طراحی الگوریتم

تمرین اول - تقسیم و حل

شهریار عطار و امین یوسفی

۱. نزدیک ترین دو جفت نقطه **۱۵ نمره**

فرض کنید ۵ نقطه به صورت زیر داده شده‌اند:

$$P = \{(1,3),(2,6),(5,1),(4,3),(3,4)\}$$

با کمک روش ارائه شده در اسلایدهای درس، نزدیک ترین جفت نقطه از بین این نقاط را پیدا کنید. مراحل کار را توضیح دهید.

پاسخ :

۱. تعریف مجموعه نقاط

$$P = \{(1,3),(2,6),(5,1),(4,3),(3,4)\}$$

۲. مرتب‌سازی نقاط

ابتدا نقاط را بر اساس مختصات x مرتب می‌کنیم:

$$P_x = \{(1,3),(2,6),(3,4),(4,3),(5,1)\}$$

سپس نقاط را بر اساس مختصات y مرتب می‌کنیم:

$$P_y = \{(5,1),(1,3),(4,3),(3,4),(2,6)\}$$

۳. تقسیم‌بندی نقاط نقاط را به دو بخش تقسیم می‌کنیم:

$$P_L = \{(1,3),(2,6),(3,4)\} \qquad P_R = \{(4,3),(5,1)\}$$

۴. حل بازگشتی

اکنون برای هر بخش (چپ و راست) نزدیک‌ترین جفت نقاط را پیدا می‌کنیم.

۱.۴. نزدیک‌ترین جفت نقاط در بخش چپ P_L

برای حل این بخش نیز باید دوباره به دو مجموعه نقاط زیر تقسیم کرد

$$P_L = \{(1,3),(2,6)\} \qquad P_R = \{(3,4)\}$$

که کمترین فاصله برای P_L برابر با $\sqrt{10}$ می باشد و در بخش راست جفت نقطه ای نداریم. حال اگر جواب های این دو بخش را با یکدیگر ترکیب کنیم داریم که:

$$d((1,3),(3,4)) = \sqrt{(3-1)^2+(4-3)^2} = \sqrt{4+1} = \sqrt{5} \approx 2.24$$
$$d((2,6),(3,4)) = \sqrt{(3-2)^2+(4-6)^2} = \sqrt{1+4} = \sqrt{5} \approx 2.24$$

بنابراین نزدیک‌ترین جفت نقاط در بخش چپ برابر است با $(1,3)$ و $(3,4)$ یا $(2,6)$ و $(3,4)$ با فاصله تقریبی 2.24.

۲.۴. نزدیک‌ترین جفت نقاط در بخش راست P_R

در مجموعه نقاط $P_R = \{(4,3),(5,1)\}$ فاصله بین این دو نقطه را محاسبه می‌کنیم:

$$d((4,3),(5,1)) = \sqrt{(5-4)^2+(3-1)^2} = \sqrt{1+4} = \sqrt{5} \approx 2.24$$

بنابراین نزدیک‌ترین جفت نقاط در بخش راست همین دو نقطه $(4,3)$ و $(5,1)$ هستند با فاصله 2.24.

۵. پیدا کردن جفت نقاط نزدیک میان دو بخش

حال باید بررسی کنیم که آیا جفت نقاط نزدیک‌تری میان نقاط دو بخش وجود دارد یا خیر. برای این منظور، نواری به عرض $2 \times \delta$ در امتداد محور x بررسی می‌کنیم، که δ کوچکترین فاصله‌ای است که تاکنون به دست آورده‌ایم.

$$\delta = 2.24$$

حال باید از P_y استفاده کرد و هر نقطه را با تعدادی نقاط بعدی آن مقایسه کرد تا نزدیک ترین جفت نقطه را پیدا کرد. این این کار را انجام دهیم متوجه می شویم که کمترین فاصله بین نقاط برابر است با:

$$d((3,4),(4,3)) = \sqrt{(4-3)^2+(3-4)^2} = \sqrt{1+1} = \sqrt{2} \approx 1.41$$

$$d = \sqrt{2} \approx 1.41$$

۲. جا به جایی ۲۰ نمره

آرایه‌ی A به طول n داده‌شده است؛ شبه کدی بنویسید که تعداد عناصری را که از سه برابر عنصر پیش از خود کمتر است را در مرتبه‌ی زمانی $\mathcal{O}(n \log n)$ برگرداند. به عبارتی تعداد j و i ها به صورتی که:

$$j > i, \, 3A[i] > A[j]$$

پاسخ :

