به نام یگانهٔ هستی‌بخش

دورهٔ آموزشی المپیاد کامپیوتر

برنامه‌سازی پویا[۱]

ما قبلاً دیده‌ایم که می‌توان یک مسئلهٔ کلی را شکست و به یک سری زیرمسئله تبدیل کرد که با حل این زیرمسئله‌ها و ادغام جواب آنها می‌توان جواب مسئلهٔ اصلی را بدست آورد. خیلی اوقات ساختار یک مسئله ما را برای شکستن آن راهنمایی می‌کند؛ به یک سری مسئلهٔ مستقل می‌رسیم و آنها را حل می‌کنیم و آنها ما را برای حل مسئلهٔ اصلی کمک می‌کنند. ولی خیلی اوقات ما به یک سری زیرمسئلهٔ مستقل نمی‌رسیم و بلکه این زیرمسائل خیلی اوقات اشتراک زیادی با هم دارند. در این صورت وقتی مستقلاً این زیرمسئله‌ها را حل می‌کنیم، حالات زیادی اتفاق می‌افتند که ما در هر زیرمسئله آنها را حل می‌کنیم؛ در حالی که یک بار حل کردن آنها کافی بوده است. در واقع در صورتی که زیرمسئله‌هایی که ما داریم مستقل از یکدیگر نباشند، با ادامهٔ این عمل زیرمسئله‌های زیادی پیدا می‌شوند که یکسان هستند ولی ما همهٔ آنها را مستقلاً یک بار حل می‌کنیم. اگر بتوانیم این جور هزینه‌ها را کم کنیم و یا اصلاً از بین ببریم، می‌توان به الگوریتم‌های بسیار بهتر و قوی‌تری رسید. ایدهٔ اصلی الگوریتم‌های پویا[۲] نیز همین نکتهٔ ساده می‌باشد: «پرهیز از محاسبهٔ دوبارهٔ چیزهای یکسان» . معمولاً با نگهداری جدولی از داده‌های بدست آمده که از زیرمسئله‌هایی که ایجاد کرده‌ایم آن را پر می‌کنیم، می‌توانیم به این مهم دست پیدا کنیم.

روش پویا یک روش پایین‌به‌بالا می‌باشد. ما معمولاً از آسان‌ترین و مشخص‌ترین زیرمسائل کار را شروع می‌کنیم. با ادغام کردن جواب آنها به جواب مسئلهٔ بزرگ‌تر می‌رسیم، و این کار را ادامه می‌دهیم تا آخر به جواب مسئلهٔ اصلی برسیم. روش «تقسیم و حل»[۳] یک روش بالابه‌پایین است. وقتی یک مسئله را می‌خواهیم با این روش حل کنیم، فوراً به مسئلهٔ اصلی حمله می‌کنیم، سپس آن را به یک سری زیرمسئله تقسیم کرده و دوباره به این زیرمسائل حمله می‌کنیم و این کار را ادامه می‌دهیم تا مسئله حل شود. درواقع می‌توان روش «تقسیم و حل» را جلوهٔ الگوریتمی استقرای ریاضی پنداشت.

---

[۱] مقاله آموزشی فوق توسط محمد مهینی برای دوره آموزشی المپیاد کامپیوتر جمع‌آوری شده است.

[۲] برنامه‌ریزی پویا (Dynamic Programming) که گاهی با اصطلاح «داینامیک» از آن یاد می‌شود، یک فن توانا در حل مسائل الگوریتمی است و موضوع این نوشتار می‌باشد.

[۳] Divide and Conquer method

مسئلهٔ اوّل: دنبالهٔ فیبوناچی ..... $O(n)$

مقدار تابعی که به شکل زیر تعریف می‌شود را برای عدد n بدست آورید:

$$f(n) = \begin{cases} f(n-۱) + f(n-۲) & n \geq ۲ \\ ۱ & n = ۰, ۱ \end{cases}$$

اگر از تابع زیر برای محاسبهٔ این تابع استفاده کنیم، زمان اجرای آن چقدر است؟

function F(n)

```
if n < 2 then return 1
else return F(n-1) + F(n-2)
```

مسئلهٔ دوم: ترکیب ..... $O(nk)$

مقدار ترکیب k از n را با استفاده از رابطهٔ زیر بدست آورید:

$$\binom{n}{k} = \begin{cases} \binom{n-۱}{k-۱} + \binom{n-۱}{k} & ۰ < k < n \\ ۱ & otherwise \end{cases}$$

اگر از تابع زیر برای محاسبهٔ این ترکیب استفاده کنیم، زمان اجرای آن چقدر است؟

function C(n,k)

```
if k = 0 or k = n then return 1
else return C(n-1,k-1) + C(n-1,k)
```

روش پویا معمولاً برای حل مسائل بهینه‌سازی استفاده می‌شود. یعنی مسائلی که می‌خواهیم مقدار یک تابع را کمینه یا بیشینه کنیم. معمولاً به جوابی که مقدار بهینه را دارد، جواب بهینه گفته می‌شود.

ساختار یک الگوریتم پویا می‌تواند به مراحل زیر شکسته شود:

۱) شناخت ساختار جواب بهینه؛

۲) به طور بازگشتی مقدار جواب بهینه را پیدا کنیم؛

۳) مقدار جواب بهینه را به ترتیب و با مدل پایین به بالا محاسبه کنیم؛ و

۴) ساختن جواب بهینه از اطلاعات بدست آمده.

مسئلهٔ سوم: ضرب ماتریسها ..... $O(n^۳)$

فرض کنید دنباله‌ای از ماتریسها به صورت $A_۱ A_۲ \dots A_n$ در اختیار داریم به گونه‌ای که ضرب $A_۱ \times A_۲ \times \dots \times A_n$ قابل تعریف باشد. هزینهٔ ضرب ماتریس A در ماتریس B یعنی $A \times B$، اگر A ماتریسی $m \times k$ و B ماتریسی $k \times p$ باشد، mkp می‌باشد. می‌خواهیم طوری بین این ماتریسها پرانتزگذاری کنیم که کمترین هزینهٔ ممکن را برای یافتن جواب ضرب $A_۱ \times A_۲ \times \dots \times A_n$ بپردازیم. مثلاً برای $n = ۴$، $(A_۱ \times (A_۲ \times A_۳)) \times A_۴$ یک نوع پرانتزگذاری می‌باشد که در آن ابتدا $A_۲ \times A_۳$ محاسبه شده و سپس $A_۱ \times (A_۲ \times A_۳)$ و در نهایت $A_۴$ در جواب آنها ضرب می‌شود تا جواب نهایی بدست آید. کمترین هزینهٔ لازم و نحوه پرانتزگذاری را بدست آورید.

