بسم رب العالمین

Subject ۹۸۴۱۱۱۳۲
Date

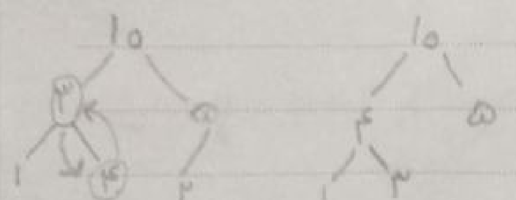
محمد عرفان زارع رستمی

تکلیف ساختمان داده

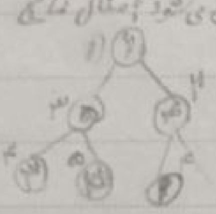
سوال 1)

الف) غلطه . درستش:

۳و۱و۵و۴و۱۵



ب) درسته . می دانیم صف اولویت دار ، داده ها براساس اولویت کارشان انجام می شود . همچنین می شود امثال تابع



داد که Heap هم (عملی از آن) می تواند جزء Priority queue داشت (اعداد بیرون دایره اولویت اند) در مثال

ج) غلطه . این تعریف linkedlist نیست بلکه تعریف queue (صف) است که قاعده ی FIFO در آن

رعایت می شود و عنصری که زودتر می آید ، زودتر خارج می شود و خروج از جلوی صف .

د) غلطه: چون در max heap عنصر Root ، بزرگترین است پس order find-max = $O(1)$. درحالی که اینجا برای

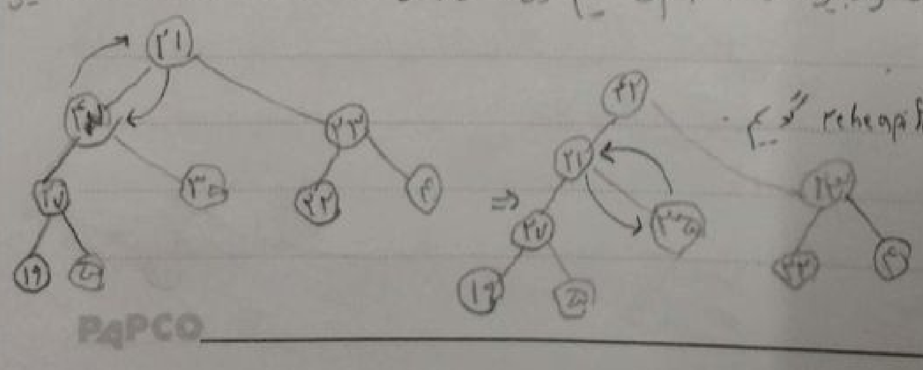
یافتن find min ، Log(n) است ولی برای Insert ، (n log n) است .

سوال ۲)

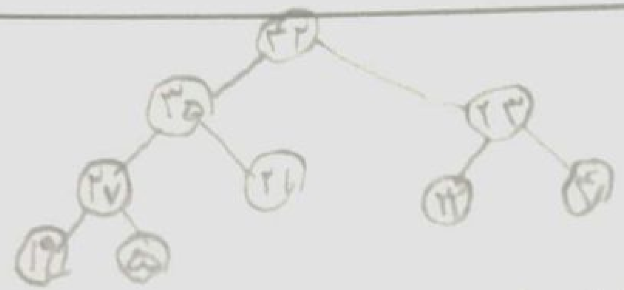
① ابتدا آخرین node را در جای Root می گذاریم ② Root جدید را با فرزند بزرگترش جابجا می کنیم و به سمت پایین درخت

می بریم تا زمانی که عنصر در جای درست خود قرار بگیرد ، ② را انجام می دهیم . این کار وقتی پایان می یابد که عنصر بزرگ تر از child هایش

بزرگتر باشد که به این مراحل reheapification downward گوییم .



