به نام خدا

پروژه درس ساختمان داده

دکتر سمانه حسینی

شرح کلی پروژه (دید از بالا)

اول اینکه سلام :) امیداورم که تا اینجا تکالیف عملی رو به خوبی انجام داده باشید چون توی پروژه همونطور که از اول ترم تا حالا چندین مرتبه گفته شد به کدهاتون نیاز دارید.
خوب بریم سر پروژه. توی پروژه این ترم قراره که شما یه بازی استراتژیک کوچیک رو شبیه سازی کنید که نه بر اساس قوانین دنیای واقعی بلکه بر اساس قوانین ساختمان داده ایی شبیه سازی میشه !!!
بازی از این قراره که یک سرزمین فرضی داریم که یه سری قلعه روی نقشه سرزمین قرار دارند و قلعه ها به یک تعداد مشخصی سرباز دارند و تمام قلعه ها با هم دیگه دشمن هستن و به هم دیگه لشکرکشی میکنن و هدفشون تسخیر قلعه های دیگس.
اما بازی هم به این سادگی ها نیست !!! و باید براساس یک سری قوانین خاص که سعی شده تا مقدار ممکن از واقعیت دور نباشه انجام بشه.

شکل ۱:

ورودی

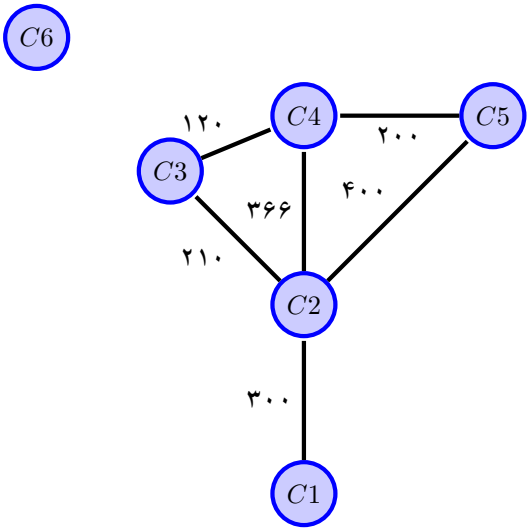
ورودی که به شما داده میشه به این صورته که یک ماتریس مجاورت گراف به شما داده میشه که هر نود روی گراف در واقع قلعه های سرزمین رو نشون میده و یال ها هم مسیرهای روی نقشه که سربازان فقط میتونن روی این مسیر ها حرکت کنند. به همراه طول مسیر ها. البته ورودی قسمت های دیگه ایی هم داره که برای پیچیده نشدن در ادامه گفته میشه. :)
خوب اولین سوالی که پیش میاد اینه که سربازان هر قلعه چه استراتژی برای حمله و دفاع دارند؟ پیشنهاد میکنم برای متوجه شدن دقیق پروژه از این قسمت به بعد رو یکبار به صورت روزنامه ایی بخونید و بار دوم به طور دقیق مطالعه کنید.

قوانین

استراتژی های حمله

۱. ظرفیت خروجی برای تمام قلعه های سرزمین یکسانه و در هر واحد زمانی به اندازه ظرفیت قلعه از اون سرباز خارج میشه(البته به شرط اینکه اون تعداد سرباز موجود باشه !!)

۲. سوال بعدی که پیش میاد اینه که سربازان به کدوم قلعه حمله کنند؟!
نحوه حمله به این صورته که ابتدا تمام قلعه هایی که به قلعه حمله کننده یک یال مستقیم وجود داره رو پیدا میکنیم و بر اساس جمعیت حال حاضر قلعه ها وزنی به هر مسیر میدیم و سربازان خروجی رو بر این اساس تقسیم میکنیم(تقسیم تعداد سربازان قلعه مورد نظر بر تعداد کل سربازان قلعه ها) .
به عنوان مثال فرض کنید قلعه ما به ۳ قلعه دیگر یال مستقیم داره که جمعیت سربازان هر کدوم به ترتیب ۲۵۰ـ۳۰۰ـ۵۰ باشه و ظرفیت خروجی از قلعه ها ۲۴ سرباز باشه بنابراین به ترتیب باید ۱۰ـ۱۲ـ۲ سرباز به هر کدوم از قلعه ها فرستاده بشه.
اگر تعداد سربازان مقدار اعشار داشت ما جز صحیح آن را در نظر میگیریم و سربازانی که اضافه می آیند را به قلعه ایی که کم ترین سرباز را دارد اختصاص میدهیم.

