Rekurzivni otoki

V severozahodni Kanadi je jezero, v katerem je otok, na katerem je jezero, v katerem je otok, na katerem je jezero, v katerem je otok. Takšnemu geološkemu pojavu pravimo *rekurzivni otok* in mu lahko pripišemo globino rekurzije, tj. koliko jezer obkroža naš otok. Zgoraj omenjeni otok ima največji nivo rekurzije na svetu -3.

Naloga

Napiši program, ki bo sprejel zemljevid dela površja in poiskal največjo globino rekurzije otoka na zemljevidu. Zemljevid je podan kot tabela znakov, kjer `.` (pika) predstavlja vodo, znak `#` pa predstavlja kopno. Vsi štirje robovi so vedno v celoti prekriti z znaki `#` (na robu je celina). Smatramo, da lahko med dvema poljema kopnega prehajamo, če sta sosednji v eni od štirih kardinalnih smeri (navzgor, navzdol, levo, desno). Če se dve kopenski masi stikata le diagonalno, ju smatramo kot različna otoka.

Vhod

V prvi vrstici vhoda sta dve števili, - W in H, ki označujeta širino in višino zemljevida. V naslednjih H vrsticah s po W znaki se nahaja zemljevid, kakor je opisan zgoraj.

Omejitve

Velja $3 \leq W, H \leq 1000$.

Podnaloge

1. podnaloga (18 točk): $W \leq 5$ in $H \leq 5$
2. podnaloga (18 točk): $W \leq 5$ ali $H \leq 5$
3. podnaloga (22 točk): $W \cdot H \leq 1000$
4. podnaloga (42 točk): Ni dodatnih omejitev.

Izhod

Program naj izpiše eno samo številko - največjo globino rekurzije otoka na sliki. Če na sliki ni otokov, izpiši 0.

Primera

1. primer

Vhod

```
10 9
##########
#........#
#.######.#
#..#...#.#
#..#.#.#.#
#.##...#.#
#.######.#
#........#
##########
```

Izhod

```
2
```

2. primer

Vhod

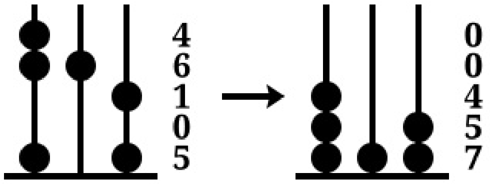
```
10 9
##########
#........#
#..####..#
#.#...#..#
#.#.#.#..#
#.#...#..#
#.#####..#
#........#
##########
```

Izhod

```
1
```

Urejanje z abakom

Avstralski fizik David Morgan-Mar si je izmislil nenavaden algoritem za urejanje seznamov števil, ki temelji na abaku (pripomočku za računanje iz palic, na katerih so nanizane kroglice). Spodnja slika prikazuje primer urejanja za seznam $4, 6, 1, 0, 5$. Števila najprej zapišemo v dvojiškem sistemu, vodoravno oz. vsako v svoji vrstici: kroglica predstavlja 1, prazen prostor pa 0. Na najbolj levi palici je tista dvojiška števka, ki označuje največjo potenco števila 2. Nato abak postavimo pokonci, da kroglice zdrsnejo na dno. Tako dobimo urejen seznam, ki, kot je vidno na sliki, ne vsebuje nujno istih števil kot prvotni seznam.



Radi bi analizirali učinkovitost tega nadvse uporabnega algoritma. Predpostavimo, da zaradi trenja kroglice drsijo z enakomerno hitrostjo: v vsaki časovni enoti se vsaka kroglica premakne eno vrstico nižje, dokler ne trči ob dno ali drugo kroglico. Urejamo seznam $a_1, a_2, \ldots, a_n$. Čez koliko časa so vse kroglice na svojih končnih položajih?

Vhod

V prvi vrstici je število n (dolžina seznama). V naslednjih n vrsticah so števila $a_1, a_2, \ldots, a_n$ (elementi seznama).

