

CEOI 2026



Dan 2
(Slovenščina)

Day 2
(Slovenian)

Rezanje cvetlic

Omejitev časa: 4 s Omejitev pomnilnika: 256 MiB

Na vrtu skupnosti CEOI gojimo zbirko posebnih cvetlic, ki svoje korenine tesno prepletejo med seboj. Če jih prerežemo, se zarastejo nazaj, razen če ne režemo preveč v živo in povzročimo nepopravljivo škodo.

Če dve cvetlici, a in b , prepleteta svoje korenine skupaj, rečemo, da sta *povezani*, sicer pa sta *nepovezani*. Korenine se zaraščajo po naslednjem pravilu: naj bosta a in b dve nepovezani cvelici. Če obstajata vsaj dve drugi cvetlici c in d , za kateri velja, da je vsaka izmed a in b povezana tako s c kot z d , se bodo korenine cvetlic a in b zrasle in postali bosta povezani.

Te cvetlice zdaj rastejo že nekaj časa in vse korenine, ki so se lahko zarastle po gornjem pravilu, so se tudi res že zarastle. Z drugimi besedami, če sta zdaj cvetlici a in b obe povezani z nekima dvema cvetlicama c in d , potem sta a in b gotovo tudi neposredno povezani med seboj.

Vrt moramo zdaj izkopati, da ga bomo preselili na prizorišče naslednje CEOI. Da olajšamo to selitev, bi radi prerezali karseda veliko korenin, vendar pa hočemo, da se bodo lahko cvetlice kasneje zarasle nazaj v svoje sedanje stanje. Koliko največ povezav med pari cvetlic lahko prerežemo, tako da se bodo kasneje še vedno zarasle nazaj v svoje sedanje stanje? Pri tem je vseeno, koliko korakov rasti bo treba, da se zarastejo nazaj.

Vhod

V prvi vrstici sta dve s presledkom ločeni celi števili, n (število cvetlic) in m (število obstoječih povezav med njimi). Sledi m vrstic, vsaka od njih pa vsebuje celi števili a_i in b_i , ki povesta, da sta cvetlici a_i in b_i povezani. Cvetlice so oštevilčene s celimi števili od 1 do n . Zagotovljeno je, da vhodni podatki ustrezajo pravilu iz opisa naloge.

Omejitve

- $1 \leq n \leq 1000$
- $1 \leq m \leq 10^5$

Podnaloge

- 1. podnaloga (20 točk): $n \leq 10$ in $m \leq 20$.
- 2. podnaloga (14 točk): $m = \frac{n \cdot (n-1)}{2}$.
- 3. podnaloga (15 točk): vsaka cvetlica je povezana z največ 7 drugimi cvetlicami.
- 4. podnaloga (15 točk): $n \leq 50$ in $m \leq 1000$.
- 5. podnaloga (36 točk): brez dodatnih omejitev.

Izhod

Izpiši eno samo celo število — največje število povezav, ki jih smemo prerezati.

Primer

Vhod

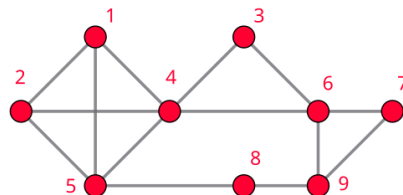
9 14
 1 2
 1 4
 1 5
 2 4
 2 5
 3 4
 4 5
 3 6
 4 6
 6 7
 6 9
 7 9
 8 9
 5 8

Izhod

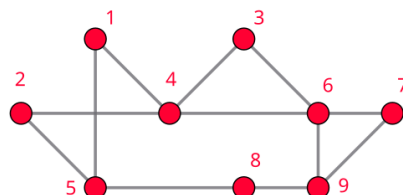
2

Komentar

Naslednja slika kaže cvetlice v gornjem primeru pred rezanjem. Lahko se prepričamo, da na podlagi pravila iz besedila naloge ne more nastati nobena nova povezava.



