

Naviqasiya

 3 san.  512 MB

$N \leq 1000$ təpədən və M tildən **birləşik, istiqamətlənməmiş sadə kaktus qrafı**¹ verilmişdir. Onun təpələrinin rəngləri var (rənglər 0-1499 aralığında olan tam ədədlərlə göstərilir). İlk olaraq bütün təpələrin rəngi 0-dır. Qrafı **deterministik yaddaşsız robot**² araşdırır. Robot bütün təpələri ən azı bir ziyarət etməlidir və sonra hərəkətini dayandırmalıdır.

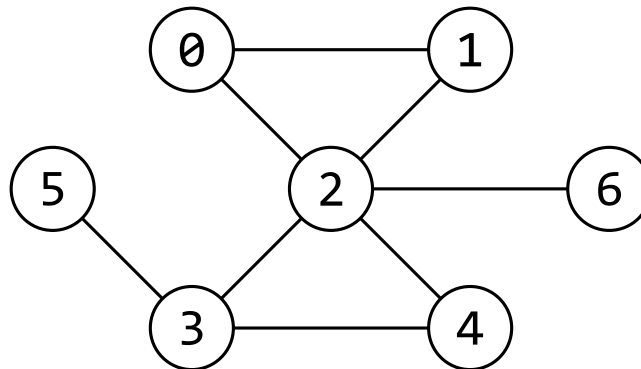
Robot istənilən təpədən başlaya bilər. Hər addımda robot olduğu təpənin rəngini və onun qonşu təpələrinin rənglərini **həmin təpə üçün sabit olan ardıcılıqla** görür (yəni təpəyə yenidən qayıdanda, qonşu təpələrin rəngləri dəyişmiş olsa belə, onların sırası dəyişməyəcək). Robot iki hərəkətdən birini edə bilər:

1. Dayanmağa qərar verir.
2. Olduğu təpəyə yeni (və ya eyni) rəng seçir və qonşulardan hansına hərəkət edəcəyini müəyyənləşdirir. Qonşu təpə 0- $D-1$ indeksləri ilə müəyyənləşdirilir, burada D olduğu təpənin qonşularının sayıdır.

İkinci halda, olduğu təpə seçilmiş rəngə boyanır (və ya eyni rəngdə qalır) və robot seçilmiş qonşu təpəyə keçir. Bu proses robot dayanmayana qədər və ya iterasiya limiti çatana qədər təkrarlanır. Robot bütün təpələrə baş çəkib $L = 3000$ addım limiti daxilində dayana bilsə, qalib gəlir, əks halda uduzur.

Sizdən tələb olunur ki, robot üçün elə bir strategiya hazırlayasınız ki, istənilən belə kaktus qrafında problemi həll edə bilsin. Bundan əlavə, istifadə olunan fərqli rənglərin sayını minimum etməyə çalışmalısınız. Burada rəng 0 həmişə istifadə olunmuş hesab olunur.

¹Əlaqəli, istiqamətlənməmiş sadə kaktus qrafı - hər bir təpədən digərinə getmək mümkündür; tillər istiqamətsizdir; özündən özünə til və təkrar tillər yoxdur; hər bir til ən çox bir sadə dövrəyə aiddir. Aşağıdakı şəkil nümunədir.



²Robot deterministik və yaddaşsızdır, yəni onun hərəkəti yalnız hazırkı gördüyü informasiyadan asılıdır və əvvəlki addımları xatırlamır. Eyni girişlər üçün eyni addımları yerinə yetirir.



İmplementasiya detalları

Robotun strategiyası aşağıdakı funksiya kimi həyata keçirilməlidir:

```
std::pair<int, int> navigate(int currColor, std::vector<int> adjColors)
```

Funksiya parametrlər kimi hazırkı təpənin rəngini və bütün qonşu təpələrin rənglərini (ardıcılıqla) qəbul edir. Funksiya dəyər olaraq qaytarmalıdır:

- Əgər robot davam etməlidirsə - yeni rəngi və keçəcəyi qonşunun indeksini cütlük kimi;
- Əgər dayanmalıdırsa - $(-1, -1)$.

Robotun hərəkətlərini seçmək üçün bu funksiya dəfələrlə çağırılacaq. Deterministik olduğundan, naviqasiya artıq bəzi parametrlərlə çağırılıbsa, bir daha həmin parametrlərlə çağırılmayacaq; əvəzinə əvvəlki qaytarma dəyəri olacaq təkrar istifadə olunur. Əlavə olaraq testlər $T \leq 5$ alt-testdən ibarət ola bilər və paralel işlədilə bilər. Bundan əlavə, proqramınızın icrası $P = 100$ dəfə baş verə bilər. Buna görə də proqram çağırışlar arasında məlumat ötürməyə cəhd etməməlidir.



Məhdudiyyətlər

- $3 \leq N \leq 1000$
- $0 \leq \text{Rəng} < 1500$
- $L = 3000$
- $T \leq 5$
- $P = 100$



Qiymətləndirmə

Alt tapşırıq üzrə toplanılan bal C - istifadə olunan fərqli rənglərin maksimal sayından asılıdır:

- Əgər robot hansısa alt-testdə uduzarsa, $S = 0$.
- Əgər $C \leq 4 \rightarrow S = 1.0$.
- Əgər $4 < C \leq 8 \rightarrow S = 1.0 - 0.6 \frac{C-4}{4}$.
- Əgər $8 < C \leq 21 \rightarrow S = 0.4 \frac{8}{C}$.
- Əgər $C > 21 \rightarrow S = 0.15$.



