



Задача Speedrun

C++ хедърен файл speedrun.h

Марсел започна да прави спийдрънове (минаване на игра за много малко време), надявайки се да постави нов Wr (World Record - световен рекорд). За жалост, той не може да овладее новите тактики, които е видял и затова трябва да намери начин да получи нечестно предимство.

Играта, на която се опитва да стане добър, се нарича *Tree Souls III* и той се опитва да получи Wr в категорията *full-tree%*.

Както може да се разбере от името, играта се развива в дърво (т.е. граф с N върха, номерирани с числата от 1 до N и $N - 1$ ребра, така че да няма цикли).

Играта протича по следния начин: героят е поставен в произволен връх на дървото. Той може да избере цяло число x и да опита да премине от върха, на който се намира, към върха с номер x . Ако има ребро между x и върха, на който е героят, той ще се придвижи към връх x , иначе нищо не се случва. Неговият спийдрън е валиден само ако види всеки връх поне веднъж.

Тук идва частта с маменето: понеже той не знае бгг с телепортацията по ребрата, то той иска да сложи сийд за света, в който се намира и по този начин той ще знае как изглежда дървото преди да е започнал минаването. Ако той просто запомни дървото, то ще е твърде очевидно, че мами и незабавно ще бъде отстранен от общността на спийдрънарите, затова той само ще постави подсказки във всеки връх (като подправи програмния код). Подсказка наричаме двоичен низ с големина l (l е едно и също за всяка подсказка). Когато Марсел се намира в някой връх, той може да прочете подсказката, която е сложена там.

Протокол за интеракция

Тази задача има две интеракции!

Вашата програма ще бъде изпълнена два пъти. Първият път е за изпълняване на първата интеракция (*фазата с поставянето на подсказки*), а вторият път е за втората интеракция (*фазата на спийдръна*).

Трябва да напишете следните функции (не трябва да пишете функция `main`).

```
void assignHints(int subtask, int N, int A[], int B[]);  
void speedrun(int subtask, int N, int start);
```

Може да викате следните функции:

```
void setHintLen(int l);  
void setHint(int i, int j, bool b);  
int getLength();  
bool getHint(int j);  
bool goTo(int x);
```

Фаза с поставяне на подсказки. Това е първото изпълнение на вашата програма. По време на тази фаза, програмата на журито ще извика вашата функция `assignHints` точно веднъж. Ще получите стойност на параметъра *subtask*, който ще е индекса на подзадачата, стойност на броя върхове в параметъра N и ребрата в параметрите A и B по следния начин: за всяко $i = 1, \dots, N - 1$ се задава ребро на дървото между $A[i]$ и $B[i]$. По време на работата на вашата функция, трябва първо да извикате функцията `setHintLen` точно веднъж, за да зададете дължината на подсказките,



които сте решили да използвате, като параметър на тази функция. След като сте извикали тази функция, може да започнете да викате функцията `setHint(i, j, b)` многократно. Едно извикване задава стойността на бит с индекс j на b за върха на дървото с номер i . Битовете на един връх са индексирани от 1 до l . В началото всички битове на подсказките на върховете са нули. По време на тази фаза, не трябва да викате функциите `getLength`, `getHint` и `goTo`. Функцията `assignHints` трябва да завърши работа, след като сте приключили с поставянето на хинтове.

Фаза на спийдрън. Това е второто изпълнение на вашата програма. По време на тази фаза, програмата на журито ще извика функцията `speedrun(subtask, N, start)` точно веднъж. Параметърът `start` е за номера на връх, от който започва героят, а вторият параметър задава стойността на броя върхове N . Допълнително, първият параметър `subtask` отново задава индекса на подзадачата. В тази функция, може да извиквате функциите `getLength()`, която ще върне l - дължината на подсказките, които сте слагали в първата фаза, функцията `getHint(j)`, която ви връща стойността на бит j за подсказката в текущия връх и също функцията `goTo`. Ако параметърът, който се подава на функцията `goTo` е номер на връх, който е свързан с ребро към текущия връх, то функцията връща `true` и героят се премества в този връх. Иначе функцията връща `false` и вие не се премествате от текущия връх. По време на тази фаза, не трябва да викате функциите `setHint` и `setHintLen`. Функцията `speedrun` следва да завърши, след като сте видели всички върхове в дървото.

