Task Navigation

⏳ 3 sec. 💾 512 MB

Կա $N \leq 1000$ գագաթ և M կող ունեցող **կապակցված չուղղորդված պարզ կակտուս գրաֆ**[1]: Նրա գագաթները գույներ ունեն (նշված ոչ բացասական ամբողջ թվերով՝ 0-ից 1499): Սկզբում բոլոր կողերը 0 գույնի են: **Դետերմինացված, հիշողություն չունեցող ռոբոտը**[2] հետազոտում է գրաֆը գագաթից գագաթ շարժվելով: Նա պետք է այցելի բոլոր գագաթները, յուրաքանչյուրն առնվազն մեկ անգամ և ավարտի իր աշխատանքը:

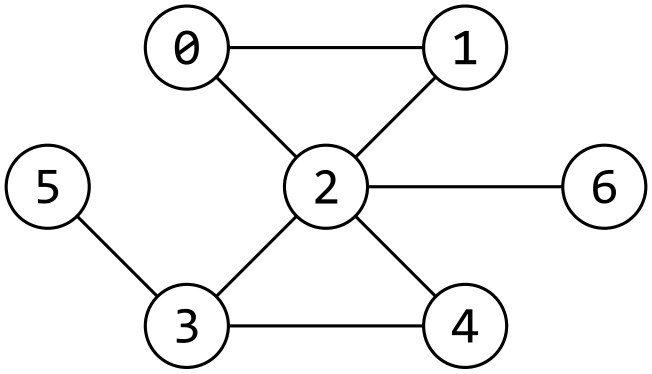
Ռոբոտը սկսում է շարժվել կամայական գագաթից: Ամեն քայլին նա տեսնում է իր ընթացիկ գագաթի գույնը և ընթացիկին հարևան բոլոր գագաթների գույները **ընթացիկ գագաթի համար ֆիքսված ինչ-որ հերթականությամբ** (այսինքն, նույն գագաթը երկրորդ անգամ այցելելու դեպքում հարևան գագաթների հաջորդականությունը նույնը կլինի, անգամ եթե նրանց գույները փոխված լինեն): Ռոբոտը կատարում է հետևյալ երկու գործողություններից մեկը.

1. Որոշում է ավարտել:
2. Ընտրում է նոր գույն ընթացիկ գագաթի համար (հնարավոր է նաև նույն գույնը թողնել) և թե որ հարևան գագաթին անցնի: Հարևան գագաթը իդենտիֆիկացվում է 0-ից $D-1$ թվով, որտեղ D-ն հարևան գագաթների քանակն է:

Երկրորդ դեպքում ընթացիկ գագաթի գույնը փոխվում է (կամ հնարավոր է, որ մնա նույն գույնը) և ռոբոտը շարժվում է դեպի ընտրված հարևան գագաթը: Սա կրկնվում է այնքան, մինչև ռոբոտը ավարտում է կամ սպառվում է քայլերի լիմիտը: Ռոբոտը հաղթում է, եթե այցելում է բոլոր գագաթները առանց քայլերի $L = 3000$ լիմիտը գերազանցելու (հակառակ դեպքում նա պարտվում է):

Դուք պետք է նախագծեք ստրատեգիա ռոբոտի համար, որպեսզի նա լուծի խնդիրը կամայական կակտուս գրաֆի համար: Բացի այդ դուք պետք է ձգտեք մինիմիզացնել ձեր լուծման կողմից օգտագործվող տարբեր գույների քանակը: Այստեղ 0 գույնը միշտ համարվում է օգտագործած:

[1] *Կապակցված չուղղորդված պարզ կակտուս գրաֆ կոչվում է այն կապակցված չուղղորդված պարզ գրաֆը (բոլոր գագաթները հասանելի են բոլոր գագաթներից, կողերը երկկողմանի են, որևէ գագաթից ինքն իրեն տանող կող չկա, երկու գագաթներ միացված են առավելագույնը մեկ կողով), որում յուրաքանչյուր կող պատկանում է առավելագույնը մեկ պարզ ցիկլի (ցիկլը կոչվում է պարզ, եթե նրանում յուրաքանչյուր գագաթ հանդիպում է առավելագույնը մեկ անգամ): Ստորև բերված է այդպիսի գրաֆի օրինակ:*