سؤال: اگر تمام حالات پرانتزگذاری را وارسی کنیم، زمان اجرا برابر است با:

$$T(n) = \frac{۱}{n}\binom{۲n-۲}{n-۱}$$

مسئلهٔ چهارم: کوتاه‌ترین مسیر .....

آیا الگوریتم‌های دایکسترا[۴] و فلوید–وارشال[۵] برای یافتن کوتاه‌ترین مسیر، الگوریتم‌های پویا هستند؟

مسئلهٔ پنجم: بزرگ‌ترین زیردنبالهٔ مشترک ..... $O(n^۲)$

دو دنبالهٔ $X = \langle x_۱, \dots, x_n \rangle$ و $Y = \langle y_۱, \dots, y_m \rangle$ داده شده‌اند. هدف پیدا کردن بیشترین تعداد اندیسهای $i_۱ < i_۲ < \dots < i_k$ می‌باشد به طوری که $x_{i_l} = y_{i_l}$ $(\forall l : ۱ \le l \le k)$ باشد. روشی برای بدست آوردن بیشترین تعداد اندیس پیدا کنید.

مسئلهٔ ششم: بزرگ‌ترین زیردنبالهٔ غیرنزولی ..... $O(n^۲)$

یک الگوریتم برای پیدا کردن بزرگ‌ترین زیردنبالهٔ غیرنزولی از دنبالهٔ اعداد $X = \langle x_۱, \dots, x_n \rangle$ ارائه دهید.

مسئلهٔ هفتم: بزرگ‌ترین زیردنبالهٔ غیرنزولی ..... $O(n \lg n)$

یک الگوریتم برای پیدا کردن بزرگ‌ترین زیردنبالهٔ غیرنزولی از دنبالهٔ اعداد $X = \langle x_۱, \dots x_n, \rangle$ ارائه دهید.

---

[۴] Dijkestra

[۵] Floyd-Warshall

مسئلهٔ هشتم: فروشندهٔ دوره‌گرد در فضای اقلیدسی دونوا[۶] ..... $O(n^۲)$

هدف در مسئلهٔ فروشندهٔ دوره‌گرد پیدا کردن کوچک‌ترین دوری است که n نقطه را که در فضای اقلیدسی هستند به هم وصل کند، می‌باشد. فرض کنید این مسیر باید به طوری باشد که این دور از چپ‌ترین نقطه شروع شود و به سمت راست برود (به چپ برنگردد) تا به راست‌ترین نقطه برسد و از آن نقطه برگردد و در برگشت نیز فقط به چپ برود (به راست برنگردد). حال الگوریتمی ارائه دهید که این کوچک‌ترین دور را برای n نقطه در فضای اقلیدسی پیدا کند.

ایدهٔ حل: $p[i]$ را برابر کوچکترین مسیر که از راس i ام (به ترتیب مرتب‌شده از چپ به راست) به رأس ۱ ام برود (فقط به چپ حرکت کند) و بعد از رأس ۱ به راس $i+۱$ ام برود و فقط به راست حرکت کند و همچنین در این مسیر از تمام نقاط بین ۱ تا $i+۱$ دقیقاً یک بار بگذریم.

مسئلهٔ نهم: نوشتن متن ..... $O(n^۲)$

فرض کنید یک متن به ما داده شده است که دارای n کلمه است و کلمهٔ i ام آن دارای طول l_i می‌باشد. همچنین ما یک دفتر داریم که در آن یک سری سطر قرار دارد. در هر سطر از این دفتر می‌توان M حرف نوشت. اگر در خطی هر دو کلمه i ام و $i+۱$ ام قرار داشتند، باید بین این دو کلمه یک فاصله قرار داده شود. به این ترتیب اگر کلمات i ام تا j ام در یک سطر بیایند $(i \le j)$ تعداد حروفی که در این خط هست $j-i+\sum_{k=i}^{j} l_k$ می‌باشد. تعداد حروفی که هر خط می‌نویسیم نباید از M بیشتر شود. هدف کمینه کردن مجموع تعداد فاصله‌های خالی می‌باشد که در انتهای سطرهایی که متن را در آنها نوشته‌ایم قرار دارد. مثلاً اگر کلمات i ام تا j ام در یک سطر بیایند $(i \le j)$ تعداد فاصله خالی که در این خط هست $M-j+i-\sum_{k=i}^{j} l_k$ می‌باشد. الگوریتمی برای پیدا کردن روش بهینهٔ نوشتن این متن پیدا کنید.

مسئلهٔ دهم: مهمانی آقای رئیس ..... $O(n)$

رئیس یک شرکت قصد دارد یک مهمانی برپا کند. در این شرکت هر نفر دقیقاً یک رئیس دارد به غیر از خود رئیس شرکت؛ به این ترتیب ساختار این شرکت به صورت یک درخت ریشه‌دار است که ریشهٔ آن رئیس شرکت و پدر هر رأس، رئیس آن شخص است. همچنین اگر نفر x که در این مهمانی بیاید، شرکت به اندازهٔ c_x پیشرفت می‌کند. طبق تحقیقاتی که انجام شده رئیس شرکت مقدار c_x ها را پیدا کرده است. وی همچنین دوست ندارد که یک نفر از کارکنان شرکت به همراه رئیسش در این مهمانی دعوت شوند. حال هدف او دعوت کردن تعدادی از کارکنان شرکت است که بعد از این مهمانی شرکت بیشترین پیشرفت را داشته باشد. الگوریتمی برای

[۶] Bitonic

محاسبهٔ افرادی که لازم است در مهمانی دعوت شوند تا شرکت بیشترین پیشرفت را داشته باشد بدست آورید.

