تمرین سوم ساختمان داده

نیما بهرنگ ۹۶۱۰۰۱۱۴

۷ بهمن ۱۳۹۷

استاد فروغمند

variable length

به طور خلاصه ابتدا اعداد را در اساس طولشان با صورت سطلی سورت می کنیم تا اعداد هم طول در سبد های یکسان باشند و چون جمع طول اعداد n است پس حداکثر همان قدر نیز زمان می برد. حال طول اعداد هر سطر برابر است و کافی است از radix sort استفاده کنیم در اوردر طول کلمات آن ها را سورت کنیم به این صورت که چون هر رقم بین ۰ تا ۹ است از ایده مرتب سازی سطلی استفاده کرده و با شروع از کوچکترین رقم، هر دسته را مرتب می کنیم و به بزرگترین رقم می رسیم. این کار به اندازه ۱۰ برابر طول عددهایی که در آن سطل قرار دارد طول می کشد پس در مجموع نیز به انداز عدد ثابتی ضربدر مجموع طول اعداد که همان n است طول می کشد.

collision Attack

۱ ۰ ۳۱ ۰ ۳۰ ۳۱ ۰ ۳۰ ۳۰ ۳۱

اگر هش ۳۱ به توان n-1 را در نظر بگیریم، به تعداد دو به توان n حالت می توان آن را تولید کرد.

به جای هر کاراکتر کد اسکی آن را می نویسیم

رشته

1000...00

0(31)000...00

0(30)(31)000...00

0(30)(30)(31)0...00

... در اینجا مکان ۳۱ متمایز کننده این رشته ها است و می تواند در هر جایی از رشته باشد که یعنی به طول رشته، رشته متماییز تولید می کند.

hash list

راه حل خلاصه : ذخیره پوینتر لینکدلیست در هش تیبل
برای حل سوال ابتدا یک لینک لیست می سازیم و اعداد را به صورت عادی به لیست پیوندیمان اضافه می کنیم.
پس قابلیت پیمایش به صورت متوالی را در زمان ثابت دارد.
در هش تیبل نیز به ازای هر کلید، پونتری که در لیست پیوندی دارد را نگه می داریم یعنی بعد از استفاده از تابع هش روی عدد مورد نظر، مقدار آدرس پوینتری که به نود متناظرش در لینکدلیست اشاره می کند را در آن خانه هش می گذاریم. پس پیدا کردن یک نود لینکدلیست در زمان ثابت معادل زمان پیدا کردن در هش تیبل است.

Random Tree

Randomized

ذخیره پوینتر لینکدلیست در هش
تابعی بازگشتی تعریف می کنیم که x, k می گیرد و تعداد اعداد بزرگتر از x را بر می گرداند ولی با این شرط که اگر بیشتر از k تا بود،دیگر ادامه نمی دهد و فقط باز می گردد
حال این تابع را روی یک راس که صدا می کنیم اگر خود راس بزرگ تر باشد، k را یکی کم کرده و سپس بچه سمت چپ خود را با (x, k-1) صدا می کند در غیر اینصورت خود آن راس و تمام بچه هایش کوچکتر از x هستند و فقط کافی است صفر را به پدرش بازگرداند
پس در حال بزرگتر بودن باید مقداری که می گیرد را از k-1 کم کند و مقدار حاصل را برای بچه دیگر خود فراخوانی کند یعنی x, k - 1 - r
حال جمع مقدار خود و دو بچه اش را برای پدرش(بازگشتی) می فرستد.
اگر راسی k
اش صفر بود،دیگر کاری نمی کند و باز می گردد. اگر مقدار تابع به ازای راس ریشه صفر باشد،جواب مسئله مثبت وگنه منفی است و ما نیز همچنین در هر گام ما مقدار k را یکی کم می کنیم و یا اگر تغییر نکند از آنجا دیگر ادامه نمی دهیم پس به ازای هر بار کم کردن حداکثر دوبار به حالت بازگرداندن صفر ممکن است برخورد کنیم پس در مجموع حداکثر 3*k عملیات انجام می دهیم که از O(k) است

Pair

از two pointer استفاده می کنیم. هدف این است که همزمان که درخت را به ترتیب از کوچکترین به سمت بزرگترین می پیماییم، چک کنیم که آیا اعدادی که خواسته مسئله را بخواهند پیدا می شود یا خیر.
برای این کار ابتدا کمینه و بیشینه را که با حداکثر ارتفاع درخت عملیات می توان پیدا کرد،میابیم. جواب اگر باشد،بین این دو است.
حال کافیست طبق روش ارائه شده در کلاس یکی یکی رئوس درخت را از بیشینه به سمت کمینه برویم تا هنگامی که مجموع دو عدد کوچکتر از مقدار خواسته شود.آنجا باید پوینتری که سمت کمینه بود را به سمت نودی بزرگتر حرکت دهیم تا مجموع بزرگتر مساوی شود و سپس از پوینتر آخر به سمت ابتدا می رویم تا کوچکتر مساوی شود.
هر جایی که به مساوی برسیم،جواب مسئله است
با این کار و استفاده از روش کلاسی،هر راس یک بار دیده می شود پس اودر n است و حافظه ای به اندازه مقدار ارتفاع برای محاسبه نود بعدی و قبلی نیاز داریم.زیرا هر مسیر از ریشه تا برگ به اندازه ارتفاع درخت نود دارد.