Omejitve

- $1 \leq n \leq 5 \cdot 10^4$
- $0 \leq a_i < 2^{31}$ za vsak $i = 1, \ldots, n$

Izhod

Izpiši čas drsenja kroglic.

Podnaloge

1. (15 točk) $n = 2$
2. (25 točk) $0 \leq a_i \leq 1$ za vsak $i = 1, \ldots, n$
3. (60 točk) Brez dodatnih omejitev.

Primer

Vhod:

```
5
4
6
1
0
5
```

Izhod:

```
3
```

Ta primer je prikazan na sliki v opisu naloge. Najdlje pada kroglica na srednji palici.

Burek King

V Dolini pod tremi vrhovi teče bistra reka, ob kateri se vije cesta, ob kateri je enakomerno razporejenih N idiličnih vasi. Vsako izmed njih krasi generično ime: "V Dolini pod tremi vrhovi i", kjer $1 \leq i \leq N$ označuje zaporedno številko vasi vzdolž ceste ob Reki pod tremi vrhovi.

V Dolini je K restavracij s hitro prehrano, ki se nahajajo v vaseh $A_1, \ldots, A_K$. V eni vasi je lahko največ ena restavracija. Prebivalci Doline obožujejo hitro prehrano, najljubša pa jim je tista, ki je najhitrejša oziroma jo ponujajo v vasi, ki jim je najbližje. Če sta dve restavraciji na enaki razdalji od vasi, se polovica prebivalcev odloči za obisk ene restavracije, druga pa zaide v drugo.

Admin, najboljši programer te Doline, si je v inozemstvu prislužil goro denarja, ki bi jo rad oplemenitil. Odprl bo M burekdžinic Burek King in želi, da večina (več kot polovica) prebivalcev Doline obiskuje burekdžinico. Iz registrov je zbral podatke o številu prebivalcev v vaseh in jih zapisal kot X_i, kjer i označuje zaporedno številko vasi. Kot po naključju je v vsaki vasi sodo število prebivalcev.

Admin je skrben gospodar, zato ne želi graditi burekdžinic vsepovprek. Je pa žal len in kode ne bo pisal sam, zato se je odločil, da nameni del svojih sredstev za izvedbo programerskega tekmovanja v zameno za vpogled v rešitve tekmovalcev. Zato te prosi, da napišeš program, ki poišče najmanjšo možno vrednost M, da bo večina prebivalcev Doline obiskovala njegovo burekdžinico, v zameno pa ti bo zagotovil točke na tekmovanju. Če tak M ne obstaja, naj program izpiše -1.

Vhod

V prvi vrstici standardnega vhoda sta celi števili N in K.

V drugi vrstici je K s presledkom ločenih celih števil A_1 A_2 ... A_K.

V tretji vrstici je N s presledkom ločenih sodih števil X_1 X_2 ... X_K ... X_N.

Omejitve

- $2 \leq K \leq N \leq 500000$
- $0 \leq X_i \leq 10^9$ za vsak $1 \leq i \leq N$
- X_i je sodo število.
- $1 \leq A_i \leq N$ za vsak $1 \leq i \leq K$

Podnaloge

1. podnaloga (17 točk): $N \leq 10$
2. podnaloga (33 točk): Vse vasi imajo enako število prebivalcev - $X_1 = X_2 = \ldots = X_N$.
3. podnaloga (21 točk): $N \leq 3000$
4. podnaloga (29 točk): Ni dodatnih omejitev.

Izhod

Na standardni izhod izpiši najmanjši M. Če ne obstaja, izpiši -1.

Primeri

1. primer

Vhod

```
5 2
2 5
4 4 4 4 4
```

Izhod

```
3
```

Komentar

V mestih 1, 3 in 4 mora odpreti restavracijo in v Burek Kingu bo jedlo 12 ljudi.