Alt tapşırıqlar

Alt tapşırıq	Ballar	Tələb olunan alt tapşırıqlar	N	Əlavə məhdudiyyətlər
0	0	—	≤ 300	Nümunə.
1	6	—	≤ 300	Qraf dövrədir. ¹
2	7	—	≤ 300	Qraf ulduzdur. ²
3	9	—	≤ 300	Qraf zəncirdir. ³
4	16	2 – 3	≤ 300	Qraf ağacdır. ⁴
5	27	—	≤ 300	Hər təpənin ən çox 3 qonşusu var və başlanğıc təpə yalnız 1 qonşuya malikdir.
6	28	0 – 5	≤ 300	—
7	7	0 – 6	—	—

¹Dövrə qrafı belə tillərə malikdir: $(i, (i + 1) \bmod N)$, $0 \leq i < N$.

²Ulduz qrafı belə tillərə malikdir: $(0, i)$, $1 \leq i < N$.

³Zəncir qrafı belə tillərə malikdir: $(i, i + 1)$, $0 \leq i < N - 1$.

⁴Ağac – dövrəsiz qrafdır.



Nümunə

Şərtəki şəkildə verilmiş qrafı nəzərdən keçirək. Bu qrafda $N = 7$, $M = 8$ və tillər bunlardır: $(0, 1)$, $(1, 2)$, $(2, 0)$, $(2, 3)$, $(3, 4)$, $(4, 2)$, $(3, 5)$ və $(2, 6)$. Qonşuluq siyahılarının sırası əhəmiyyətli olduğuna görə, onları ayrıca veririk:

Uc	Qonşu uclar
0	2, 1
1	2, 0
2	0, 3, 4, 6, 1
3	4, 5, 2
4	2, 3
5	3
6	2

Tutaq ki, robot 5-ci təpədən başlayır. Onda aşağıdakı (uğursuz) hərəkət ardıcılığı mümkün ola bilər:

#	Rənglər	Uc	navigate çağırışı	Geri dəyər
1	0, 0, 0, 0, 0, 0, 0	5	navigate(0, {0})	{1, 0}
2	0, 0, 0, 0, 0, 1, 0	3	navigate(0, {0, 1, 0})	{4, 2}
3	0, 0, 0, 4, 0, 1, 0	2	navigate(0, {0, 4, 0, 0, 0})	{0, 3}
4	0, 0, 0, 4, 0, 1, 0	6	¹ navigate(0, {0})	{1, 0}
5	0, 0, 0, 4, 0, 1, 1	2	navigate(0, {0, 4, 0, 1, 0})	{8, 0}
6	0, 0, 8, 4, 0, 1, 1	0	navigate(0, {8, 0})	{3, 0}
7	3, 0, 8, 4, 0, 1, 1	2	navigate(8, {3, 4, 0, 1, 0})	{2, 2}
8	3, 0, 2, 4, 0, 1, 1	4	navigate(0, {2, 4})	{1, 1}
9	3, 0, 2, 4, 1, 1, 1	3	navigate(4, {1, 1, 2})	{-1, -1}

Burada robot 6 müxtəlif rəngdən istifadə edib: 0, 1, 2, 3, 4 və 8 (qeyd: 0 başlanğıcda bütün təpələrdə olduğu üçün, hətta robot onu heç vaxt geri qaytarmasa belə, istifadə olunmuş hesab olunur). Robot 9 addım işləyib, sonra dayanıb. Lakin uğursuz olub, çünki 1-ci təpəni ziyarət etmədən dayanıb.

¹Qeyd: 4-cü iterasiyada navigate çağırışı əslində baş verməyəcəkdi. Çünki o, 1-ci iterasiyadakı çağırışla eynidir və qiymətləndirici əvvəlki cavabı istifadə edəcək. Amma bu, yenə də iterasiya sayılır.

Nümunə qiymətləndirici (Sample grader)

sample grader proqramınızın çoxsaylı icrasını işə salmır, yəni bütün navigate çağırışları eyni icra daxilində olacaq.

Giriş formatı belədir: əvvəlcə T (alt-testlərin sayı) verilir. Sonra hər alt-test üçün:

- sətir 1: iki tam ədəd - N və M ;
- sonrakı M sətirdə: iki tam ədəd - A_i və B_i , i -ci tillərin birləşdirdiyi təpələr ($0 \leq A_i, B_i < N$).

sample grader sizin həllinizin istifadə etdiyi fərqli rənglərin sayını və robot dayanmazdan əvvəl neçə iterasiya etdiyini çap edir. Əgər robot uğursuz olarsa, xəta mesajı verəcək.

Standart olaraq **sample grader** hər iterasiyada robotun nə gördüyünü və nə etdiyini ətraflı göstərir. Əgər bunu gizlətmək istəsəniz, DEBUG dəyişəninin dəyərini `true`-dan `false`-a dəyişə bilərsiniz.