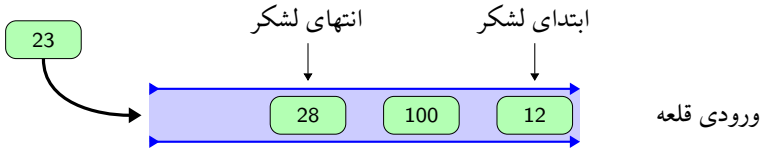


۳. سربازان سرعت مشخصی دارند که به شما داده میشود و همینطور جاده ها طول مشخصی دارند بنابراین سربازان الزاما در واحد زمانی بعدی به مقصد نمیرسند.
به عنوان مثال فرض کنید طول مسیر بین دو قلعه ۱۰ کیلومتر باشد و سرعت حرکت سربازان در هر واحد زمانی ۲ کیلومتر باشد بنابراین سربازان در ۵ واحد زمانی بعدی به پشت دروازه های قلعه مورد نظر میرسند.

ورود به قلعه

۱. ظرفیت ورودی قلعه ها به خاطر دفاع سربازان داخل شهر خیلی کم تر از ظرفیت خروجیه اونه که این ظرفیت ورودی بر اساس تعداد سربازان داخل قلعه ها مشخص میشه و هر چی سربازان داخل قلعه بیشتر باشند ظرفیت ورودی اون کمتر میشه.
این ظرفیت ورودی اینطوری مشخص میشه که ازسقف تقسیم تعداد سربازان پشت دروازه قلعه بر تعداد سربازان داخل قلعه به دست میاد.

۲. برای شبیه سازی سربازان دشمن پشت دروازه شهر باید حتما از صف استفاده کنید که در تمرین ها به صورت تمپلیت نوشتید(سربازانی که به پشت دروازه های شهر میرسند وارد صف میشوند و سربازانی که نوبتشون میرسه برای حمله به شهر از سر دیگر صف خارج میشوند.
یک نکته که باید دقت کنید اینه که ما داریم میگیم سربازان پشت دروازه های شهر باید وارد صف بشند اما توجه داشته باشید در واقع ترتیب اون ها که به چه ترتیبی وارد صف بشند از موقع خروجشون از قلعه مشخص شده.



داخل قلعه

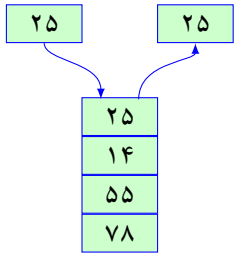
۱. هر سرباز یک میزان قدرتی داره که به شما داده میشه و قدرت سربازان سعی میشه از یک توزیع نرمال پیروی کنه.

۲. سربازان دشمنی که در هر واحد زمانی وارد قلعه میشوند به طریق خاصی وارد جنگ با سربازان داخل قلعه میشوند!! که در ادامه آورده شده.

سربازان داخل قلعه به فرم یک درخت AVL شکل گرفته اند تا فرمانده بتواند سربازانی که قرار است با سربازان دشمنی که وارد قلعه شده اند را زودتر انتخاب کند!! و این انتخاب به این شکل است که فرمانده سربازی که کم ترین میزان اختلاف قدرت از لحاظ قدرمطلق با سرباز نفوذی را دارد انتخاب میکند و آن هارا وارد میدان جنگ میکند.

۳. برای اینکه کمی شانس هم در نتیجه به جنگ بین دو سرباز دخیل بشه.سربازان برنده به این صورت مشخص میشوند که از ساختمان داده fibonacci heap استفاده میکنیم و ابتدا تمام سربازانی که داخل قلعه میباشند را وارد ساختمان داده fibonacci heap میکنیم (بر اساس قدرت سربازان) سپس تمام سربازانی که قرار نیست وارد جنگ بشوند را از ساختمان داده حذف میکنیم و سربازی که در عمق کم تری از سرباز دیگر است در جنگ بین سرباز متناظر خود میبازد و اگر دو سرباز در عمق یکسانی بودند سربازی که قدرت بیشتری دارد برنده میشود. دقت شود که امکان دارد در یک واحد زمانی چند سرباز دشمن وارد قلعه شوند(به ترتیبی که داخل صف اند وارد میشوند) بنابراین در یک واحد زمانی امکان وجود چند جنگ بین دو سرباز همزمان وجود دارد.
انتخاب هایی که بر اساس درخت AVL بود بر اساس ترتیب صف ورودی قلعه انجام میشود.
اما یک fibonacci heap برای تصمیم گیری برنده ها استفاده میشود یعنی در انتها تنها سربازانی که قرار است با یکدیگر بجنگند داخل fibonacci heap وجود دارند.

۴. هر قلعه یک پشته متناظر خود دارد که سربازانی که کشته میشوند وارد آن میشوند و در هر واحد زمانی اگر سربازی داخل پشته وجود داشته باشد به تعداد مشخصی سرباز از آن خارج میشوند. تعداد سربازان خارج شده میتواند اعشاری باشد به عنوان مثال اگر این مقدار 0.5 باشد یعنی در هر ۲ واحد زمانی یک سرباز خارج میشود. دقت شود قدرت سربازان تغیری نمیکند.