1. primer

Vhod

```
4 2
1 4
2 2 8 4
```

Izhod

```
1
```

Komentar

Najbolje je postaviti burekdžinico v vas 3. Tedaj ima osem obiskovalcev iz vasi 3 in enega iz vasi 2.

Sušenje rjuh

Opral sem r enakih rjuh, sedaj pa bi jih rad posušil na v enakomerno razmaknjenih vrveh, ki so enako dolge kot rjuhe. Vem, da se mi rjuho splača obesiti čez več vrvi, bržkone pa časa sušenja ne morem poljubno zmanjšati. No, takole so mi povedali: rjuha, obešena čez $1 \leq k \leq m$ vrvi, se bo posušila v času $1/k$, če jo obesim čez najmanj m vrvi, pa bo suha po času $1/m$.

Najmanj koliko časa se bo sušila žehta?

Aha, še to. Ni mi všeč, če sta dve rjuhi *istočasno* obešeni čez različno število vrvi. Grdo izgleda.

Vhod

Vhod je sestavljen iz ene same vrstice, ta pa vsebuje števila v, r in m.

Omejitve

- $1 \leq v \leq 10^6$.
- $1 \leq r \leq 10^6$.
- $1 \leq m \leq 20$.

Podnaloge

1. podnaloga (3 točke): $m = 1$.
2. podnaloga (5 točk): $v = 2$.
3. podnaloga (7 točk): $v \leq 10$, $r \leq 10$, $m \leq 5$.
4. podnaloga (11 točk): $v \leq 10$, $r \leq 10$.
5. podnaloga (19 točk): $v \leq 10^3$, $r \leq 10^3$.
6. podnaloga (23 točk): $v \leq 10^5$, $r \leq 10^5$.
7. podnaloga (32 točk): Ni dodatnih omejitev.

Izhod

Izpiši čas sušenja v obliki a/b, kjer je $\gcd(a, b) = 1$.

Primer 1

Vhod

```
10 8 3
```

Izhod

```
5/6
```

Komentar

Najprej vzamem tri rjuhe in vsako od njih razvlečem čez tri vrvi. Po času 1/3 bodo suhe. Nato obesim še preostalih pet rjuh, pri čemer vsako razvlečem čez dve vrvi. Te bodo suhe po času 1/2. Skupni čas sušenja tako znaša 5/6. To je optimum; ostale možnosti so kvečjemu slabše.

Primer 2

Vhod

```
10 9 3
```

Izhod

```
1/1
```

Komentar

Tokrat lahko obesim vseh devet rjuh hkrati (vsako obesim čez eno vrv) ali pa trikrat po tri rjuhe (vsako obesim čez tri vrvi) ali pa celo enkrat po pet rjuh in enkrat po štiri rjuhe (obakrat vsako obesim čez dve vrvi). V vseh treh primerih se bo žehta posušila v času 1.

Vijačniki

Zaporedoma bomo privili n vijakov, ki pripadajo tipom 0, 1, . . ., $m-1$ (npr. 0 = križni, 1 = ploščati, 2 = imbus . . .). Za vsakega od m tipov imamo po k vijačnikov. Vijak privijemo v t_v časovnih enotah, vijačnik pa se mora nato hladiti še t_h časovnih enot, preden je na voljo za naslednje vijačenje. Napiši program, ki izpiše porabo časa do trenutka, ko privijemo zadnji vijak.

Vhod

V prvi vrstici vhoda so zapisana števila n, m, k, t_v in t_h, druga pa vsebuje n števil z intervala $[0, m-1]$ (tipi posameznih vijakov).

Omejitve

- $1 \leq n \leq 10^6$.
- $1 \leq mk \leq 10^6$.
- $1 \leq t_v \leq 10^6$.
- $0 \leq t_h \leq 10^6$.

