تمرین دوم ساختمان داده

نیما بهرنگ ۹۶۱۰۰۱۱۴

۱۱ آذر ۱۳۹۷

استاد فروغمند

تکراری

```
for(int i = 1; i <= n + 1; i++){
   v = a[i];
   if(v != i){
      if(a[v] == v)
         return v;
      else{
         swap(a[v], a[i]);
         i--;
      }
   }
}
```

الگوریتم بالا روی خانه های جدول جلو می رود و هر عدد را در خانه ی خودش می گذارید یعنی i -> a[i] و اگر در خانه a[i] مقدار درستش باشد، و ما باید همان مقدار را دوباره در آن خانه بگذاریم یعنی از عدد i تکراری داریم

پالیندروم

```
cur = list.head
next = cur.next()
for(int i = 0; i < n/2; i++){
   temp = next.next();
   next.next() = cur;
   cur = next;
   next = temp;
}
if(n % 2 == 1)
   next = next.next()
while(cur != list.head){
   if(cur.v != next.v)
      return not;
   cur = cur.next();
   next = next.next();
}
return true;
```

الگوریتم فوق نصف اول لیست را با اوردر حافظه ۱ به صورت برعکس لیست می کند و سپس روی نصف اول و نصف دوم حرکت می کند و چک می کند که برابر باشند

مولد

جواب این مسیله همان عدد کاتالان است به این صورت که با هر بار اضافه کردن یکی به سمت راست برویم و با هربار حذف کردن، یکی به سمت بالا برویم که طبق فرض کاتالان و فرض مسئله تعداد بالاها نمی تواند از تعداد راست ها بیشتر شود پس جواب انتخاب n از 2n تقسیم بر n + 1 است

$$\frac{\binom{2n}{n}}{n+1}$$

هرمی

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

نیمه مرتب

اعداد را k تا k تا جدا می کنیم
حال 2*k تای اول را با مرج سورت سورت می کنیم
حال k تای دوم و سوم را باهم سورت می کنیم و به همین ترتیب تا انتها چون هر عدد حداکثر k تا با جایگاهش فاصله دارد پس همواره در جایگاه واقعیش قرار می گیرد زیرا طبق الگوریتم ما با k تا بعد و قبل خود مقایسه و مرتب می شود
اوردر الگوریتم برابر $\frac{n}{k} * klg(k) = nlg(k)$ است
اگر عددی در جایگاه درست خود قرار نگیرد،باید بیشتر از k تا با جایگاهش فاصله داشته باشد که با فرض مسئله در تناقض است

مرتب سازی

اعداد را با a. b. c, d, e نمایش می دهیم ابتدا دو عضو اول را با یک مقایسه مرتب می کنیم. حال عضو سوم را با عضو چهارم مقایسه می کنیم. با توجه به تقارن عملیات ها، با این کار به یک چهارم جایگشت های ممکن می رسیم فرض می کنیم ترتیب درست آنها a,b و c,d باشد

حال با مقایسه a, c باز به طور متقارن به نصف جایگشت ها می رسیم زیرا تنها کافیست لیبل a,c و b,d تعویض کنیم تا به نیمه دیگر برسیم

حال ترتیب درست حروف به شکل های زیر باشد

c,a,b,d
c,a,d,b
c,d,a,b

حال اگر a, e را مقایسه کنیم به دو مجموعه

۱. c,a,e,b,d
c,a,b,e,d
c,a,b,d,e
c,a,d,b,e
c,a,e,d,b
c,d,a,e,b
c,d,a,b,e

سپس با مقایسه b,e به دو دسته می رسیم

(آ) c,a,e,b,d
c,a,e,d,b
c,a,d,e,b
c,d,a,e,b

وبا مقایسه e,d به دو دسته دوتایی می رسیم که جواب اولی با b,d و دومی با a,d یکتا مشخص می شود

(ب) c,a,b,e,d
c,a,b,d,e
c,a,d,b,e
c,d,a,b,e

وبا مقایسه b,d به دو دسته دوتایی می رسیم که جواب اولی با d,e و دومی با d,a یکتا مشخص می شود

۲. e,c,a,b,d
c,e,a,b,d
e,c,a,d,b
c,e,a,d,b
e,c,d,a,b
c,e,d,a,b
c,d,e,a,b

سپس با مقایسه c,e به دو دسته می رسیم

(آ) e,c,a,b,d
e,c,a,d,b
e,c,d,a,b

وبا مقایسه b,d به دو دسته می رسیم که یکی یکتا و دیگری با مقایسه d,a یکتا مشخص می شود

(ب) c,e,a,b,d
c,e,a,d,b
c,e,d,a,b

c,d,e,a,b

وبا مقایسه a,d به دو دسته دوتایی می رسیم که جواب اولی با d,b و دومی با d,e یکتا مشخص می شود

انتخابگر

همان کد الگوریتم مرتب سازی سریع را استفاده می کنیم ولی اینبار بعد از پارتیشن کردن،اگر محور، k امین عدد بود یعنی k - ۱ عدد کمتر از آن بود همان محور را بازمی گردانیم در غیر اینصورت اگر کمتر بود مثلا r امین عدد بود، در بین بزرگتر هایش به دنبال k-r امین می گردیم در غیر اینصورت بین کوچکتر هایش به دنبال k امی می گردیم

برای اوردر برنامه کافیست روی اینکه مجور ما در حالت مرتب شده چندمین است حالت بندی کنیم اگر

$$k<r<n$$

را انتخاب کنیم باید تابع را به ازای

$$f(r - 1, k)$$

صدا کنیم و در غیر اینصورت به ازای

$$f(n - r, k - r)$$

که اگر استقرایی بخواهیم حل کنیم می شود امید ریاضی

$$E = \frac{n-1}{n} + \frac{n-2}{n} + ...\frac{1}{n} = \frac{n}{2}$$

که از اوردر خواسته شده است