مسئلهٔ یازدهم: حرکت روی صفحهٔ شطرنجی ..... $O(n^۲)$

در یک صفحهٔ شطرنجی $n \times n$، در هر خانهٔ (i,j) آن مقداری پول به اندازهٔ $p(i,j)$ قرار دارد. ما می‌خواهیم از یکی از خانه‌های پایینی این صفحه شروع کرده و به سمت بالا برویم به طوری که بیشترین پول ممکن را جمع کنیم. ما هرگاه به یک خانه رفتیم می‌توانیم پولی که در آن خانه می‌باشد را برداریم. در حرکت به سمت بالا اگر در خانهٔ (i,j) باشیم، می‌توانیم به خانهٔ بالایی آن یعنی $(i+۱,j)$ و به خانهٔ بالا سمت راست آن یعنی $(i+۱,j+۱)$ و به خانهٔ بالا سمت چپ آن یعنی $(i+۱,j-۱)$ در صورت وجود برویم. به ما کمک کنید از مسیری برویم که بیشترین پول ممکن را جمع کنیم.

مسئلهٔ دوازدهم: دستگاه «کار انجام ده» ..... $O(nT) = O(n^۲)$

فرض کنید یک دستگاه داریم و n کار $a_۱$ تا a_n که می‌توانند توسط این دستگاه انجام شوند. هر کار a_i به اندازه t_i $(۱ \le t_i \le n)$ واحد زمانی طول می‌کشد تا انجام شود و اگر تا زمان d_i انجام شود سود p_i به ما داده می‌شود و در غیر این صورت هیچ سودی به ما داده نمی‌شود. دستگاه ما در هر واحد زمانی می‌تواند یک واحد زمانی از یک کار را انجام دهد. همچنین اگر کار a_i به دستگاه داده شد، باید بی‌وقفه تا اینکه کار به اتمام برسد در دستگاه انجام شود. هدف برنامه‌ریزی انجام کارها، یعنی ترتیب دادن کارها به دستگاه می‌باشد بطوری که بیشترین سود را بدست آوریم. روشی برای پیدا کردن این ترتیب بدست آورید.

مسئلهٔ سیزدهم: تبدیل رشته‌ها ..... $O(n^۲)$

فرض کنید u و v دو رشته باشند. ما می‌خواهیم رشتهٔ u را به رشتهٔ v با عملهای زیر تبدیل کنیم:

حذف یک کاراکتر؛

اضافه کردن یک کاراکتر در یک مکان؛ و

عوض کردن یک کاراکتر.

یک الگوریتم بدهید که این کار را با کمترین عملیات انجام بدهد.

مسئلهٔ چهاردهم: ضرب الفبایی ..... $O(n^۲|\Sigma|)$

فرض کنید یک الفبای $\sum$ و حاصل ضرب هر دو عنصر در این الفبا داده شده است. الگوریتمی

بدست آورید که محاسبه کند که آیا برای رشته $x = x_1 \times x_2 \times \ldots \times x_n$ می‌توان طوری پرانتزگذاری کرد که حاصل این عبارت یک عنصر خاص در این الفبا شود. همچنین تعداد روش‌های پرانتزگذاری را که این مقدار حاصل شود بدست آورد.

مسئلهٔ پانزدهم: رودخانه و قایقها ..... $O(n^3)$

در رودخانهٔ کارون n نقطه وجود دارد که در آنها می‌توان قایق اجاره کرد. فرض کنید این نقاط در راستای رودخانه به ترتیب از ۱ تا n شماره‌گذاری شده‌اند. همچنین فرض کنید هزینه اجاره کردن قایق از نقطهٔ i و رفتن تا خانهٔ j ام a_{ij} باشد. روشی ارائه دهید که با کمترین هزینه از نقطهٔ ۱ با اجاره کردن تعدادی قایق به نقطهٔ n ام برسیم.

مسئلهٔ شانزدهم: رودخانه و آدم ..... $O(n^3)$

در رودخانهٔ کارون n نقطه وجود دارد که در آنها می‌توان قایق اجاره کرد. فرض کنید این نقاط در راستای رودخانه به ترتیب از ۱ تا n شماره‌گذاری شده‌اند. همچنین فرض کنید زمان رفتن از نقطهٔ i به نقطهٔ j ام توسط قایق، a_{ij} و زمان رفتن از نقطهٔ i به نقطهٔ j ام با پای پیاده، b_{ij} باشد. روشی ارائه دهید که در کمترین زمان ممکن از نقطهٔ ۱ به نقطهٔ n ام برسیم.

مسئلهٔ هفدهم: خرد کردن پول (کوله پشتی) ..... $O(nm)$

فرض کنید مقدار n تومان پول داریم و می‌خواهیم تمام این پول را با سکه‌های به اندازهٔ $c_1, \ldots c_m$ خرد کنیم. روشی برای خرد کردن پیدا کنید که کمترین تعداد سکهٔ لازم برای خرد کردن n تومان پول را بدست آورد.

مسئلهٔ هجدهم: رابطه ..... $O(n^2)$

فرض کنید n شیء در اختیار داریم که می‌توانیم به ترتیبی آنها را پشت سر هم قرار داده و بین آنها علائم $<$ و $=$ را قرار دهیم. تعداد راههای انجام این کار را بدست آورید. مثلاً برای $n = 2$، ۳ روش و برای $n = 3$، ۱۳ روش وجود دارد.

مسئله‌ی نوزدهم: مثلث بندی ..... $O(n^3)$

نقاط یک n ضلعی به ما داده شده است. می‌خواهیم این n ضلعی را طوری مثلث بندی کنیم که کمترین هزینه را داشته باشد. هزینهٔ یک مثلث‌بندی برابر مجموع طول پاره‌خط‌هایی که کشیده‌ایم می‌باشد.
جواب: گرفتن یک یال n ضلعی و ساخت مثلثی که این یال در آن قرار دارد با پیدا کردن نقطهٔ

مسئلهٔ بیستم: **مزرعهٔ مستطیلی** ..... $O(mn)$

یک منطقهٔ مستطیلی به شکل یک جدول $m \times n$ می‌باشد. در این منطقه یک زارع می‌خواهد یک مزرعهٔ مستطیل شکل بسازد. ولی بعضی از خانه‌های این مستطیل قابل کشت نمی‌باشد. فرض کنید این خانه‌ها مشخص شده باشند. هدف این زارع پیدا کردن بزرگترین مزرعهٔ قابل کشت در این منطقه می‌باشد، به او کمک کنید که این مزرعه را پیدا کند.