[2]*Ռոբոտը կոչվում է դետերմինացված և հիշողություն չունեցող, եթե նրա գործողությունը կախված է միայն իր ընթացիկ մուտքային տվյալներից (այսինքն, ինքը քայլից քայլ անցնելիս տվյալներ չի պահում), և նա միշտ նույն մուտքային տվյալների դեպքում կատարում է նույն գործողությունը:*

Իրականացման մանրամասներ

Ռոբոտի ստրատեգիան պետք է իրականացնել հետևյալ ֆունկցիայի տեսքով.

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

Այն որպես պարամետր ստանում է ընթացիկ գագաթի գույնը և ընթացիկի հարևան գագաթների գույները (հերթով): Այն պետք է վերադարձնի pair, որի first-ը ընթացիկ գագաթի նոր գույնն է, իսկ second-ը այն հարևան գագաթի համարն է, որին պետք է անցնի ռոբոտը: Եթե դրա փոխարեն ռոբոտը պետք է ավարտի աշխատանքը, այդ դեպքում ֆունկցիան պետք է վերադարձնի $(-1, -1)$ զույգը:

Այս ֆունկցիան կանչվելու է բազմիցս՝ ռոբոտի գործողությունները ընտրելու համար: Քանի որ այն դետերմինացված է, եթե `navigate`-ը արդեն կանչվել է որոշ պարամետրերով, այն այլևս չի կանչվի նույն պարամետրերով. դրա փոխարեն կօգտագործվի նախորդ վերադարձի արժեքը: Բացի այդ, յուրաքանչյուր թեստ կարող է պարունակել $T \leq 5$ հատ ենթաթեստ (տարբեր գրաֆներ և/կամ սկզբնական դիրքեր), և դրանք կարող են աշխատել զուգահեռաբար (այսինքն՝ ձեր ծրագիրը կարող է ստանալ հերթագայված կանչեր տարբեր ենթաթեստերի վերաբերյալ): Վերջապես, `navigate`-ի կանչերը կարող են տեղի ունենալ ձեր ծրագրի **առանձին գործարկումներում** (բայց երբեմն նաև նույն գործարկման ընթացքում): Ձեր ծրագրի գործարկումների ընդհանուր թիվը $P = 100$ է: Այս ամենի պատճառով, ձեր ծրագիրը չպետք է փորձի փոխանցել տեղեկատվություն տարբեր կանչերի միջև:

Սահմանափակումներ

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

Գնահատում

Ենթախնդրի համար տրվում է նրա համար սահմանված միավորի S մասը, որը կախված է C-ից՝ տվյալ ենթախնդրի, ինչպես նաև այն ենթախնդիրների, որոնցից ինքը կախված է, համար պատրաստված թեստերի համար ձեր ծրագրի կողմից օգտագործված գույների (ներառյալ 0 գույնը) մաքսիմալ քանակից հետևյալ կերպ.

- Եթե ձեր լուծումը որևէ ենթաթեստի վրա սխալ է աշխատում, ապա $S = 0$:
- Եթե $C \leq 4$, ապա $S = 1.0$:
- Եթե $4 < C \leq 8$, ապա $S = 1.0 - 0.6\frac{C-4}{4}$:
- Եթե $8 < C \leq 21$, ապա $S = 0.4\frac{8}{C}$:
- Եթե $C > 21$, ապա $S = 0.15$:

Ենթախնդիրներ

Ենթախնդիր	Միավոր	Պահանջվող ենթախնդիրներ	N	Լրացուցիչ սահմանափակումներ
0	0	–	≤ 300	Օրինակը
1	6	–	≤ 300	Գրաֆը ցիկլ է[1]
2	7	–	≤ 300	Գրաֆը աստղ է [2]
3	9	–	≤ 300	Գրաֆը ճանապարհ է [3]
4	16	$2-3$	≤ 300	Գրաֆը ծառ է [4]
5	27	–	≤ 300	Յուրաքանչյուր գագաթ ունի առավելագույնը 3 հարևան գագաթներ և ռոբոտը սկսում է այնպիսի գագաթից որն ունի 1 հարևան գագաթ:
6	28	$0-5$	≤ 300	–
7	7	$0-6$	–	–