۵. اگر تعداد سربازان داخل قلعه که از قلعه دفاع میکنند صفر شود آن قلعه تسخیر میشود و جزو یکی از قلعه های دشمن میشود و دیگر سربازی از پشته مربوط به آن قلعه وارد قلعه نمیشوند اما سربازانی که زنده میباشند همچنان به عمر خود ادامه میدهند تا زمانی که کشته شوند.

۶. اگر سرباز دشمن داخل قلعه بر سرباز داخل قلعه پیروز شود در واحد زمانی بعدی نیز همچنان میتواند با سرباز دیگری بجنگد که به همان ترتیبی که در بالا ذکر شد انتخاب میشود.
(دقت شود در هر واحد زمانی fibonacci heap از ابتدا ساخته میشود و بر اساس آن تصمیم گیری صورت میگیرد)

۷. دقت شود سربازانی که برای جنگ اعزام میشوند از همان درخت AVL انتخاب میشوند و سربازانی که از پشته وارد قلعه میشوند نیز وارد همان درخت AVL میشوند.

۸. سربازانی که برای جنگ به خارج قلعه فرستاده میشوند این گونه انتخاب میشوند که تماما از پایین ترین گره های درخت AVL انتخاب میشوند.
به عنوان مثال ۱۰ سرباز برای فرستاده شدن به خارج انتخاب میشوند و درخت AVL ما ۸ گره بدون فرزند داشته باشد(برگ) ابتدا آن ۸ گره انتخاب میشوند و از درخت حذف میشوند سپس آن ۲ سرباز دیگر از برگ های تازه به وجود آمده انتخاب میشوند(نحوه انتخاب مهم نیست تنها موردی که اهمیت دارد انتخاب برگ ها میباشد.)

موارد دیگر

- حتما این سوال براتون پیش میاد که اگر دو لشکر از سربازان قلعه های مختلف در یک مسیر بودند چه اتفاقی میفته برای راحتی فرض میشه که دو لشکر بدون درگیری از کنار هم رد میشند و درگیری ها تنها در داخل قلعه هاست.

- اگر قلعه ایی به تسخیر قلعه دیگر درآمد همچنان سربازان کشته شده پس از خروج از استک وارد قلعه اصلی میشوند نه تسخیری.

- حتما برای پیمایش سرزمین از یکی از الگوریتم های dfs یا bfs استفاده کنید.

- لطفا اگر ابهامی بود در مورد پروژه حتما حتما در **Quera** و به صورت عمومی سوالتون رو بپرسید و تیک ایمیل رو هم بزنید که هر مشکل یکدفعه پاسخ داده بشه و برای کس دیگه اون مشکل پیش نیاد.

نحوه تحویل

نحوه تحویل به این صورت است که به برنامه شما ورودی های مورد نیاز داده میشه و برنامه شما باید قابلیت این را داشته باشد که نشان دهد در هر واحد زمانی چه تعداد سرباز در کدام قسمت نقشه قرار دارند.
همینطور این قابلیت موجود باشد که مقادیر داخل ساختمان های داده ها در هر واحد زمانی برنامه قابل مشاهده باشد به عنوان مثال بتوان ساختمان داده درخت AVL که برای قلعه دوم میباشد را در زمان ۲ مشاهده کرد(قبل از انتخاب سربازان) باید شکل درخت به طور کامل چاپ شود(یعنی فرزند های هر گره و والد آن با مشاهده در خروجی قابل شناسایی باشد.) یا به عنوان مثال برای fibonacci heap شکل درخت کامل قابلیت چاپ شدن داشته باشد. همینطور استک ها و صف ها و ...
دقت شود پیاده سازی این موارد برای اطمینان حاصل کردن از صحت پروژه شما الزامی میباشد.
حتما توی کد زنی به خوانا بودن کد و نام گذاری مناسب توجه کنید.اگر قسمتی از کد موقع تصحیح خوانا نبود احتمال کسر نمره وجود دارد.

موارد امتیازی

۱. کدنویسی صحیح و بهینه !! یعنی اینکه سعی کنید از کلاس ها و قابلیت های زبان برنامه نویسی به خوبی استفاده کنید(اگر از کلاس بندی استفاده نکنید پیاده سازی پروژه خیلی سخت میشه اما با استفاده از کلاس بندی مناسب پروژه در واقع به نوشتن چند الگوریتم و ساختمان داده محدود میشه که اکثرشونم پیاده سازی کردید.).

۲. پیاده سازی برنامه ایی که کم ترین تعداد سربازی که یک قلعه دلخواه نیاز دارد تا بتواند تمام قلعه ها را تسخیر کند.(گراف ورودی همبند است.)
به عنوان مثال برنامه شماره قلعه ۲ را بگیرد و در خروجی حداقل تعداد سرباز مورد نیاز برای اینکه بتواند تمام قلعه ها را تسخیر کند را چاپ کند و همینطور قابلیت دیباگ را داشته باشد(یعنی بتوان در هر مرحله بر نحوه اجرای بازی نظارت داشت.)