ایدهٔ حل: مقدار p_{ij} را برابر تعداد خانه‌های قابل کشت در سطر i ام و بعد از خانه j ام بگیرید. همچنین d_{ij} را برابر بزرگترین مستطیل که گوشه بالا سمت چپ آن خانه (i, j) باشد، در نظر بگیرید.

مسئلهٔ بیست و یکم: **تعداد درختها** ..... $O(nk)$

الگوریتمی برای محاسبهٔ تعداد درخت‌های دودویی ریشه‌دار با n رأس و ارتفاع k بدست آورید.

مسئلهٔ بیست و دوم: **کوله‌پشتی** ..... $O(nW)$

یک دزد برای دزدی به یک خانه می‌رود. در این خانه یک سری جنس قرار دارد. جنس i ام در این خانه w_i کیلوگرم وزن و ارزش c_i دارد. همچنین این دزد یک کوله‌پشتی با گنجایش W کیلوگرم دارد. به این دزد کمک کنید تا بیشترین سود ممکن را بکند، یعنی جنسهایی را بردارد که سود او را بیشینه کند.

مسئلهٔ بیست و سوم: **آقای نجّار** ..... $O(nT)$

یک نجار قصد دارد که برای مغازهٔ خود که تازه ساخته است، یک سری وسیله برای فروش درست کند. او برای این کار T زمان وقت دارد. از طرفی n نوع وسیله وجود دارد که او می‌تواند بسازد که ساخت وسیلهٔ i ام t_i واحد زمانی طول می‌کشد و قیمت آن p_i می‌باشد. او می‌خواهد وسایلی که در مغازه‌اش می‌گذارد مجموعاً بیشترین قیمت ممکن را داشته باشند (دقت کنید که از هر وسیله می‌تواند چند تا بسازد)، الگوریتمی بدست آورید که به این نجار در ساخت وسایل کمک کند.

سؤال: الگوریتم شما باید از حافظه $O(T)$ استفاده کند.

مسئلهٔ بیست و چهارم: **مزرعهٔ مربعی** ..... $O(n^2)$

یک منطقهٔ مربعی به شکل یک جدول $n \times n$ می‌باشد. در این منطقه یک زارع می‌خواهد یک مزرعهٔ مربعی شکل بسازد. ولی در بعضی از خانه‌های این مستطیل درخت وجود دارد. فرض کنید این خانه‌ها مشخص شده‌اند. هدف این زارع پیدا کردن بزرگ‌ترین مزرعهٔ قابل کشت در این منطقه می‌باشد؛ به او کمک کنید که این مزرعه را پیدا کند.
جواب: مقدار p_{ij} را برابر تعداد خانه‌های قابل کشت در سطر i ام و بعد از خانه j ام بگیرید و q_{ij} را تعداد خانه‌های در ستون j ام و بعد از خانهٔ i ام بگیرید، همچنین d_{ij} را برابر بزرگ‌ترین مربعی که گوشهٔ بالا سمت چپ آن خانهٔ (i, j) باشد، در نظر بگیرید.

مسئلهٔ بیست و پنجم: لیست پول ..... $O(mnk)$

فرض کنید n نوع پول داریم و می‌توانیم حداکثر تعداد m پول استفاده کنیم. بیشترین مقدار k را پیدا کنید که بتوان هر کدام از مقادیر بین ۱ تا k را با حداکثر m پول ساخت.

مسئلهٔ بیست و ششم: جعبه‌های شکلات ..... $O(nw^2)$

یک شرکت شکلات سازی n نوع جعبهٔ مختلف از یک شکلات دارد. جعبهٔ i ام دارای $c_i \leq w$ تا شکلات می‌باشد. ما می‌خواهیم بدانیم بیشینهٔ k چقدر است که با این n نوع جعبه نتوان k تا شکلات خرید. الگوریتمی برای بدست آوردن k ارائه نمایید.
ایدهٔ حل: باقیمانده‌های بزرگ‌ترین جعبه را باید ساخت.

مسئلهٔ بیست و هفتم: روش‌های پول‌سازی ..... $O(nk)$

تعداد راههایی را که می‌توان عدد n را با استفاده از سکه‌های $c_1, \ldots, c_k$ تومانی ساخت چندتا می‌باشد، اگر از هر سکه بتوان به تعداد دلخواه استفاده کرد.

سؤال: الگوریتم شما باید از حافظهٔ $O(n)$ استفاده کند.

مسئلهٔ بیست و هشتم: تور با هواپیما ..... $O(n^3)$

در یک کشور n شهر وجود دارد. بین یک سری از این شهرها خطوط هوایی دوطرفه وجود دارد. حال می‌خواهیم از چپ‌ترین شهر این کشور شروع کرده و از یک سری شهر بگذریم به طوری که همیشه به سمت راست برویم و به راست‌ترین شهر برسیم و از آن شهر دوباره حرکت کرده و با طی کردن یک سری شهر در جهت چپ به شهر اول بازگردیم به طوری که در این بین بیشترین شهر ممکن را طی کرده باشیم. الگوریتمی برای پیدا کردن این مسیر بدست آورید.
ایدهٔ حل: d_{ij} را برابر بیشترین تعداد شهر که می‌توان در یک مسیر از i به ۱ و از ۱ به j

داشت تعریف می‌کنیم (۱ سمت چپ‌ترین شهر است).