→ صفحه بعد



سوال ۳) صف اولویت دار (Priority queue) مشابه stack است که برای بیان اولویت اضافه شده باشد و عنصر با اولویت بالاتر

به عنصر با اولویت پایین‌تر ترجیح داده می‌شود. Stack براساس LiFo و Priority queue براساس اولویت. اساس کار به این صورت می‌شود یک متغیر جدا ایجاد

عنصر آخر را ذخیره کند که بالاترین اولویت را می‌شود شمارنده بالاترین اولویت را ذخیره کرد که به آن key (کلید) گویند. حال دقت کنیم این می‌شود:
(count)

① یک کلاس Stack با Priority queue می‌سازیم علاوه بر صف اولیه یک متغیر شمارنده (count) که داخل آن تنظیم کنیم. ② در تابع Push یک عدد را

دریافت کرده و تعداد را زیاد کرده و count و عنصر موجود را در Priority queue، Push می‌کنیم. ③ تابع به عنوان top می‌بینیم که عنصر ذخیره شده در

بالا که بیشترین عدد جفت ذخیره شده در بالای Priority queue است را برمی‌گرداند. ④ تابع isEmpty هم چک می‌کند که صف خالی می‌ماند یا نه. ⑤ تابع Pop هم

چک می‌کند که صف خالی است یا نه و در صورت خالی نبودن، تعداد را کاهش و Priority queue را Pop می‌کند.

در نهایت با پیاده‌سازی stack و در صورت خالی نبودن آن، قسمت بالای را چاپ می‌کند و آن را Pop می‌کند.

سؤال ۴

دو stack را در نظر می‌گیریم. (مانند in و out) در متغیر هم برای بالای آنها به نام های top_in و top_out نگه داشته ایم

ورودی خروجی نگه داشتن

که top_in بالاترین عنصر ورودی (in) و top_out بالاترین عنصر خروجی (out) می باشد. به وضوح طول stack در in را Length

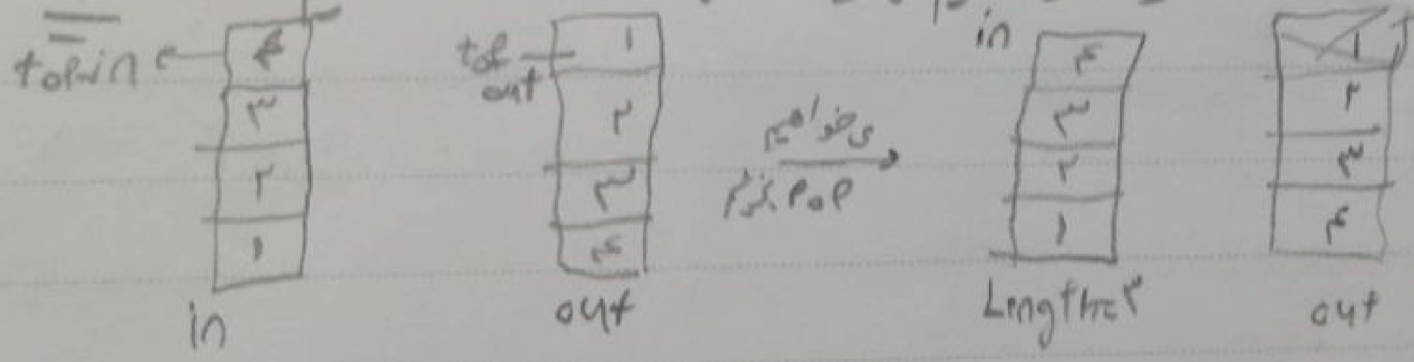
اگر عنصری از in حذف شود، از Length یک واحد کم می‌شود ولی top-in پایین نمی‌آید و تغییر نمی‌کند. top-in

زمانی تغییر می‌کند که یک عنصر در stack، in اضافه شود (push). پس در push عنصر به in اضافه می‌شود و Length

یک واحد زیاد می‌شود و top-in اضافه می‌شود. هنگامی که pop رخ بدهد، میاییم top_in را برابر یک متغیر میانی

(temp) می‌گذاریم و یک for به تعداد Length از ده و عناصر in را به شکل عکس در out می‌ریزیم. حال چون طول

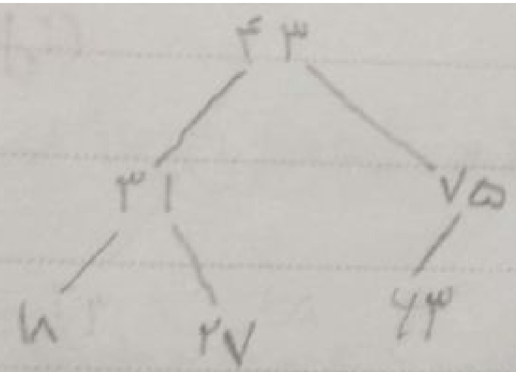
شده بالایک pop را دوباره می‌آییم، عنصر را بر می‌داریم. چون ما Length و top-in را داریم، آرد در ثابت و ماطول



