



به نام خداوند جان و خرد

نام و نام خانوادگی: محمد حسین کریمی

شماره دانشجویی: 9731833

پروژه آتش نشانی

بهار 1400

الگوریتم کلی

الگوریتم کلی پیدا کردن تمام مسیر ها از مبدا به مقصد در یک گراف است. در ابتدا ما نیاز به شبیه سازی گراف داریم. شروط لازم برای پیدا کردن مسیر ها این است که اولاً بین راس ها یال وجود داشته باشد دوماً از یک راس بیشتر از یکبار عبور نکنیم.

توضیح کد

از `map` برای نشون دادن لیست مجاورت استفاده کردیم و از یک لیست استفاده کردیم برای نگه داشتن راس ها و همچنین بصورت دو طرف راس ها رو اضافه می کنیم و لیست مجاورت را تشکیل می دهیم

متد `add edge`

ابتدا با `vlist` چک می کنیم که آیا راسی که کاربر وارد کرده قبلاً تو لیست بوده یا نه اگر نبود اضافه می کنیم. سپس میایم چک می کنیم آیا این راسی که وارد شده اولین بار وارد شده یا خیر اگر بار اول بود یک لیست مجاورت خالی برای آن راس درست می کنیم بعد به لیست مجاورت مون اون راس هایی که مرتبط هست را اضافه می کنیم همین کار را بر عکس انجام می دهیم چون گراف را بی جهت در نظر گرفتیم.

متد `print paths`

یک متدی است که در واقع کارهای اولیه مثل تعریف `visited, path` که مقدار نهایی ما در آن باشد. برای همه راس ها مقدار `visited` را `false` می کند.

متد `print pathsUtil`

ابتدا مقدار `visited` مبدا را `true` می کنیم. سپس `path` که جوابمون هست `path_index` مقدارش را برابر `s` می زاریم که این `source` ما هست یعنی همیشه از یک شروع می کنیم اما می تونیم این رو تغییر بدیم سپس `path_index` (ایندکس مسیرمون هست یعنی نشان هنده شماره ایستگاه مان هست) را صفر می زاریم یعنی ما از جواب به مبدا رسیدیم بعد یکی یکی مقدار را زیاد می کنیم حالا اگر به مبدا رسیده بودیم میایم مسیر را چاپ می کنیم و می ریم خط بعد در واقع ارایه `path` جوابهایی که درست است را در خودش نگه می دارد اگر نرسیده بودیم بر می گردیم به لیست مجاورت مون که ببینیم با چه راس های دیگه ای در ارتباط است و سراغ راس هایی می ریم که اولاً ارتباط داشته باشد ثانیاً مقدار `visited` آن `false` باشد. در پایان هم

path_index را یک واحد کم می کنیم چون یک مقدار اضافه بهش کردیم و s visited را مجدد false می کنیم تا دوباره بتونیم ازش استفاده کنیم.

در main هم مسیر هارا از کاربر می گیرد تا زمانی که ورودی 0 و 0 نبود. این برنامه تا بی انتها ادامه می دهد تا آتش سوزی های جدیدی گزارش می شود. در انتها print path صدا زده میشه تا خروجی را برای ما چاپ کند

پیچیدگی زمانی

$$O(n^n)$$

مونت کارلو:

$n+1$ (رئوسی که با راس مبدا یال مشترک دارند) * (رئوسی که علاوه بر داشتن یال مشترک با راس مورد نظر ممکن است به مسیر مورد نظر ما ختم شوند+...) (

اجرای برنامه

```

6
1 2
1 3
3 4
3 5
4 6
5 6
2 3
2 4
0 0
CASE 1: Print all paths to 6
1 2 3 4 6
1 2 3 5 6
1 2 4 3 5 6
1 2 4 6
1 3 4 6
1 3 5 6
1 3 2 4 6

4
2 3
3 4
5 1
1 6
7 8
8 9
2 5
5 7
3 1
1 8
4 6
6 9
0 0
CASE 1: Print all paths to 4
1 5 2 3 4
1 5 7 8 9 6 4
1 6 4
1 6 9 8 7 5 2 3 4
1 3 2 5 7 8 9 6 4
1 3 4
1 8 7 5 2 3 4
1 8 9 6 4

```