Ограничения

- $1 \leq N \leq 1000$
- Нека означим с Q , броят на извикванията на `goTo`, които са върнали `false`. Следващият ферман описва ограничения за l и Q , които трябва да бъдат спазени, за да получите точки за дадена подзадача.

Подзадача 1 (21 точки)

- $l \leq N$
- $Q \leq 2000$
- За тази подзадача ще получите 21 точки, ако всички тестове са преминали успешно, а иначе ще получите 0.

Подзадача 2 (8 точки)

- $l \leq 20$
- $Q \leq 2000$
- Дървото е звезда, т.е. има връх x ($1 \leq x \leq N$), който е свързан с всеки друг връх.
- За тази подзадача ще получите 8 точки, ако всички тестове са преминали успешно, а иначе ще получите 0.

Подзадача 3 (19 точки)

- $l \leq 20$
- $Q \leq 2000$
- Степента на всеки връх е най-много 2.
- За тази подзадача ще получите 19 точки, ако всички тестове са преминали успешно, а иначе ще получите 0.

Подзадача 4 (12 точки)

- $l \leq 316$



- $Q \leq 32000$
- За тази подзадача ще получите 12 точки, ако всички тестове са преминали успешно, а иначе ще получите 0.

Подзадача 5 (40 точки)

- $Q \leq 2000$
- Точките s за всеки тест в подзадачата ще се изчисляват по следния начин. Ако $l > 40$, то $s = 0$. Ако $l \leq 20$, то $s = 40$. В противен случай, $s = 60 - l$. По този начин, ако $l = 40$, то ще получите половината точки за теста, ако $l \leq 20$, то ще получите пълен брой точки за теста. Точките, които ще получите за подзадачата ще са равни на минималната стойност s от всички тестове в подзадачата.

Примерна интеракция

По време на първата интеракция (фазата по поставяне на подсказки), функция на състезателя ще бъде извикана по следния начин:

```
assignHints(  
    /* subtask = */ 1,  
    /* N       = */ 5,  
    /* A       = */ {-, 1, 2, 3, 3},  
    /* B       = */ {-, 2, 3, 4, 5});
```

Тази функция може да реши да сложи стойност на $l = 2$:

```
setHintLen(/* l = */ 2);
```

След което, функцията може да реши да остави всички битове на подсказките на 0, освен бит с индекс 1 на връх 2, който се задава със стойност 1:

```
setHint(  
    /* i = */ 2,  
    /* j = */ 1,  
    /* b = */ 1);
```

След което функцията приключва работа и след това първото изпълняване на програмата на състезателя ще бъде прекратено и по този начин всякакви данни, които са били запазени в паметта от това изпълнение на програмата ще бъдат загубени. Нека обобщим, подсказките са "00" за всеки връх освен връх номер 2, а в него подсказката е "10".

По време на втората интеракция (фазата на спийдръна), функция на състезателя ще бъде извикана по следния начин:

```
speedrun(  
    /* subtask = */ 1,  
    /* N       = */ 5,  
    /* start   = */ 1);
```

Тази функция започва да прави спийдрън на дървото:

```
getLength(); // = 2  
getHint(1);  // = 0  
getHint(2);  // = 0  
goTo(2);     // = true  
getHint(1);  // = 1  
getHint(2);  // = 0
```



Като обобщение на горните извиквания: в началото героят е бил във връх 1 и е открил, че $l = 2$ и подсказката във връх 1 е "00", след което успешно се премества на връх 2 и научава, че подсказката във връх 2 е "10". Изпълнението на тази функция може да продължи по следния начин:

```
goTo(2);    // = false ( не може да отидете от връх 2 отново в него )  
goTo(5);    // = false ( няма ребро от връх 2 до връх 5 )  
goTo(3);    // = true  
goTo(4);    // = true  
goTo(3);    // = true  
goTo(5);    // = true
```

По това време всички върхове на дървото са видени и така функцията `speedrun` може да приключи работа. Стойността на $Q = 2$, защото има 2 извиквания на `goTo`, които са върнали `false`.