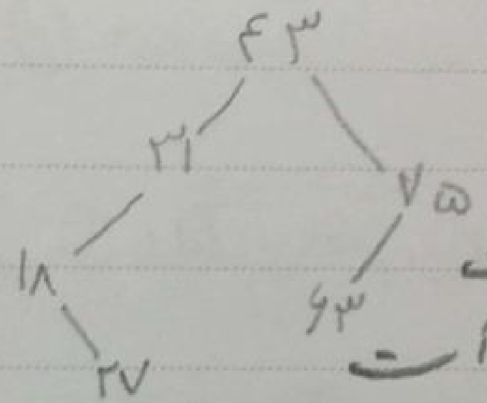
خروجی تابع pop

را داریم. مثال

سؤال ۵) 1 تا



BST نیست



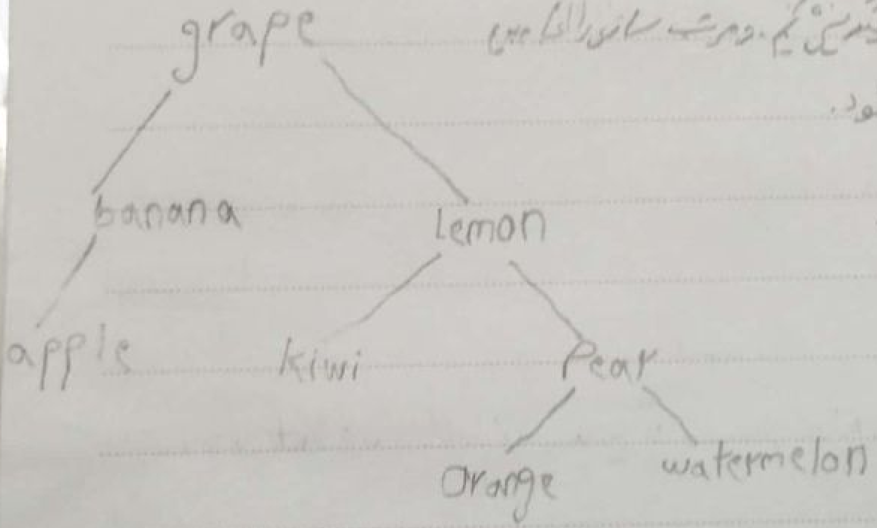
این tree را می‌توان جزو BST دانست
که preorder آن عبارت داده شده است

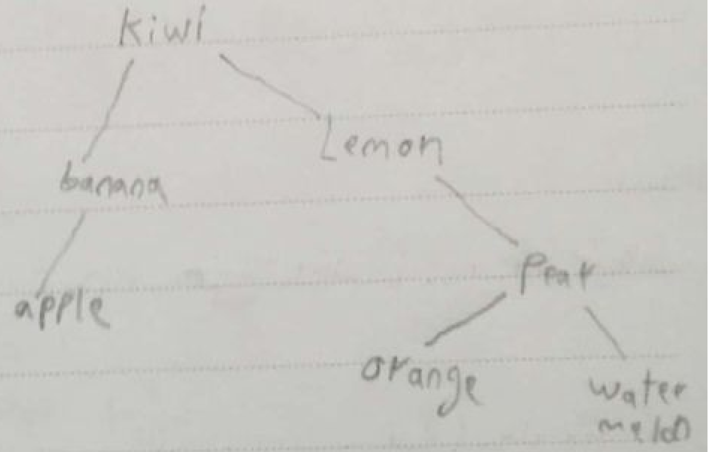
در واقع با توجه به رعایت قوانین BST و اینکه این تنها حالتی می‌باشد که با آن preorder ساخته می‌شود

تنها یک حالت داریم.

سوال ۶)

باحذف grape و اینکه کوچکترین عنصر بزرگتر آز خودش را جایگزین کنیم. درخت سازی را انجام می‌دهیم که بدان successor گویند که kiwi آست. درنتیجه شکل 1 ایجاد می‌شود.





حالت دوم زمانی آست که عنصر بعدی که predecessor آست را جایگزین عنصر Root (ریشه) می‌کنیم که اینجا banana آست.