همانند merge sort تنها با این تفاوت که نیاز به دو تابع مرج داریم یکی merge عادی یکی هم مانند:

```
function merge1(left_begin, left_end, right_begin, right_end, result_begin):
        ans = empty list

        while left_begin <= left_end and right_begin <= right_end:
            if *left_begin < *right_begin:
                min = *left_begin
                add min to ans
                increment left_begin
            else:
                min = *right_begin
                add min to ans
                increment right_begin

        if left_begin > left_end and right_begin <= right_end:
            while right_begin <= right_end:
                min = *right_begin
                add min to ans
                increment right_begin

        if right_begin > right_end and left_begin <= left_end:
            while left_begin <= left_end:
                min = *left_begin
                add min to ans
                increment left_begin

    function merge2(left_begin, left_end, right_begin, right_end, number_of):
         ans = empty list

        while left_begin <= left_end and right_begin <= right_end:
            if *left_begin <= 3 * (*right_begin):
                min = *left_begin
                add min to ans
                increment left_begin
            else:
                increment number_of by (left_end - left_begin + 1)
                min = *right_begin
                add min to ans
                increment right_begin
```

که در آن هر گاه عضوی از آرایه‌ی دوم انتخاب شود، یعنی به اندازه‌ی آرایه اول وارونگی داشتیم و پاسخ را زیاد می‌کنیم، چون برای مرحله بعدی آرایه باید مرتب شده باشد از مرج عادی استفاده کرده و برای مرحله بعد از دو آرایه مرتب یک آرایه مرتب می‌سازیم و تابع مرج سورت هم به صورت زیر است که در آن $numberOf$ تعداد وارونگی‌ها در آرایه‌ی $wannasort$ است.

```
function merge_sort(wanna_sort, left, right, numberOf):
    if left == right:
        return

    mid = left + (right - left) / 2

    merge_sort(wanna_sort, left, mid, numberOf)
    merge_sort(wanna_sort, mid + 1, right, numberOf)

    merge2(wanna_sort.begin() + left,
           wanna_sort.begin() + mid,
           wanna_sort.begin() + mid + 1,
           wanna_sort.begin() + right, number_of)

    merge1(wanna_sort.begin() + left,
           wanna_sort.begin() + mid,
           wanna_sort.begin() + mid + 1,
           wanna_sort.begin() + right,
           wanna_sort.begin() + left)
```

۳. باشگاه مشت‌زنی ۲۰ نمره

باشگاه مشت‌زنی محله امیرآباد، n عضو دارد. هر عضو این باشگاه، یک درجه دارد که نشان دهنده قدرت آن عضو می‌باشد. مربی باشگاه می‌خواهد یک مسابقات داخلی بین افراد باشگاه برگزار کند. این مسابقه بدین صورت است که اعضای باشگاه در یک صف قرار می‌گیرند و هرکس به افرادی که در سمت راستش قرار دارند مشت می‌زند. اگر قدرت فرد مشت زننده از کسی که مشت را دریافت می‌کند بیشتر باشد، فرد دریافت کننده مشت بیهوش می‌شود. به ازای هر شخص تعیین کنید که چند نفر بر اثر مشت او بیهوش می‌شوند. هزینه راه حل شما باید از $O(n \log n)$ باشد.

پاسخ :

در ابتدا برای نشان دادن نتیجه راه‌حل، آرایه‌ای به نام $count$ با n عضو که همگی مقدار ۰ دارند، ایجاد می‌کنیم که عضو iام آن نشان دهنده تعداد افرادی است که با مشت نفر iام در صف بیهوش می‌شود. برای حل این سوال از مرتب‌سازی ادغامی استفاده می‌کنیم. در حین ادغام دو زیرآرایه، اعضای زیرآرایه سمت راست، در آرایه اصلی، همگی در سمت راست اعضای زیرآرایه سمت چپ قرار می‌گیرند، پس می‌توان از این ویژگی برای حل این سوال استفاده کرد. ادامه راه حل را با استفاده از یک مثال بیان می‌کنیم:

فرض کنید قدرت و ترتیب قرارگیری اعضا به صورت آرایه $[5, 2, 6, 1]$ است. ابتدا آرایه را به زیرآرایه‌های کوچکتر می‌شکنیم تا به جایی می‌رسیم که هر زیرآرایه یک عضو دارد. پس از این مرحله، زیرآرایه‌ها به صورت $[5], [2], [6], [1]$

می‌شوند. می‌دانید که برای ادغام دو زیرآرایه، عضو اول آنها با یکدیگر مقایسه می‌شود و عضو کوچکتر در آرایه ادغامی قرار می‌گیرد. حال در حین ادغام کردن دو زیرآرایه، اگر عضو انتخابی از زیرآرایه سمت راست باشد، همه اعضای زیرآرایه چپ، یکی به مقدار خانه متناظرشان در آرایه $count$ اضافه می‌شود. پس از این عملیات، زیرآرایه‌ها به صورت $[2,5], [1,6]$ و آرایه $count = [1,0,1,0]$ می‌شوند.

