

### Задача 1. Монопол

Дени многу ужива во играњето на играта монопол. Но, времетраењето на една игра е многу долго - од 5 до 7 часа. Затоа, Дени почнува да размислува за промена на класичните правила. Играчите во монопол обично се движат во една насока преминувајќи од поле на поле и по извесно време се враќаат повторно на почетокот, па пак почнуваат од таму и така натаму. Во новата верзија, движењето повторно ќе биде од поле на поле, но со таа разлика што од сегашната позиција може да има неколку можности за следниот потег. Дени сака да најде такви насочени врски меѓу полињата што ќе му овозможат на играчот никогаш да не се врати на полето каде што бил, без разлика за тоа како се движи (се разбира следејќи ги правилата). На овој начин играта ќе трае пократко.

Дени веќе почна да ја прави новата табла: го избра бројот на полиња  $N$  (полињата се нумерирани од 1 до  $N$ ) и направи листа со  $M$  врски (секоја врска има насока и не постои врска што поврзува едно поле само со себе). Ако полето  $i$  е поврзано со полето  $j$ , тогаш не постои насочена врска во спротивна насока, т.е. од полето  $j$  до полето  $i$ , а исто така нема други насочени врски од полето  $i$  до полето  $j$ . Кога Дени мислеше дека ја има подготвено новата табла, одеднаш забележа дека состојбата што ја сака (при движењето од поле на поле со користење на врските не можете да се вратите на претходно посетеното поле) не важи за нејзината листа на врски. Таа прво помислила да отстрани некои од насочените врски, но тоа ќе значи повторно пишување на листата што може да биде долг процес. Затоа Дени реши да ја смени насоката на некои од насочените врски.

#### Задача

Бидејќи редовно играш монопол со Дени сакаш да и помогнеш со пишување на програмата монопол, која треба да и каже кои врски да ги промени за да се одржи наведената состојба. Програмата треба да ја содржи функцијата `find_reverse` која ќе се компајлира со програмата на жирито.

#### Имплементациски детали

Функцијата `find_reverse` мора да го има следниов прототип:

```
std::string find_reverse (int N, int M, int connections[][2]);
```

Оваа функција се повикува само еднаш од програмата на жирито со три параметри:  $N$  – бројот на празни полиња во новата табла,  $M$  – бројот на насочени врски во списокот на Дени и врските на низата кои имаат  $M$  редови од кои секој се состои од два броја  $x$  и  $y$  – почетното и завршното поле за насочената врска. Оваа функција треба да врати бинарна низа со должина  $M$  – по редоследот на врските на низата, каде за секоја врска треба да ставите „1“ во низата ако врската треба да се преврти (да биде во спротивна насока) и „0“ во спротивно. Ако има повеќе од едно решение, може да го вратите кое било од нив.

До системот треба да ја испратите датотеката `monopoly.cpp` која содржи имплементација на функцијата `find_reverse`. Датотеката може да содржи други функции и код кои и се потребни на Вашата програма, но не смее да содржи главна функција – `main`. Исто така, не треба да се обидуваме да читате од стандарден влез ниту да се обидуваме да пишувате на стандарден излез!

#### Услови

- ♣  $3 \leq N \leq 5 \times 10^5$
- ♣  $3 \leq M \leq 1.5 \times 10^6$

### Подзадачи

Подзадача	Поени	$N$	$M$	Дополнителни ограничувања
1	0	–	–	Примерот
2	15	$\leq 7$	$\leq 21$	–
3	40	$\leq 10^3$	$\leq 5 \times 10^3$	–
4	25	$\leq 10^5$	$\leq 5 \times 10^5$	–
5	20	$\leq 5 \times 10^5$	$\leq 1.5 \times 10^6$	–

За да ги добиете поените за дадена подзадача потребно е Вашето решение да ги помине сите тестови за таа подзадача.

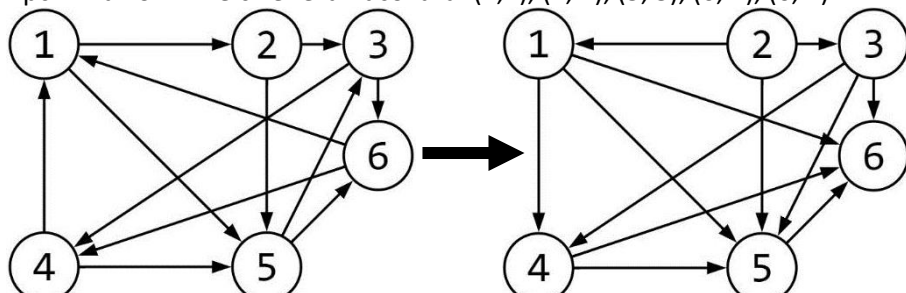
### Локално тестирање

За локално тестирање ви е дадена датотеката Lgrader.cpp. Потребно е да го ставите во истата папка со Вашата програма monopoly.cpp и треба да ги компајлирате датотеките Lgrader.cpp и monopoly.cpp заедно. На таков начин ќе направите програма за проверка на исправноста на Вашата функција. Програмата на стандарден влез ќе ја бара следната низа од податоци:

- во првиот ред: два цели броеви – бројот на празни места  $N$  и бројот на врски  $M$  на новата табла
- на следните  $M$  линии: на секоја линија треба да има два цели броја  $x$  и  $y$  – почеток и крај на секоја насочена врска.

Излезот на програмата ќе биде бинарната низа што треба да ја најдете.

### Пример од локално тестирање

Влез	Излез	Објаснување
6 12 1 2 1 5 2 3 2 5 3 4 3 6 4 1 4 5 5 3 5 6 6 1 6 4	100000101011	<p>Врски на кои им е сменета насоката: (1,2), (4, 1), (5, 3), (6, 1), (6, 4).</p>  <p>Сликата погоре ги прикажува полињата, прво, со насочените врски од списокот на Дени и, второ, откако ќе се смени насоката на врските со соодветното „1“ од излезот. Јасно е дека на почетокот користејќи ги насочените врски (1, 2), (2, 3), (3, 4) и (4, 1) можеме да почнеме од полето 1 и движејќи се по овие насочени врски ќе се вратиме повторно на тоа поле. Може да се види дека состојбата дефинирана од Дени важи во добиената табла. Забележете дека има и други валидни решенија:</p> <p>(4, 1), (5, 3), (6, 1), (6, 4)</p> <p>(1, 5), (2, 3), (2, 5), (3, 4), (3, 6), (4, 5), (5, 6)</p>