مسئلهٔ بیست و نهم: موسیقی ..... $O(n^۲m)$

فرض کنید n آهنگ داریم. همچنین m تا دیسک فشرده برای ذخیره‌سازی این آهنگ‌ها وجود دارد. هر دیسک گنجایش ذخیره کردن T دقیقه آهنگ را دارد و یک آهنگ نمی‌تواند روی دو تا دیسک مختلف قرار گیرد. همچنین از آنجایی که ما موسیقیدان خوبی هستیم، ترتیب قرار گرفتن این آهنگها روی دیسک‌ها مهم است؛ به این منظور ما ترتیبی از این آهنگها داریم که باید این ترتیب حفظ شود و همچنین بیشترین تعداد آهنگ ممکن را ذخیره کنیم. الگوریتمی ارائه دهید که بیشترین آهنگ ممکن را که می‌توان در این دیسکها ذخیره کرد بدست آورد. **ایدهٔ حل:** d_{ij} را برابر بیشترین تعداد آهنگ بین آهنگ های ۱ تا i ام که می‌توان در j تا دیسک جا داد تعریف کنید.

مسئلهٔ سی‌ام: بازی با دنباله ..... $O(n^۲)$

یک دنبالهٔ n تایی از اعداد در اختیار داریم. دو نفر به این ترتیب بازی می‌کنند که در هر لحظه نفری که نوبت با اوست، یکی از اعداد ابتدا یا انتهای دنباله را انتخاب می‌کند و آن عدد از دنباله حذف می‌شود. هدف افراد جمع کردن بیشترین مجموع ممکن از اعداد است. اگر نفر اول و دوم بهترین بازی خود را در هر لحظه انجام دهند، نتیجه بازی را در پایان بگویید؛ یعنی بگویید که هر نفر چه امتیازی را کسب می‌کند.

مسئلهٔ سی و یکم: مجموعه‌های برابر ..... $O(n^۳)$

الگوریتمی بدست آورید که تعداد راههای افراز مجموعهٔ $\{۱,\dots,n\}$ را به دو مجموعه بدست آورد، که این دو مجموعه، مجموع اعدادشان با هم برابر باشد.

مسئلهٔ سی و دوم: پیشوندی از زیررشته‌ها ..... $O((n+m)L)$

یک رشتهٔ S به طول L در اختیار داریم و n زیررشته که مجموعه طول همهٔ آنها برابر m می‌باشد. می‌خواهیم طول بزرگ‌ترین پیشوندی از رشتهٔ S را بدست آوریم که توسط این زیررشته‌ها قابل ساخت باشد. الگوریتمی برای پیدا کردن این مقدار بدست آورید.

مسئلهٔ سی و سوم: مثلث مقدار ..... $O(n^۲)$

یک مثلث متساوی الاضلاع با ارتفاع n را در نظر بگیرید. فرض کنید در هر مختصات صحیح آن یک مقدار پول قرار دارد. ما می‌توانیم از یکی از خانه‌های پایینی آن حرکت کرده و هر دفعه به یکی از دو مختصات صحیح بالایی مختصاتی که در آن قرار داریم برویم تا به رأس بالایی

مثلث برسیم. هدف جمع کردن بیشترین پول در این مسیر می‌باشد. الگوریتمی برای بدست آوردن بیشترین پول ممکن بدهید.

مسئلهٔ سی و چهارم: دنباله‌های دودویی ..... $O(nL)$

فرض کنید مجموعهٔ S، مجموعهٔ تمام رشته‌های دودویی (رشته‌های از (۰, ۱)) می‌باشد که تعداد ۱ های آنها کمتر یا مساوی L می‌باشد. الگوریتمی ارائه دهید که k امین عضو این مجموعه را بدهد، اگر اعضای این مجموعه به ترتیب الفبایی مرتب شده باشند.

مسئلهٔ سی و پنجم: مجتمع مسکونی ..... $O(n^2)$

یک مجتمع مسکونی آپارتمانی به شکل جدولی $n \times n$ داریم. در خانه‌ی مربعی جدول که در سطر i ام و ستون j ام است خانه‌ای به ارتفاع h_{ij} قرار دارد. به یک ساختمان دلباز می‌گوییم، اگر کسی با ایستادن روی آن (هر قدر کوتاه باشد)، با خطی افقی یا عمودی بیرون این مجتمع را ببیند. می‌خواهیم ارتفاع ساختمانها را طوری زیاد کنیم که تمام ساختمان‌ها دلباز شوند و میزان ساخت و ساز کمینه شود. d_{ij} را ارتفاع اضافه شده به ساختمانها می‌گیریم. الگوریتمی برای پیدا کردن d_{ij} ها بدست آورید.

مسئلهٔ سی و ششم: مجموعه‌های برابر ..... $O(n^3)$

اتاقی به شکل یک جدول $n \times n$ داریم که در برخی از خانه‌های آن مجسمه قرار دارد! بیرون اتاق $2n$ میله (خیلی بلند) تعبیه شده است که n تای آنها به صورت عمودی بالای n ستون جدول و n تای دیگر به صورت افقی سمت راست سطرهای جدول می‌باشند. می‌خواهیم بیشترین مساحت اتاق را با وارد کردن قسمتی از میله‌ها (در همان راستا) به داخل جدول بپوشانیم به طوری که هیچ دو میله‌ای هم دیگر را قطع نکنند و میله‌ها از مجسمه‌ها رد نشوند. الگوریتمی طراحی کنید که بیشترین مساحتی را از اتاق پیدا کند که می‌توان با میله‌ها پوشاند.

مسئلهٔ سی و هفتم: دورترین برگ ..... $O(n)$

یک درخت ریشه‌دار داریم. به ازای هر رأس v می‌خواهیم دورترین برگی را بیابیم که در زیردرخت آن رأس نباشد. دورترین برگ یعنی برگی که بیشترین فاصله را از رأس v دارد. فرض کنید برای هر رأس به غیر از ریشهٔ درخت پدر آن داده شده است. الگوریتمی بسازید که برای هر رأس دورترین برگ با خاصیت بالا را پیدا کند.

مسئلهٔ سی و هشتم: امید به بازگشت ..... $O(nk)$

روی محور x ها n نقطه داده شده است. می‌خواهیم با k بازهٔ بسته فاصله‌ی بین نقطهٔ اول تا آخر را بپوشانیم، به طوری که سر و ته بازه‌ها روی نقاط قرار بگیرد. طول بازه می‌تواند صفر باشد. الگوریتمی برای انجام این کار ارائه دهید که طول بزرگترین بازه کمینه شود.