حال حین ادغام دو زیرآرایه موجود، در ابتدا عدد 1 انتخاب می‌شود، پس به خانه متناظر اعداد 2 و 5 در آرایه $count$ یکی اضافه می‌شود و این آرایه به صورت $count = [2,1,1,0]$ می‌شود. سپس به این علت که عدد 6 از 2 و 5 بزرگتر است، دو عضو بعدی که در حین ادغام انتخاب می‌شوند، از زیر آرایه سمت چپ هستند و در این شرایط آرایه $count$ تغییری نمی‌کند. سپس عدد 6 انتخاب می‌شود که در اینجا نیز چون زیرآرایه سمت چپ خالی است، آرایه $count$ تغییری نمی‌کند. پس جواب نهایی برابر $count = [2,1,1,0]$ می‌شود.

این روش جواب درست را به ما می‌دهد اما به علت اینکه هربار باید مقادیر آرایه $count$ تغییر داده شود، اردر الگوریتم $O(n^2)$ می‌شود. برای حل این مشکل، یک متغیر به نام $temp$ و با مقدار اولیه صفر ایجاد می‌کنیم و هربار که در حین ادغام یک عضو از زیرآرایه سمت راست انتخاب می‌شود، مقدار آن را یکی اضافه می‌کنیم. هر زمان که عضو انتخابی از زیرآرایه سمت چپ بود، به خانه متناظر همه اعضای زیرآرایه سمت چپ عدد $temp$ را اضافه کرده و مقدار این متغیر را دوباره صفر می‌کنیم.

برای درک بهتر، فرض کنید زیرآرایه‌ها به صورت $[2,5,6], [9,11,15]$ باشند؛ حال برای ادغام دو زیرآرایه، عضو انتخابی هربار از زیرآرایه سمت راست است و در نتیجه هربار باید خانه متناظر اعضای زیرآرایه سمت چپ را در آرایه $count$ تغییر دهیم. برای جلوگیری از این کار، تا زمانی که عضو انتخابی از زیرآرایه سمت راست است، متغیر $temp$ را زیاد می‌کنیم تا اینکه عضو انتخابی از زیرآرایه چپ شود. در این مثال، عضو انتخابی از زیرآرایه سمت چپ است تا اینکه اعضای زیرآرایه سمت چپ همگی در آرایه ادغامی قرار گیرند پس مقدار متغیر $temp$ برابر 3 می‌شود و این مقدار، به صورت یکجا به خانه متناظر اعضای زیرآرایه سمت چپ اضافه می‌شود. با این کار، هزینه راه حل از $O(nlogn)$ می‌شود.

۴. تیم پرجمعیت ۲۵ نمره

باشگاه آقای امینی n عضو دارد. هر عضو این باشگاه، یک درجه دارد که نشان دهنده قدرت آن عضو می‌باشد. اعضای باشگاه در یک صف کنار یکدیگر قرار گرفته‌اند. آقای امینی می‌خواهد یک گروه از آن‌ها را برای مسابقات انتخاب کند. اعضای این گروه باید در صف از چپ به راست کاملا صعودی باشند. هرچه تعداد اعضای گروه بیشتر باشد، شانس قهرمانی آن‌ها در مسابقات بیشتر است. آقای امینی که به علت قهرمانی‌های متعدد، مسابقات برایش کسل‌کننده شده است، برای انتخاب گروه، شرط جدیدی وضع می‌کند. این شرط این است که اختلاف درجه هر دو عضو متوالی این گروه، حداکثر به اندازه k باشد. به آقای امینی کمک کنید که بهترین گروه واجد شرایط را انتخاب کند. راه‌حل شما باید پیچیدگی زمان اجرای $O(n \log n)$ داشته باشد.
(راهنمایی: از درخت دودویی استفاده کنید)

پاسخ :

از یک $segment\ tree$ برای حل سوال استفاده می‌کنیم. برای راحتی، ساختار درخت را به صورت یک لیست که ایندکس‌هایش از ۱ آغاز می‌شوند در نظر می‌گیریم که همه خانه‌های آن در ابتدا با 0 پر شده است. مقدار ایندکس iام نشان‌دهنده تعداد اعضای گروهی است که بیشترین عضو را دارد و آخرین عضو آن گروه عدد i است.

از عضو اول صف شروع می‌کنیم. فرض می‌کنیم درجه آن فرد، i باشد. حال مقدار $SEGTree[i]$ را برابر $1 + max(SEGTree[i-k:i])$ قرار می‌دهیم. $max(SEGTree[i-k:i])$ نشان دهنده گروه با بیشترین عضو، که با عددی که درحال بررسی آن هستیم، حداکثر kتا اختلاف دارد، می‌باشد. به این مقدار یک عدد اضافه می‌شود چون عددی که در حال بررسی آن هستیم نیز به گروه اضافه می‌شود.

این کار را برای همه اعضای صف انجام می‌دهیم و جواب نهایی، برابر بیشترین مقدار ذخیره شده در درخت می‌باشد.
محاسبه هر یک از $max(SEGTree[i-k:i])$ ها به علت استفاده از $segment\ tree$ در $O(logn)$ انجام می شود (انجام یک $rangequery$ در این درخت در اردر $O(logn)$ انجام می‌شود). حال از آنجا که این کار را n بار انجام می‌دهیم پس اردر $O(nlogn)$ است.
توضیحات بیشتر برای ساخت و استفاده از $segment\ tree$.

۵. عضو متمایز ۲۵ نمره

یک مجموعه به نام A داریم که دارای n عضو می‌باشد. می‌خواهیم مقادیر داخل این مجموعه را به صورتی پر کنیم که به ازای هر زیر رشته در آن، حدقل یک عضو وجود دارد که مقدار آن با بقیه مقادیر داخل آن زیر دنباله متفاوت است. به عبارت دیگر:

$$\forall i, j\, (i \leq j \implies \exists k\, (i \leq k \leq j \wedge \forall t\, (i \leq t \leq j \wedge t \neq k \implies A[k] \neq A[t])))$$

نشان دهید که می توان مقادیر داخل مجموعه را با $\lceil \log_2(n) \rceil + 1$ عدد متفاوت پر کرد، به طوری که شرط گفته شده نقض نشود.

به طور مثال فرض کنید یک مجموعه به طول 5 داریم، اگر خانه‌های آرایه را به شکل زیر پر کنیم هر زیر رشته دارای حداقل یک عضو است که از بقیه متمایز است:

$$[1, 2, 3, 1, 2]$$

اگر سعی کنید این آرایه را با تعداد کمتری عدد متفاوت پر کنید همواره زیر رشته‌ای پیدا خواهید کرد که در آن عضوی که از بقیه متمایز باشد وجود ندارد.

پاسخ :

کافی است که ابتدا عضو وسط مجموعه را انتخاب کنیم، مجموعه را به دو بخش تقسیم کنیم، هر دو بخش را با اعداد مناسب پر کنیم، سپس عضو وسط را با یک عدد جدید اختصاص می‌دهیم. بدین صورت هر زیردنباله‌ای انتخاب شود یا شامل عضو وسط می باشد (که در این صورت همان عضو وسط عضو متمایز ما می‌باشد) و یا زیر دنباله شامل عضو وسط نیست، که در این صورت در یکی این از دو بخش راست یا چپ افتاده که چون دوباره به نحوی مقداردهی شده که شرایط مسئله را نقض نمی‌کند پس قطعا عضوی وجود دارد که متمایز است. حال برای اثبات اینکه با $\lfloor \log_2(n) \rfloor + 1$ عدد می‌توان آن را پر کرد از استقرا استفاده می‌کنیم. (برای این بخش از 2^k استفاده خواهیم کرد که راحت‌تر باشد.)

شرط پایه: می‌دانیم که برای مجموعه با طول 1 فرمول داده شده درست است و برای آن به 1 عدد نیاز داریم.

گام استقرا: فرض می‌کنیم برای 2^k درست است حال می‌خواهیم ثابت کنیم برای 2^{k+1} نیز درست است. اگر

عضو وسط را حذف کنیم و به دو بخش تقسیم کنیم حداکثر طول هر بخش $\lfloor(2^{k+1}/2)\rfloor$ می‌باشد. می‌دانیم که در شرط استقرا صدق می‌کند پس تعداد اعداد مورد نیاز جدید ما برابر است با تعداد اعدادی که برای این بخش نیاز بود به علاوه عدد جدید برای عضو وسط:

$$(\lfloor \log_2(\lfloor(2^{k+1}/2)\rfloor)\rfloor + 1) + 1 = (\lfloor \log_2(2^k)\rfloor + 1) + 1 = k + 2 = \lfloor \log_2(2^{k+1})\rfloor + 1$$

و از آنجا که برای 2^k این فرمول صدق می‌کند، پس برای مجموعه با طول کمتر از آن نیز می‌توانیم با همین تعداد عدد، مجموعه را پر کنیم. پس در نتیجه برای مجموعه‌هایی به طول بین 2^k و 2^{k-1} نیز می‌توانیم مجموعه را با $k+1$ عدد پر کنیم.

توجه داشته باشید که این سوال یک راه حل نیز دارد که در آن از سقف به جای کف استفاده می‌شود و برای راحتی شما در صورت سوال از سقف استفاده شده اما نوشتن هر دو راه نمره را می‌گیرد.