­­Даден е свързан непретеглен граф с *N* върха и *M* двупосочни ребра. Прост път наричаме последователност от $K\geq 1$ върха *V1, V2, …, VK*, такива че $V\_{i}\ne V\_{j}$ при $i\ne j$ и в графа има ребро между $V\_{i}$ и $V\_{i+1}$ за *i=1, 2, …, K-1*.

Дефинираме стойността на даден прост път *V1, V2, …, VK*, като

$\sum\_{i=1}^{K}i×V\_{i}$. Напишете програма **maxpath**, която намира път с колкото може по-голяма стойност.

**Вход**

От първия ред на файла maxpath.in се въвеждат две цели положителни числа *N* и *M* – съответно броя върхове и броя ребра в графа. От следващите *M* реда се въвеждат по две цели положителни числа, описващи съответното ребро на графа. Гарантирано е, че няма ребро от връх към себе си и няма ребра, описани повече от веднъж във входа.

**Изход**

На първия ред на изходния файл maxpath.out изведете едно цяло положително число *K* – броя на върховете в избрания от вас прост път. На следващите *K* реда изведете по едно цяло положително число, описващо текущия връх в избрания от вас път.

**Оценяване**

Ако изходът не съвпада с описания формат или изведените от вас върхове не образуват прост път, ще получите 0 точки за съответния тест.

В противен случай ще получите $score × (\frac{yours+1}{best+1})^{2}$ точки, където *score* са точките предвидени за съответния тест, *yours* е стойността на намерения от вас път, а *best* е най-голямата стойност на път, намерена от участник за съответния тест.

**Ограничения**

|  |  |
| --- | --- |
| Част от тестовете | Ограничения за N и M |
| 30% | $$N=100, M\in [\frac{N\left(N-1\right)}{40}; \frac{N\left(N-1\right)}{5}]$$ |
| 30% | $$N=1 000, M\in [\frac{N\left(N-1\right)}{40}; \frac{N\left(N-1\right)}{10}]$$ |
| 40% | $$N=100 000, M\in [200 000; 500 000]$$ |

Във всяка от трите групи от таблицата по-горе, половината от тестовете ще се генерират с първия алгоритъм, а другата половина – с втория алгоритъм, посочени по-долу.

**Генериране на тестовете**

При генерирането на графите са използвани два алгоритъма:

* *Алгоритъм 1:* Генерираме дърво, като на всеки връх, различен от *1*, поставяме произволен родител с по-малък номер. Към получения граф добаряме произволни ребра *(x,y)*, стига те да не са добавени преди, докато общия брой ребра не стане равен на *M*. Накрая номерата на върховете се разбъркват произволно.
* *Алгоритъм 2:* Генерираме произволно число *T1* от *1* до *N*, след това произволно число *T2*от *T1+1* до *N* и така нататък, докато *Tk* не стане равно на *N*. Образуваме *K* пътечки (навръзваме върховете с ребра в указания ред): *{1, 2, …, T1}, {T1+1, T1+2, …, T2}, …, {Tk-1+1, …, Tk=N}*. От всяка пътечка, различна от първата, избираме произволен връх и добавяме ребро между него и произволен връх от предишните пътечки. Към получения граф добаряме произволни ребра *(x,y)*, стига те да не са добавени преди, докато общия брой ребра не стане равен на *M*. Накрая номерата на върховете се разбъркват произволно.

**Ограничение по време: 5 сек**

**Ограничение по памет: 256 MB**

**Примерен тест**

|  |  |
| --- | --- |
| **Вход (maxpath.in)** | **Изход (maxpath.out)** |
| 5 51 22 32 52 43 5 | 44235 |

Предложеният изход би дал резултат 37.