مسئلهٔ سی و نهم: خرد کردن سکه (کوله پشتی) ..... $O(nW)$

الگوریتمی ارائه دهید که چک کند آیا می‌توان پولی به اندازهٔ W تومان را با سکه‌هایی به اندازهٔ $c_1, \ldots, c_n$ خرد کرد. الگوریتم شما باید حافظه‌ای از $O(W)$ استفاده کند.

سوال: آیا می‌توانید الگوریتم خود را طوری تغییر دهید که سکه‌های مورد استفاده برای ساخت W را هم بدست آوریم.

مسئلهٔ چهلم: شاه‌گذاری ..... $O(n\mathbf{2}^{2m})$

الگوریتمی ارائه دهید تا تعداد راه‌هایی که می‌توان تعدادی شاه را در یک صفحه شطرنجی $n \times m$ قرار داد بطوری که همدیگر را تحدید نکنند، بدست آورد.

سوال: آیا می‌توانید الگوریتم خود را طوری تغییر دهید که از زمان $O(nF(m)^2)$ شود، که $F(m)$، m امین عنصر در دنباله فیبوناچی می‌باشد.

مسئلهٔ چهل و یکم: اسب‌گذاری ..... $O(n\mathbf{2}^{3m})$

الگوریتمی ارائه دهید تا تعداد راه‌هایی که می‌توان تعدادی اسب را در یک صفحه شطرنجی $n \times m$ قرار داد بطوری که همدیگر را تحدید نکنند، بدست آورد.

مسئلهٔ چهل و دوم: خانه‌سازی ..... $O(\mathbf{2}^{3m} \log n)$

الگوریتمی برای پیدا کردن تعداد راه‌هایی که می‌توان یک جدول $\mathbf{2}^n \times m$ را با کاشی‌هایی به شکل یک تک خانه و یا ۳ خانه که یک کنج را ایجاد می‌کنند پوشاند، بطوری که هیچ دو کاشی‌ای با هم همپوشانی نداشته باشند، بدست آورید.

مسئلهٔ چهل و سوم: دیوار ..... $O(n^2)$ و $O(n \sum h_i)$

یک دیوار داریم که از n ستون تشکیل شده است. در هر ستون از این دیوار تعداد h_i آجر قرار دارد. در هر حرکت می‌توانیم یک آجر از بالای یک ستون برداشته و در بالای یکی از دو ستون مجاور آن (در صورت وجود) قرار دهیم. الگوریتمی ارائه دهید که در کمترین حرکت کاری کند که اختلاف کوتاهترین ستون دیوار و بلندترین ستون آن حداکثر ۱ واحد باشد.

مسئلهٔ چهل و چهارم: مسافرت با ماشین ..... $O(n^2)$

وحید می‌خواهد از شهر a_1 پس از طی شهرهای $a_2, \ldots, a_{n-1}$ به شهر a_n ام برود. در ابتدای سفر یعنی در شهر a_1 باک بنزین ماشین او پر است. او مقدار بنزینی که از شهر a_i به a_{i+1} مصرف می‌شود، می‌داند. ضمناً مبلغ هر لیتر بنزین در شهر a_i، c_i است. او می‌خواهد با بنزین زدن در بعضی شهرها به مقصد برسد. اما وقتی در یک شهر بیش از نصف باک بنزین ماشین او پر است و می‌تواند به شهر بعدی برسد، در این شهر بنزین نمی‌زند. ضمناً اگر در یک شهر بنزین بزند، حتماً باک بنزین خود را پر می‌کند. او می‌خواهد تصمیم بگیرد در چه شهرهایی باید بنزین بزند تا با حداقل هزینه به مقصد برسد. الگوریتمی طراحی کنید که مشکل وحید را حل کند.

مسئلهٔ چهل و پنجم: تعداد رشته‌ها ..... $O(nk)$

الگوریتمی ارائه دهید که تعداد رشته‌های به طول n از نویسه‌های a و b که اختلاف تعداد a و b ها در هر زیررشته متوالی این رشته حداکثر برابر k باشد را به دست آورد.

مسئلهٔ چهل و ششم: گلوله‌ها ..... $O(n^2 L)$

n گلوله داده شده است. وزن گلولهٔ i ام، w_i و رنگ هر گلوله آبی یا قرمز است. می‌خواهیم همه این گلوله‌ها را در دو کفهٔ یک ترازو قرار دهیم تا دو کفهٔ ترازی به تعادل برسد و نیز تعداد مهره‌های از هر رنگ یک کفه با مهره‌های از همان رنگ در کفه دیگر برابر باشد. الگوریتمی ارائه دهید تا تشخیص دهد این کار امکان‌پذیر است یا نه.

مسئلهٔ چهل و هفتم: الفبای تبدیل ..... $O(nm)$

الفبای $A = \{a_1, a_2, \ldots, a_k\}$ داده شده است. دو رشتهٔ S_1 و S_2 به طول‌های n و m از الفبای $A \cup \{*\}$ می‌باشند. نویسهٔ $*$ می‌تواند هر رشته از الفبای A باشد. می‌خواهیم کوتاه‌ترین رشتهٔ S از الفبای A را طوری پیدا کنیم تا S_1 و S_2 پس از گسترش $*$، زیر رشته‌هایی از S شوند. مثلاً اگر $A = \{a, c, g, t\}$، $S_1 = *a*cgta*aa*g*$ و $S_2 = *agtcg*ag*$ و $S = agtcgtaaag$ یک جواب است. الگوریتمی برای این مسئله ارائه دهید.

مسئلهٔ چهل و هشتم: میز گرد ..... $O(n^3)$ و $O(n^2)$

دور یک میز n نفر نشسته‌اند که در میان آن‌ها بعضی با بعضی دیگر آشنایند. افراد دور این میز را به ترتیبی که نشسته‌اند با شروع از یک فرد دلخواه از ۱ تا n شماره‌گذاری کرده‌ایم. لازم است ترتیبی اتخاذ شود که هر کس ۲ نفر مجاور خود را بشناسد. می‌خواهیم با بلند کردن حداقل تعداد نفرات از سر میز به این مهم دست یابیم. برنامه‌ای بنویسید که ما را در انتخاب کسانی را

که بلند می‌کنیم، یاری دهد. پس از بلند کردن تعداد لازم از افراد، دو نفر مجاور هستند اگر کسی بین آن دو ننشسته باشد.

