بسمه تعالی

ساختمان داده ـ دکتر علیمی
دانشجو : مهرسا پوریا
شماره دانشجویی : ۹۵۱۰۱۲۴۷

نیمسال اول ۹۷ـ۹۸
سوال تئوری تمرین سری دوم

تئوری آغازی نو

در این سوال برای هر یک از خواسته ها از یک پشته[1] استفاده می‌کنیم ؛جزییات هربار استفاده از این پشته به شرح زیر است.

خواسته اول

$$\text{For each i, maximum j} : A_j < A_i \quad \& \quad 1 \leq j < i$$

برای این حالت پشته را به صورت زیر می سازیم: (متناظر با خطوط ۲۶ تا ۳۵ کد آپلودشده)
ابتدا صفر را وارد پشته می‌کنیم. (عدد متناظر ایندکس صفر را نیز برابر -1 قرار می‌دهیم.)
از ابتدای رشته ورودی، در هر مرحله یک عدد از رشته ورودی را با عنصر متناظر ایندکس بالای پشته مقایسه می‌کنیم، تا زمانی که عنصر متناظر بالای پشته از عدد ورودی بزرگتر یا مساوی است، عنصر بالای پشته را pop می کنیم. هر وقت عنصر متناظر بالای پشته از عدد ورودی کوچکتر شد از حلقه pop کردن خارج می‌شویم.
ایندکس j خواسته برای عدد ورودی برابر عنصر بالای پشته است.
سپس ایندکس عدد ورودی را در پشته وارد می‌کنیم.
الگوریتم را برای عدد بعدی تکرار می‌کنیم.
اثبات درستی :
برای اولین عدد پاسخ درست است چون -1 از هر عدد مثبتی کوچکتر است پس در مرحله مقایسه ایندکس متناظر آن که همان ۰ است، در پشته باقی می‌ماند و سپس به عنوان j برای اولین عنصر معرفی می‌شود. در سایر مراحل هر بار از بالای پشته ایندکس های متناظر با اعدادی که از عدد ورودی بزرگترند را حذف می‌کنیم، اینکار به این علت است چون از اعداد متناظر با آنها از عدد ورودی بزرگترند امکان ندارد به عنوان j مناسب معرفی شوند و هم اینکه برای اعداد بعدی هم اولیت با عدد کوچکتر قبلی است و بزرگترهای قبل آن اهمیتی ندارند. حلقه‌ی pop زمانی متوقف می‌شود که عنصر بالای پشته متناظر با عدد کوچکتری باشد و چون اعداد به ترتیب از اول وارد پشته شده اند؛ عنصر بالای پشته همه‌ی شرایط j گفته‌شده برای آن برقرار است و به درستی به عنوان j اعلام می‌شود.
تحلیل زمانی:
از آنجا که از پشته استفاده کرده‌ایم و حداکثر n بار push و pop انجام می‌دهیم و هرکدام در O(1) صورت می‌گیرد بنابراین پیچیدگی زمانی برابر O(n) خواهد بود.

خواسته دوم

$$\text{for each i, minimum j} : A_j < A_i \quad \& \quad i < j \leq n$$

[1] stack

برای این حالت پشته را به صورت زیر می سازیم: (متناظر با خطوط ۴۰ تا ۵۲ کد آپلودشده)
ابتدا $n+1$ را وارد پشته می‌کنیم. (عدد متناظر ایندکس $n+1$ را نیز برابر -1 قرار می‌دهیم.)
از انتهای رشته ورودی شروع می‌کنیم و در هر مرحله یک عدد از رشته ورودی را با عنصر متناظر ایندکس بالای پشته مقایسه می‌کنیم، تا زمانی که عنصر متناظر بالای پشته از عدد ورودی بزرگتر یا مساوی است، عنصر بالای پشته را pop می کنیم. هر وقت عنصر متناظر بالای پشته از عدد ورودی کوچکتر شد از حلقه pop کردن خارج می‌شویم.
ایندکس j خواسته برای عدد ورودی برابر عنصر متناظر ایندکس بالای پشته است. طبق قرارداد گفته شده برای عدم وجود، اگر بالای پشته برابر $n+1$ بود که معادل وجود نداشتن j است؛ 0 را به عنوان آن اعلام می‌کنیم.
سپس ایندکس عدد ورودی را در پشته وارد می‌کنیم.
الگوریتم را برای عدد قبلی در رشته ورودی تکرار می‌کنیم.
اثبات درستی :
برای آخرین عدد پاسخ درست است چون -1 از هر عدد مثبتی کوچکتر است پس در مرحله مقایسه ایندکس متناظر آن که همان $n+1$ است ، در پشته باقی می‌ماند و سپس طبق قرارداد 0 به عنوان j برای آخرین عنصر معرفی می‌شود. در سایر مراحل هر بار از بالای پشته ایندکس های متناظر با اعدادی که از عدد ورودی بزرگترند را حذف می‌کنیم، اینکار به این علت است چون از اعداد متناظر با آنها از عدد ورودی بزرگترند امکان ندارد به عنوان j مناسب معرفی شوند و هم اینکه برای اعداد قبلی هم اولیت با عدد کوچکتر بعدی است و بزرگترهای بعد آن اهمیتی ندارند. (یعنی به دنبال کوچکترین عدد بعدی هستیم) حلقه‌ی pop زمانی متوقف می‌شود که عنصر بالای پشته متناظر با عدد کوچکتری باشد و چون اعداد به ترتیب از آخر وارد پشته شده اند؛ عنصر بالای پشته همه‌ی شرایط j گفته‌شده برای آن برقرار است و به درستی به عنوان j اعلام می‌شود.
تحلیل زمانی:
از آنجا که از پشته استفاده کرده‌ایم و حداکثر n بار push و pop انجام می‌دهیم و هرکدام در O(1) صورت می‌گیرد بنابراین پیچیدگی زمانی برابر O(n) خواهد بود.

تحلیل زمانی کل

از آنجا که کد از بخش‌هایی مستقل با پیچدگی زمانی O(n) تشکیل شده پس پیچدگی زمانی کل آن برابر O(n) می‌باشد.