Na naslednji sliki imajo cvetlice po rezanju dve povezavi manj. Povezava med 1 in 2 se lahko zaraste nazaj, ker sta cvetlici 1 in 2 obe povezani s cvetlicama 4 in 5. Podobno se lahko povezava med 4 in 5 zaraste nazaj, ker sta cvelici 4 in 5 obe povezani s cvetlicama 1 in 2.



Stolpi

Omejitev časa: 1 s Omejitev pomnilnika: 256 MiB

Imaš n računalnikov in m stolpov, razporejenih na samih različnih položajih vzdolž neke premice. Računalnike moraš s kabli povezati v pare tako, da se vsak kabel začne pri nekem računalniku, obišče katerekoli stolpe (ne samo te med računalnikoma) in se konča pri nekem drugem računalniku. Kabel lahko stolpe obišče v poljubnem vrstnem redu; lahko gre tudi mimo kakšnega stolpa, ne da bi ga obiskal; lahko tudi sploh ne obišče nobenega stolpa in poveže računalnika neposredno, ne more pa povezati računalnika s samim sabo. Število računalnikov je sodo.

Naj bosta a in b položaja dveh računalnikov, povezanih s takšnim kablom, in naj bodo $x_1, \dots, x_k$ položaji stolpov, ki jih kabel obišče. Dolžina kabla je potem $|a - x_1| + |x_1 - x_2| + \dots + |x_{k-1} - x_k| + |x_k - b|$. *Oceno* kabla definiramo kot $f \cdot u - l$, kjer je l dolžina kabla, f je neka fiksna konstanta, u pa je število različnih stolpov, ki jih kabel obišče (torej število različnih stolpov v $x_1, \dots, x_k$). Več kablov lahko obišče isti stolp in ta stolp prispeva k oceni vseh teh kablov.

Napiši program, ki izračuna največjo možno vsoto ocen kablov, s katerimi lahko vse računalnike povežemo v pare (vsak računalnik mora torej pripadati natanko enemu paru ali, kar je ekvivalentno, vsak računalnik mora biti povezan z natanko enim kablom).

Vhod

V prvi vrstici je T , število testnih primerov. Sledijo testni primeri eden za drugim. Vsak testni primer obsega tri vrstice. V prvi od teh vrstic so tri cela števila: n (število računalnikov), m (število stolpov) in konstanta f . V drugi vrstici je n celih števil $a_1, a_2, \dots, a_n$ — položaji računalnikov. V tretji vrstici je m celih števil $b_1, b_2, \dots, b_m$ — položaji stolpov.

Omejitve

Naj bosta N oz. M vsoti vrednosti n oz. m po vseh testnih primerih.

- $1 \leq T \leq 10^4$
- $1 \leq N, M \leq 2 \cdot 10^5$
- $0 \leq f \leq 10^9$
- Vsak n je sodo število.
- $1 \leq a_i, b_i \leq 10^9$
- V okviru posameznega testnega primera so vsi položaji računalnikov in stolpov različni.

Podnaloge

1. podnaloga (5 točk): $N \leq 5000$, $m = 1$.
2. podnaloga (10 točk): $T \leq 20$, $n \leq 10$, $m \leq 100$.
3. podnaloga (27 točk): $N, M \leq 5000$.

- 4. podnaloga (21 točk): $N \leq 5000$.
- 5. podnaloga (37 točk): brez dodatnih omejitev.

Izhod

Izpiši T celih števil, vsako v svoji vrstici — največjo možno vsoto ocen kablov za vsak testni primer.