ایدهٔ حل: d_i را برابر کمترین تعداد افراد که باید بلند شود که از نفر i به طور ساعت‌گرد تا نفر آخر شرط مسئله برقرار باشد.

مسئلهٔ چهل و نهم: گراف سیاه و سفید ..... $O(n^3)$

در گراف سادهٔ G به هر یال e عدد w_e نسبت داده شده، همچنین هر کدام از یال‌های آن با یکی از دو رنگ سیاه و سفید رنگ شده است. هدف پیدا کردن مسیری از یک راسی s به راس t می‌باشد به طوری که هزینهٔ آن بهینه باشد. هزینهٔ یک مسیر برابر است با مربع مجموع وزن‌های یال‌های سیاه آن مسیر به اضافه تعداد یال‌های سفید آن مسیر، یعنی:

$$l(P) = (\sum_{e \in P \ , \ e \ is \ black} w_e)^2 + |\{e \in P | e \ is \ white\}|$$

و شما $l(P)$ که می‌یابید که بین s و t است را کمینه کنید.

مسئلهٔ پنجاه و ام: طرح سوال یا اسرار جلسهٔ کمیته ..... $O(nh)$ و $O(hw)$

n سوال در اختیار داریم که به هر کدام از آنها یک نمره w_i نسبت داده‌ایم که $w_i \leq w$. هدف پیدا کردن تعدادی سوال است که بیشترین مجموع سوالات را بوجود بیاورند و البته این مجموع از h[۷] کمتر شود و البته طی این شرایط هدف ثانویه ما یعنی کمینه بودن تعداد سوالات برقرار شود. روشی برای پیدا کردن این مقادیر بدست آورید.

مسئلهٔ پنجاه و یکم: جواب معما ..... $O(n^3)$

معما از این قرار است که یک سری گوی جادویی با رنگ‌های مختلف در یک ردیف قرار داده بود. هر بار فرشته مجموعه‌ای از چند گوی همرنگ مجاور هم را انتخاب می‌کند و آن‌ها را همزمان منفجر می‌کند. در صورتی که با یک لمس کردن فرشته k گوی منفجر شده باشد، k^2 شکلات به فرشته داده می‌شود. هدف فرشته بدست آوردن بیشترین تعداد شکلات است. به او برای بدست آوردن بیشترین شکلات کمک کنید.

مسئلهٔ پنجاه و دوم: امید در زندگی ..... $O(n \log n)$

n عدد a_1 تا a_n به ما داده شده است. می‌خواهیم بعد از انجام عملیاتی به هر سوال به شکل زیر در زمان $O(1)$ پاسخ دهیم:

[۷] حبناک: حد بیشینهٔ نمرات المپیاد کامپیوتر

کوچکترین عدد بین اعداد a_i تا a_j چند می‌باشد؟

روشی ارائه دهید که با انجام عملیاتی از مرتبه $O(n \log n)$ کاری کنید که به هر سوال در زمان $O(۱)$ پاسخ دهیم.

مسئلهٔ پنجاه و سوّم: شیطان بزرگ ..... $O(nh)$

n برج در یک دریف پشت سر هم قرار دارند. هر کدام از این برج‌ها چند طبقه دارد. در داخل هر طبقه یک آدم کار می‌کند که کشتنش مقداری مفید می‌باشد (اگه بکشیمش به ما به همان اندازه پول میدهند !!!) البته متأسفانه یک سری آدم خودی هم توی این برج هستند بطوری که با کشتن آنها مقداری پول از ما می‌گیرند. تعدادی هواپیما در اختیار داریم (به میزان کافی !!!). قرار است با استفاده از این هواپیماها این برج‌ها را خراب کنیم. هر وقت یک هواپیما به یک طبقه از یک برج می‌خورد همهٔ طبقات بالای آن طبقه به اضافهٔ خود آن طبقه نابود میشوند. به برخی دلایل امنیتی موظف هستید تخریب رو طوری انجام بدین که در نهایت برجی نباشد که هم سمت راستش و هم سمت چپش برج بلندتر از آن وجود داشته باشد (هر برجی یا باید از همهٔ سمت راستیهایش بلندترمساوی باشه یا از همهٔ سمت چپیهایش یا از هر دو). چیزی که بدیهی است این است که شما می‌خواهید بیشترین سود را بدست آورید. الگوریتمی برای محاسبه بیشترین سود بدهید. دقت کنید که ارتفاع این برج‌ها کمتر مساوی h می‌باشد.

مسئلهٔ پنجاه و چهارم: استوانه ..... $O(n^۳)$ و $O(n^۲)$

یک استوانه مشبک داریم که از ارتفاع آن n سطر و محیط آن نیز دارای n ستون می‌باشد، یعنی در مجموع $n^۲$ خانه در روی محیط آن قرار داند. همچنین در هر خانه از این استوانه مقداری پول قرار دارد. فرض کنید ستون‌های این استوانه را از ۱ تا n و سطرهای آن را نیز از ۱ تا n به ترتیب شماره‌گذاری کرده‌ایم. حال می‌خواهیم از خانه $(۱, ۱)$ یک استوانه حرکت کرده و هر لحظه به پایین بیاییم و یا به سمت راست برویم و بیشترین پول را از جدول جمع کرده و در آخر در خانه (n, n) باشیم. الگوریتمی برای پیدا کردن بیشترین پول که می‌توان جمع کرد، بدست آورید.

مسئلهٔ پنجاه و پنجم: جدول ..... $O(nk)$

جدولی $n \times n$ از اعداد مثبت داریم که می‌خواهیم تعدادی از خانه‌هایش را انتخاب کنیم. با این شرط که اگر یک خانه انتخاب شود، تمام خانه‌های ردیف بالاییش هم انتخاب شده باشد. همچنین می‌خواهیم مجموع اعداد این خانه‌ها برابر مقدار معلوم k باشد. الگوریتمی برای انجام این کار ارائه دهید.

مسئلهٔ پنجاه و ششم: کارخانه چوب‌بری عجیب ..... $O(n^2)$

در یک کارخانهٔ چوب‌بری عجیب برای اینکه یک تکه چوب را در یک مرحله به k تکه برش بزنند c_k واحد پول گرفته می‌شود. می‌خواهیم یک تکه چوب را باکمترین خرج دقیقاً n قسمت کنیم.

مسئلهٔ پنجاه و هفتم: شیر فروشی ..... $O(nw)$

علی آقا شیر فروش می‌خواهد به مشتری‌های خود شیر بفروشد. هر مشتری مقدار w لیتر شیر می‌خواهد. او می‌خواهد کمترین تعداد پیمانه از بین n پیمانه با گنجایش‌های $a_1, \ldots, a_n$ را انتخاب کند به طوری که بتواند w لیتر شیر را با آنها از ظرف اصلی خود بردارد و به ظرف مشتری بریزد. به علی آقا کمک کنید.

مسئلهٔ پنجاه و هشتم: تصحیح اوراق ..... $O(nT)$

n برگه برای تصحیح در اختیار داریم. دو نفر مشغول تصحیح برگه‌های می‌باشند. این n برگه به ترتیب روی هم چیده شده‌اند و برگه i ام t_i ثانیه طول می‌کشد تا تصحیح شود ($t_i \leq T$). نفر اول محمد و نفر دوم حسین است. محمد در هر لحظه اگر برگه‌ای را تصحیح نکند، می‌تواند برگه بالایی را برای تصحیح بردارد و یا اینکه هر مقداری که دوست دارد صبر کند. ولی حسین در هر لحظه که برگه‌ای تصحیح نکند، اگر محمد برگه رویی را برندارد او آن را برای تصحیح بر می دارد. محمد می‌خواهد بیشترین برگه را تصحیح کند. به او برای این کار کمک کنید. یعنی روشی به او یاد بدهید که بیشترین تعداد برگه را تصحیح کند.

مسئلهٔ پنجاه و نهم: بسته‌بندی کمینه ..... $O(nP^2)$ و $O(nP)$

یک کارخانه دارای یکی سیستم بسته‌بندی کالا به شرح زیر است: n کالا که هر یک وزنی کعادل w_i دارد به ترتیبی از پیش تعیین شده وارد مرکز بسته‌بندی می‌شود. در مرکز بسته‌بندی دو بستهٔ باز قرار دارد که هر یک حداکثر P واحد وزن را در خود جای می‌دهد. سیستم ما در برابر کالایی که وارد می‌شود. این عکس‌العمل‌ها را می‌تواند نشان دهد:

- آن را در یکی از بسته‌های فعلی قرار دهد.
- یکی از بسته‌ها را ببندد و کنار بگذارد و یک بسته خالی مشابه را به جایش قرار دهد و کالا را در آن قرار دهد.

در انتهای کار بسته‌های غیرخالی فعلی را هم بسته و کنار می‌گذاریم. هدف ما کم کردن تعداد بسته‌هایی است که استفاده کرده‌ایم. الگوریتمی طراحی کنید که با کمترین تعداد بسته این کار را

انجام دهد.

مسئلهٔ شصت‌ام: کارمند تنبل ..... $O(nT)$

n کار با زمان‌های پردازش $t_1, \ldots, t_n$ به یک کارمند تنبل محول شده‌اند. موعد شروع کار i ام a_i و موعد اتمام آن d_i است. کار i ام فقط در بازهٔ $[a_i, d_i]$ قابل انجام است و به محض شروع هر کار باید تا اتمام کار آن را بدون وقفه انجام داد. اما چون کارمند تنبل است، سعی می‌کند همیشه کمترین زمان ممکن را کار کند. این کارمند رئیسی دارد که مرتباً در هر لحظه او را تحت نظر دارد: اگر ببیند کارمند بیکار است در حالی که کاری «قابل اجرا» وجود دارد، او را اخراج می‌کند. در لحظه t به کاری «قابل اجرا» می‌گوییم که از موعد شروع آن گذشته باشد و اگر در همین لحظه شروع به انجام این کار کنیم، حداکثر تا موعد اتمام آن تمام شود. این کارمند سعی دارد به نحوی زیرمجموعه‌ای از کارها را انتخاب کند تا اولاً اخراج نشود و ثانیاً مجموع زمان‌هایی که کار انجام می‌دهد کمینه شود. دقت کنید که $d_i < T$ می‌باشد. الگوریتمی برای کمک به او پیدا کنید.

Memoization

یک تغییر در روش پویا که معمولاً کارایی را بالا می‌برد، استفاده از فن «برنامه‌سازی پویا» ولی از بالا به پایین می‌باشد. در واقع ما در روش پویا یک جدول را مبنای کار قرار داده و آن را پر می‌کنیم، ولی کنترل روی اینکه چه خانه‌هایی باید پر شود، یعنی چه خانه‌هایی لازم است پر شوند، خیلی مورد دقت واقع نمی‌شود. در روش بازگشتی پویا که Memoize نام دارد، در ابتدا خانه‌های جدول یک مقدار ابتدایی می‌گیرند که نشان می‌دهد، آن خانه‌ها هنوز پر نشده‌اند. سپس مسئله به زیرمسئله‌ها تبدیل شده و زیر مسئله‌ها حل می‌شوند؛ همان روشی که در حل مسائل بازگشتی داشتیم. ولی در هر حل یک زیرمسئله ابتدا بررسی می‌شود که آیا این زیرمسئله تا به حال حل شده است یا خیر؛ اگر حل شده بود، از مقداری که در جدول هست استفاده می‌شود و اگر حل نشده بود، در این لحظه حل می‌شود، به این ترتیب که خود این مسئله باز به یک سری زیر مسئله تقسیم شده و این زیر مسئله‌ها به همان روش حل می‌شوند. به این ترتیب دیگر حالات یا زیرمسئله‌هایی که در روش پویا حل می‌شود، ولی نیاز به حل کردن آنها نبوده است، در روش اخیر حل نمی‌شوند و همچنین متکی به روش پویا است. یعنی هر زیرمسئله حداکثر یک بار حل می‌شود.

مسئله‌ی ؟م: ضرب ماتریس‌ها ..... $O(n^2)$