوش شده ذخیره میشود. در غیر این صورت، حتما min از قبل مقدار دارد و اگر مقدار مورد نظر برای پوش (x) بزرگتر یا مساوی آن باشد پوش شدن به صورت استاندارد انجام میگیرد ولی در صورتی که این مقدار از min کوچکتر باشد، min به این مقدار جدید آپدیت شده و $2x - min$ روی استک پوش خواهد شد.
- **Pop:** در این حالت فرض میشود که همواره حداقل یک عنصر در استک وجود دارد. حال، برای pop کردن عنصر سر استک (x) را بررسی میکنیم و دو حالت خواهیم داشت: ۱) x از min بزرگتر یا مساوی آن است که در این حالت x را پاپ میکنیم و کار تمام است. ۲) x از min کوچکتر است که در این حالت مقدار min به $2 * min - x$ آپدیت شده، و عنصر مورد نظر پاپ

میشود. دقت کنید که در این حالت مقدار اصلی عنصر پاپ شده همان min قبلی بوده، و مقدار پاپ شده بصورت مستقیم از استک مقداری نسبی است.

نکته: در این سوال با توجه به محدودیت اردر $O(1)$ برای حافظه جانبی، مجاز به استفاده از آرایه و یا استک بصورت موازی نیستید. همچنین، امکان تعریف عناصر استک اصلی بصورت pair های دوتایی از مقادیر اصلی و فرعی وجود نخواهد داشت.

۳. با داشتن یک لیست پیوندی یک طرفه ورودی شامل تعداد **نامشخصی** نود اولیه از اعداد صحیح، الگوریتمی طراحی کنید که لیست پیوندی خروجی دهد که در آن k نود انتخابی بصورت رندوم، پیش از تمامی نود های دیگر لیست ورودی قرار بگیرند. برای این سوال مجاز هستید حداکثر k بار لیست ورودی را با شروع از نود اول پیمایش کنید. توجه کنید که احتمال selection هر نود انتخاب نشده در هر مرحله، باید بصورت یکنواخت یا uniform توزیع شده باشد.

پاسخ:

در شرایطی که تعداد نود های اولیه مشخص بود، به سادگی میتوانستیم با k با پیمایش لیست اصلی و انتخاب یک عدد تصادفی از 1 تا N و در ادامه حذف عنصر انتخابی برای مرحله بعد مسئله را حل کنیم. اما با توجه به اینکه تعداد نود های اولیه مشخص نیست، در اینجا از روشی موسوم به Reservoir Sampling استفاده میکنیم که میتوانید از طریق این لینک آشنایی بیشتری با آن کسب کنید. در این مدل از sampling، در ابتدا فرض میکنیم که نود اول نتیجه (r) انتخاب تصادفی خواهد بود، و سپس در هر مرحله از پیمایش نود های بعد از آن در لیست یک عدد تصادفی از 1 تا n انتخاب میکنیم و در صورتی که این مقدار برابر 1 بود، مقدار r را با نود جدید آپدیت میکنیم. در این حالت با یکبار پیمایش کامل لیست بدون نیاز به داشتن تعداد نود های اولیه میتوان یک نود رندوم را از آن انتخاب و برای k-1 مرحله بعدی حذف کرد. حال برای k-1 مرحله بعدی، میتوان از sampling ساده با شمارش تعداد نود ها در پیمایش اول و یا همان روش Reservoir Sampling استفاده کرد. توجه کنید که با توجه به توضیحات تکمیلی ارائه شده هر انتخاب میبایست در شرایطی انجام گیرد که نود انتخابی مرحله قبل

حذف شده و بنابراین احتمال انتخاب هر نود برای انتخاب های اول تا k ام به ترتیب $\frac{1}{N}$ ، $\frac{1}{N-1}$ ، $\frac{1}{N-2}$ ، ...، و $\frac{1}{N-k+1}$ خواهد بود.

برای مطالعه درباره تابع چگالی احتمال در این متد sampling میتوانید به لینک داده شده مراجعه فرمایید.

برای این سوال، کافیست با انتخاب نود ها بصورت تصادفی، آنها را به ترتیب انتخاب شده از لیست اولیه حذف و به انتهای لیستی ثانویه اضافه کنیم. در نهایت با جایگذاری مقدار head لیست اولیه در tail لیست ثانویه، لیست ثانویه همان خروجی مورد نظر خواهد بود.

۴. به کمک دو پشته، یک ساختار داده صف را پیاده سازی کرده، و هزینه زمانی اعمال enqueue و dequeue را برای این صف محاسبه کنید.

پاسخ:

در اینجا میتوان میتوان از دو حالت با هزینه زمانی زیاد برای عملیات های به ترتیب dequeue و enqueue استفاده کرد. در اینجا حالت اول را توضیح میدهیم ولی حالت دوم نیز بصورتی مشابه قابل پیاده سازی میباشد.

در این حالت برای enqueue کردن، صرفا عنصر مورد نظر را در استک اول push میکنیم و کار تمام است ولی برای dequeue کردن، تمامی عناصر پشته اول به جز عنصر آخر را به پشته دوم انتقال میدهیم. سپس عنصر آخر استک اول را به عنوان عنصر dequeue شده معرفی کرده و pop میکنیم. در نهایت تمامی عناصر را مجددا به ترتیب به پشته اول منتقل میکنیم.

۵. با استفاده از پشته، برای یک آرایه ورودی با طول n از اعداد طبیعی مقدار اولین عدد بزرگتر از هر یک از اعضای این آرایه را با ارائه الگوریتمی از پیچیدگی زمانی $O(n)$ گزارش دهید. برای مثال، برای آرایه ورودی $[3, 4, 14, 2, 18]$، الگوریتم شما باید خروجی $[4, 14, 18, 18, -1]$ را تولید کند.

پاسخ:

در ابتدا عنصر اول را در استک پوش میکنیم، سپس عناصر بعدی را به ترتیب پیمایش میکنیم و برای هر یک از این عناصر (x) دو حالت خواهیم داشت:

- عنصر x از top استک کوچکتر است. کافیست x را در استک push کنیم.
- عنصر x از top استک بزرگتر است. در این حالت عنصر بعدی بزرگتر از top استک و تمامی عناصر بعد از آن که از x کوچکتر هستند همان x خواهد بود. بنابراین تمامی این عناصر را pop کرده، و در نهایت x را در استک پوش میکنیم.

در نهایت، در صورتی که پیمایش آرایه اصلی به اتمام رسید، برای تمامی عناصر باقی مانده در استک عدد بزرگتری وجود نخواهد داشت، و بنابراین 1- را به عنوان خروجی برای آنها در نظر میگیریم.

موفق باشید.