Primer

Vhod

```
4
2 1 100
1 10
11
4 1 10
2 4 6 8
20
4 1 10
2 4 6 8
5
6 3 10
2 13 4 8 6 10
5 1 9
```

Izhod

```
89
-4
12
51
```

Vim

Omejitev časa: 1 s Omejitev pomnilnika: 256 MiB

Težko bi našli kaj bolj *geekovskega*, kot je stari Vim — urejevalnik, v katerem lahko počnemo reči, o katerih bi v »običajnih« urejevalnikih kvečjemu sanjali, če smo pripravljeni vložiti kak teden učenja in še kak mesec vaje, da se nam nenavadni, a močni ukazi, ki jih ponuja, usidrajo v »mišični spomin«. No, tako kot pri drugih urejevalnikih imamo tudi pri Vimu *kurzor*, ki se na začetku nahaja na prvem znaku besedila, in tako kot pri drugih urejevalnikih tudi pri Vimu obstaja *odložišče* (angl. *clipboard*), ki je namenjeno shranjevanju (ter posredno kopiranju in premikanju) kosov besedila. Odložišče je na začetku prazno.

Pri tej nalogi bomo predpostavili, da se v urejevalniku na začetku nahaja natanko en znak '-', naš cilj pa je končati z natanko n zaporednimi minusi. Pri tem si bomo pomagali s štirimi ukazi:

- **h**: Če se kurzor nahaja na prvem znaku, potem ta ukaz ne naredi ničesar, sicer pa premakne kurzor za en znak v levo.
- **l**: Če se kurzor nahaja na zadnjem znaku, potem ta ukaz ne naredi ničesar, sicer pa premakne kurzor za en znak v desno.
- **Y**: Zaporedje znakov, ki se razteza od vključno položaja kurzorja do konca besedila, skopira v odložišče, pri čemer »povozi« morebitno predhodno vsebino odložišča. (Če se kurzor nahaja na prvem znaku besedila, se bo v odložišče torej skopiralo celotno besedilo.)
- **P**: Kopijo besedila, shranjenega v odložišču, vstavi pred znak, na katerem se nahaja kurzor, in kurzor premakne na zadnji vstavljeni znak. Vsebina odložišča se ne spremeni. Če je odložišče prazno, se ne zgodi nič.

Napiši program, ki prebere število n , izpiše pa minimalno število ukazov, ki jih potrebujemo, da v Vimu pod opisanimi pogoji končamo s točno n zaporednimi znaki '-'. Program naj izpiše tudi primer zaporedja z minimalnim številom ukazov.

Vhod

Prva vrstica vsebuje število testnih primerov t . V naslednjih t vrsticah so podani testni primeri z iskano dolžino niza n .

Omejitve

- $1 \leq t \leq 100$.
- $1 \leq n \leq 10^7$.

Podnaloge

Če izpišeš samo pravilno število ukazov, ne pa tudi primera zaporedja ukazov, ali napačno zaporedje ukazov, boš pri tisti podnalogi prejel polovico točk.

- 1. podnaloga (20 točk): $n \leq 100$.
- 2. podnaloga (8 točk): $n \leq 1000$.
- 3. podnaloga (18 točk): $n \leq 10^4$.
- 4. podnaloga (18 točk): $n \leq 10^5$.
- 5. podnaloga (18 točk): $n \leq 10^6$.
- 6. podnaloga (18 točk): brez dodatnih omejitev.

Izhod

Za vsak testni primer izpiši v svoji vrstici iskano minimalno število ukazov in primer takega zaporedja ukazov.

Primer

Vhod

2
21
2

Izhod

10 YPYPhPYPPP
2 YP

Komentar

Kot vidimo v naslednji tabeli, nas v prvem primeru do cilja privede zaporedje YPYPhPYPPP. Znak '=' v stolpcu Zaslon predstavlja znak '-', na katerem se nahaja kurzor.

Korak	Ukaz	Zaslon	Odložišče
0		=	(prazno)
1	Y	=	-
2	P	=-	-
3	Y	=-	--
4	P	=---	--
5	h	=---	--
6	P	=----	--
7	Y	=----	-----
8	P	-----=-----	-----
9	P	-----=-----	-----
10	P	-----=-----	-----