Podnaloge

1. podnaloga (7 točk): $m = 1$, $k = 1$.
2. podnaloga (12 točk): $k = 1$.
3. podnaloga (17 točk): $k \leq 100$.
4. podnaloga (30 točk): $m = 1$.
5. podnaloga (34 točk): Ni dodatnih omejitev.

Primer

Vhod

```
8 2 2 1 5
1 1 1 0 0 0 0 1
```

Izhod

```
16
```

Komentar

Prvega vijaka se z enim od vijačnikom tipa 1 lotimo ob času 0 in ga privijemo ob času 1. Drugega vijaka se z drugim vijačnikom tipa 1 lotimo ob času 1 in ga privijemo ob času 2. Sedaj se oba vijačnika tipa 1 hladita; eden se bo ohladil ob času 6, drugi pa ob času 7. Tretjega vijaka se bomo tako lahko lotili šele ob času 6, privili pa ga bomo ob času 7. Četrtega vijaka se bomo lotili takoj ob času 7, saj sta oba vijačnika tipa 0 pripravljena na uporabo. Petega vijaka se lotimo ob času 8, šestega ob času 13, sedmega ob času 14, osmega pa ob času 15 (privijemo ga ob času 16).

Ljubljanske trole

Ljubljanski potniški promet želi vzpostaviti trolne linije med postajami v mestu. Mesto je sestavljeno iz več postaj, ki so med seboj povezane s cestami različnih dolžin. Po vsaki cesti lahko potujemo v obe smeri.

Ko postavimo trolno linijo, izberemo zaporedje postaj, ki so med seboj povezane s cestami, in trola se bo ustavila na vsaki postaji linije. To pomeni, da se lahko s katere koli postaje na tej liniji pripeljemo do katere koli druge postaje, ki je tudi na isti liniji.

Cilj je vzpostaviti čim manjše število linij, tako da je mogoče doseči katero koli postajo s katere koli druge postaje, bodisi neposredno bodisi s prestopanji. Poleg tega želimo, da je skupna dolžina linij čim manjša.

Napiši program, ki izračuna najmanjše število linij, potrebnih za povezavo vseh postaj, nato pa izpiše še najmanjšo vsoto dolžin teh linij. Opomba: pri izbiri linij vedno upoštevamo le tiste postavitve, ki uporabljajo minimalno število linij.

Vhod

V prvi vrstici se nahaja T.

Za tem se v vhodu nahaja T testnih primerov. Vsak testni primer je oblike: * V prvi vrstici testnega primera se nahaja N, tj. število postaj. Postaje so oštevilčene od 1 do N. * Sledi $N-1$ vrstic oblike: a_i b_i w_i, ki predstavlja cesto med a_i in b_i z dolžino w_i.

Izhod

Za vsak testni primer izpiši dve števili: najprej najmanjše število linij, potem pa še najmanjšo skupno vsoto.

Omejitve

- $1 \leq T \leq 10^4$
- $3 \leq N \leq 2 \cdot 10^5$
- Če vsoto vseh N skozi testne primere označimo s S_N, potem $S_N \leq 2 \cdot 10^5$.
- $1 \leq a_i, b_i \leq N$
- $1 \leq w_i \leq 10^9$
- Med poljubnima postajama bo vedno natanko ena pot.

Podnaloge:

- (11 točk) $w_i = 0$
- (4 točke) Za natanko eno postajo velja, da gre z nje več kot ena cesta.
- (14 točk) $N \leq 8$ in $S_N \leq 500$
- (21 točk) $S_N \leq 500$
- (12 točk) $S_N \leq 5000$
- (38 točk) Brez dodatnih omejitev.

Primer

Vhod

```
3
3
3 2 1
3 1 2
10
10 7 2
5 2 4
4 2 1
9 5 3
7 6 2
```

```
10 1 1
8 6 5
3 1 3
6 2 7
8
4 3 1
5 2 4
4 1 2
7 3 1
8 1 5
6 3 2
3 2 5
```

Izhod

```
1 3
2 35
2 20
```