[1] Ցիկլ գրաֆն ունի այսպիսի կողեր. $(i, (i+1) \bmod N)$, որտեղ $0 \leq i < N$:
[2] Աստղ գրաֆն ունի այսպիսի կողեր. $(0, i)$, որտեղ $1 \leq i < N$:
[3] Ճանապարհ գրաֆն ունի այսպիսի կողեր. $(i, i+1)$, որտեղ $0 \leq i < N-1$:
[4] Ծառ կոչվում է ցիկլ չունեցող գրաֆը:

Օրինակ

Դիտարկենք խնդրի տեքստում եղած նկարում պատկերված գրաֆը, որի համար $N = 7$, $M = 8$, և կողերն են՝ $(0,1)$, $(1,2)$, $(2,0)$, $(2,3)$, $(3,4)$, $(4,2)$, $(3,5)$ և $(2,6)$: Բացի այդ, քանի որ յուրաքանչյուր գագաթի համար հարևան գագաթների ցուցակը կարևոր է, դա ներկայացնենք աղյուսակի միջոցով.

Գագաթ	Հարևան գագաթներ
0	$2, 1$
1	$2, 0$
2	$0, 3, 4, 6, 1$
3	$4, 5, 2$
4	$2, 3$
5	3
6	2

Ենթադրենք ռոբոտը սկսում է 5 գագաթից: Այդ դեպքում հնարավոր է այսպիսի (անհաջող) փոխգործողությունների հաջորդականություն.

#	Գույներ	Գագաթ	navigate-ի կանչ	Վերադարձի արժեք
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}`

Այստեղ ռոբոտն օգտագործում է 6 տարբեր գույն. 0, 1, 2, 3, 4 և 8 (նկատենք, որ 0-ն պետք է հաշվել որպես օգտագործված անգամ եթե ռոբոտը երբեք 0 գույն չի վերադարձնում, քանի որ բոլորը սկսում են 0 գույնից): Ռոբոտը 9 քայլ է կատարում մինչև ավարտելը: Սակայն նա ձախողվում է, քանի որ չի այցելում 1 գագաթը:

[1]Նկատենք, որ `navigate`-ի կանչը 4 քայլին իրականում տեղի չի ունենում, որովհետև դա համարժեք է 1 քայլի կանչին, այնպես որ գրեյդերը պարզապես կօգտագործի այդ կանչի ժամանակ ձեր ֆունկցիայի վերադարձրած արժեքը: Սակայն դա հաշվվում է որպես ռոբոտի կատարած քայլ:

Գրեյդերի նմուշ

Գրեյդերի նմուշը ձեր ծրագիրը չի աշխատեցնում մի քանի անգամ, այնպես որ `navigate`-ի բոլոր կանչերը կատարվում են ձեր ծրագրի միևնույն գործարկման ժամանակ:

Մուտքային տվյալների ձևաչափն այսպիսին է. Սկզբում կարդացվում է T ենթաթեստերի քանակը: Ապա յուրաքանչյուր ենթաթեստի համար.

- տող 1. երկու ամբողջ N և M թվեր,
- տող $2+i$ (որտեղ $0 \le i < M$) երկու ամբողջ A_i և B_i թվեր, դրանք այն գագաթների համարներն են, որոնք իրար են կապվում i-րդ կողի միջոցով ($0 \le A_i, B_i < N$):

Հետո գրեյդերի նմուշը կտպի ձեր ծրագրի կողմից օգտագործած տարբեր գույների քանակը և քայլերի քանակը, որ նրան պետք են նախքան աշխատանքն ավարտելը: Եթե ձեր ծրագիրը սխալվի, գրեյդերի նմուշը կտպի հաղորդագրություն սխալի մասին:

Լռելյայն գրեյդերի նմուշը տպում է մանրամասն տեղեկություն թե ինչ է տեսնում ռոբոտը և ինչ է անում յուրաքանչյուր քայլին: Դուք կարող եք անջատել դա, `DEBUG`-ի արժեքը `true`-ից դարձնելով `false`: