طراحی الگوریتم

راهنمایی پروژه ۳

سوال ۱ (بازی کامپیوتری):

مساله در حال ورودی دادن یک گراف با یال های آن است. اگر حالت c=1 را فقط در نظر بگیریم پس یال هایی که باید حفظ کنیم حتما باید کمترین مقدار را داشته باشند تا با فروش بقیه یال های حداکثر کنسول را بتوانیم خریداری کنیم. (در حالتی که c!=1 باشد خب شاید باید درخت هایی که دقیقا مقدار کمینه را ندارند ولی تعداد کنسول هایی که می خریم فرقی نمی کند هم در نظر بگیریم که مساله سخت تری خواهد بود)

پس باید دنبال درخت پوشای کمینه (mst) باشیم. جواب وقتی ۱ است که راهی وجود نداشته باشد که این یال فروخته نشود یعنی در هیچ mst ای نباشد. پس سوال پیدا کردن یال هایی است که در هیچ یک از mst های گراف نباشد. (می دانیم مجموع یال های درخت پوشای کمینه از بقیه درخت ها کمتر است ولی ممکن است چند درخت این مقدار کمینه را داشته باشند که همه آن ها mst محسوب می شوند)

همچنین گراف ساده نیست و ممکن است بین یک جفت راس دو یال با وزن مختلف باشد (که البته هر دوی آن ها دو طرفه هستند!) که اگر یکی بیشتر از دیگری باشد در هیچ mst ای نمی آید و اگر برابر باشند خروجی آن ها یکسان خواهد بود.

برای حل سوال کافیست مقداری الگوریتم کروسکال را تغییر دهید. صرفا به این الگوریتم به عنوان جعبه سیاهی که مساله درخت پوشای کمینه را حل میکند نگاه نکنید و سعی کنید آن را روی مثال های کوچک اجرا کنید و نحوه اجرا آن را متوجه شوید.

اردر زمانی مطلوب مساله همان $e.lge$ که مربوط به کروسکال است می باشد ولی با $O(n^3)$ نیز می توانید به غیر از دو تا تست بقیه تست ها را بگیرید.

سوال ۲ (نوروز):

باز هم یک گراف بدون جهت وزن دار در اختیار داریم. اگر به مثال اول دقت کنید نکته ای که هر فرد حتما باید بعد از هر مهمانی به خانه برگردد را می بینید. و اینکه ترتیب ورودی عید دیدنی ها مهم است و باید دقیقا به همین ترتیب اجرا شود (یا به عبارتی دیگر در هنگام عید دیدنی x با y باید همه عیددیدنی های این دو نفر قبل از این زمان انجام شده باشد) و دو نفر نیز می توانند همزمان در خانه یک نفر عید دیدنی کنند! (همانطور که در مثال اول می بیند لازم نیست زمان شروع و پایان عید دیدنی شان نیز یکسان باشد)

طبعا فاصله هر جفت راسی را احتیاج داریم که ابتدا باید محاسبه کنیم. (چون n حداکثر ۵۰۰ است می توانیم از $O(n^3)$ استفاده کنیم.)

سپس باید روی عید دیدنی ها حرکت کنیم و دانه دانه آن ها را به صورت حریصانه در زودترین حالت ممکن انجام دهیم.

اگر از زبان c++ استفاده می کنید به خاطر k,w که حداکثر ۱۰ به توان ۹ هستند، ممکن است فاصله راس ها یا زمان رسیدن ها بزرگ شود و از int خارج شود که برای رفعش از تایپ long long استفاده کنید.

سوال ۳ (استاندارد شریفی):

در این سوال باید سعی کنید دنباله ای را با توجه به محدودیت هایی که دارد تولید کنید. از راهنمایی سوال استفاده می کنیم و صورت عبارات را بازنویسی می کنیم. (دنباله ها را با شروع از ۰ در نظر بگیرید و S_i را باز تعریف می کنیم یعنی $S_i = \sum_{j=0}^{i-1} a_j$ که مثلا S_n برابر مجموع کل آرایه می شود)

پس شرط $a - b > c$ به $S_{b+1} - S_a > c$ و شرط $a - b < c$ به $S_{b+1} - S_a < c$ تبدیل می شود. می دانیم برای آرایه a جواب وجود دارد اگر و تنها اگر برای آرایه S جواب وجود داشته باشد.

حال باید ربط این مساله را با گراف بفهمیم! در بحث کوتاهترین مسیر باید به یک نامساوی فکر کنیم، وقتی یک یال $e_{u,v}$ بین دو راس u و v وجود دارد (فرض کنید جهت دار از u به v) درباره فاصله آن

ها از یک مبدا مشخص (d_v) می توان نامساوی $d_v - d_u \leq e_{u,v}$ را نوشت (چون مسیری به v که ابتدا به u بیاییم سپس یال $e_{u,v}$ را طی کنیم یکی از مسیر های به v است پس فاصله یا اندازه کوتاهترین مسیر به این راس حداکثر $d_u + e_{u,v}$ خواهد بود)

باید با بررسی این گراف بفهمید که جواب برای دنباله وجود دارد یا نه و در صورت امکان آن را بسازید.

سوال ۴ (بشکن):

در این سوال باید وزن یال های پاک شده را جوری تعیین کنید که فاصله راس ۱ تا n مقدار d شود. هر یال مقدار حداقل ۱ و حداکثر d را می تواند داشته باشد. (البته اگر چون محدودیت w ده به توان نه است از این مقدار عدد بزرگ تری نمی توان اختصاص داد!)

فرض کنید که همه یال ها نامشخص مقدار d داشته باشند اگر فاصله را در این حالت حساب کنیم و کمتر از d شود آنگاه قطعا مساله جواب ندارد و به طرز مشابه اگر همه یال های نامشخص را ۱ بگذاریم و فاصله از d بیشتر شود نیز جوابی برای مساله نداریم. ثابت می کنیم به غیر از این دو حالت همواره مسئله جواب دارد.

یال های نامشخص را (k یال) به ترتیب دلخواه در نظر میگیریم. ابتدا همه ۱ هستند و ابتدا وزن اولین یال را دانه دانه افزایش می دهیم تا به d برسد سپس به سراغ یال دوم می رویم و از ۱ تا d تغییرش می دهیم و همین کار را برای یال سوم تا آخر می کنیم.

ابتدا که همه ۱ هستند فاصله کوچکتر مساوی d و آخر که همه d هستند فاصله بزرگتر مساوی d است. وقتی وزن یک یال یکی اضافه می شود بدیهی است که فاصله بین راس ۱ تا n حداکثر یکی اضافه می شود، پس زمانی وجود دارد که این فاصله دقیقا مقدار d را داشته باشد.

البته باید برای kd مقدار یال ها فاصله را حساب کنیم که مرتبه $O(kdn.lgm)$ را ایجاد می کند که برای محدودیت های مسئله زمان زیادی است و باید آن را با ایده هایی کاهش دهید.

همچنین توجه کنید ممکن است تا ۱۰ به توان ۹ تا راس داشته باشیم که تایم یا مموری به این اندازه نمی توانیم اختصاص دهیم ولی چون حداکثر ۱۰ به توان ۵ یال داریم اکثر آنها درجه صفر دارند و بی فایده هستند و در نتیجه نباید در پردازش حضور داشته باشند. می توانید راس ها را اندیس گذاری

جدیدی با سقف کمتر (حداکثر ۲ برابر تعداد یال ها) بکنید. (به این کار اصطلاحا compress گفته می شود)