Racecar

Нане се заувек повукао из такмичарског програмирања. Након што је доста времена размишљао о томе чиме би следеће могао да се бави, одлучио је да постане возач Формуле 1.

Његова прва трка се одржава на веома досадној кружној стази дужине d метара, а на стази се налази n возача, укључујући и њега. Трка ће трајати t секунди.

Међутим, трка је прилично неправедна. Сваки аутомобил почиње са неке позиције на кружној стази и креће се константном брзином, израженом у метрима у секунди, при чему никоја два аутомобила немају исту брзину.

Формално, аутомобил i почиње на позицији p_i и креће се брзином s_i, при чему за свака два аутомобила $i \neq j$ важи $s_i \neq s_j$.

Стаза је прилично уска, па није лако пратити претицања између аутомобила. Нане жели да зна колико ће се укупно пута догодити да један аутомобил претекне други, али пошто се повукао из информатике, потребна му је твоја помоћ да то израчуна!

Претицање између два аутомобила дефинишемо као ситуацију у којој се они у неком тренутку налазе на истој позицији (тај тренутак не мора бити цео број секунди). Такође, могуће је да један аутомобил у истом тренутку претекне више других аутомобила, а ако један аутомобил сустигне други тачно на крају трке, и то се рачуна као претицање.

Улаз

Улаз се састоји од $n + 1$ редова.

У првом реду налазе се, редом, цели бројеви n, d и t.

У $(i + 1)$-вом реду, за $1 \leq i \leq n$, налазе се цели бројеви p_i и s_i.

Излаз

Исписати један број: колико пута неки аутомобил претекне неки други аутомобил.

Напомена: одговор можда неће моћи да стане у 32-битни целобројни тип.

Ограничења

- $2 \leq n \leq 2 \cdot 10^5$
- $3 \leq d \leq 5 \cdot 10^5$
- $1 \leq t \leq 10^9$
- $1 \leq p_i, s_i < d$
- За свако $1 \leq i < j \leq n$ важи $s_i \neq s_j$.

Подзадаци

Подзадатак 1: $n = 2$ (7 поена)

Подзадатак 2: $t \leq 100, n \leq 10$ и $s_i \leq 10$ за све $1 \leq i \leq n$ (9 поена)

Подзадатак 3: $n \leq 2000$ (40 поена)

Подзадатак 4: За све $1 \leq i < n$ важи $p_i < p_{i+1}$ и $s_i < s_{i+1}$ (20 поена)

Подзадатак 5: Без додатних ограничења. (24 поена)

Тест примери

Улаз 1

```
2 5 5
1 1
1 2
```

Излаз 1

```
2
```

Објашњење

Аутомобил 2 престиже аутомобил 1 на самом почетку трке, а они се поново сусрећу и на самом крају трке.

Улаз 2

```
10 50 16
4 5
1 3
1 2
2 8
49 10
22 13
13 4
16 11
37 7
49 12
```

Излаз 2

```
70
```

Treemax

Рукфаш се, *иако тамо живи*, некако изгубио у Башчаршији! Потребна му је ваша помоћ да пронађе излаз.

Башчаршија се може представити као повезан ацикличан неусмерен граф са n чворова (стабло). Неки чворови су означени, док други нису. Гарантује се да су све уписане ознаке различити цели бројеви из опсега $[1, n]$.

У овој верзији Башчаршије (представљеној стаблом) постоји начин да се израчуна колико је лако да се изгубите. За сваки пар чворова посматра се највећа ознака чвора на путу између њих (укључујући и почетни и крајњи чвор), а затим се све те вредности саберу.

Формално, за два различита чвора u и v, дефинишимо $cost(u, v)$ као највећу ознаку чвора на путу између њих, укључујући оба крајња чвора. Дефинишимо *изгубљеност* као меру тога колико је лако изгубити се. Изгубљеност стабла је:

$$\sum_{1 \leq u < v \leq n} cost(u, v).$$

На Башчаршији тренутно постоје чворови на којима није уписана ознака. Да бисте помогли Рукфашу, потребно је да одредите да ли је могуће означити неозначене чворове тако да изгубљеност стабла буде најмања међу свих $n!$ могућих пермутација ознака.

Формално, нека је min најмања изгубљеност међу свих $n!$ пермутација ознака, без обзира на чворове са већ уписаним ознакама.

Потребно је одредити да ли се неозначеним чворовима могу доделити недостајуће ознаке, без мењања већ уписаних ознака, тако да све ознаке чине пермутацију бројева од 1 до n, а изгубљеност добијеног стабла буде једнака min.

Улаз

Први ред садржи T, број тест примера.

За сваки тест пример, у првом реду налази се број n.

У следећем реду налази се низ a од n целих бројева, где a_i представља ознаку (број) уписану на i-том чвору, односно 0 ако на том чвору ништа није уписано. Гарантује се да су све вредности (које нису нула) међусобно различите.

У наредних $n-1$ редова налазе се по два цела броја u, v, који представљају грану између чвора са *индексом* u и чвора са *индексом* v.

Излаз

Исписати T линија: у свакој од њих исписати YES **(обавезно великим словима!)** ако је могуће означити чворове тако да изгубљеност буде најмања могућа, а NO у супротном.

Ограничења

- $1 \leq T \leq 2 \cdot 10^5$
- $2 \leq n \leq 2 \cdot 10^5$
- $1 \leq u, v \leq n$
- $0 \leq a_i \leq n$
- За све $i < j$ такве да $a_i \neq 0$ и $a_j \neq 0$, важи $a_i \neq a_j$
- Нека је S збир вредности n за све тест примере. Важи $S \leq 2 \cdot 10^5$

Подзадаци

Подзадатак 1 (5 поена)

$S \leq 7$

Подзадатак 2 (12 поена)

$S \leq 20$

Подзадатак 3 (23 поена)

$S \leq 2000$

Подзадатак 4 (18 поена)

Бројем 1 је означен неки чвор.

Подзадатак 5 (10 поена)

Сваки чвор има највише 10 суседа.

Подзадатак 6 (32 поена)

Без додатних ограничења.

Примери

Улаз 1

```
2
3
3 1 2
1 3
1 2
3
0 1 0
1 2
2 3
```

Излаз 1

```
NO
YES
```

Објашњење 1

Означавањем из првог тест примера добија се изгубљеност 9, док је најмања изгубљеност међу свим пермутацијама ознака једнака 8.

Једно могуће означавање за други тест пример јесте $a = [2, 1, 3]$. Изгубљеност при овом означавању је 8. Може се показати да није могуће постићи мању изгубљеност.

Улаз 2

```
1
15
4 0 0 0 6 0 0 0 14 0 0 0 0 0 11
1 2
2 3
2 4
2 5
1 6
6 7
7 8
8 9
1 10
10 11
11 12
12 13
13 14
14 15
```

Излаз 2

```
YES
```

Улаз 3

```
2
12
1 0 4 0 8 0 7 0 9 0 11 0
1 2
2 3
3 4
4 5
3 6
6 7
1 8
8 9
9 10
10 11
9 12
11
4 0 0 0 0 0 6 0 0 0 8
1 2
2 3
3 4
4 5
1 6
6 7
1 8
8 9
9 10
10 11
```

Излаз 3

```
YES
YES
```

Sketch show

Фарук већ месецима припрема своју скеч представу, али још увек није пронашао место за њено одржавање. Жели да сви у Сарајеву погледају представу, па је одлучио да је одржи на планини Требевић.

Фарук не жели да место одржавања буде ни превисоко ни прениско: жели да буде на надморској висини тачно x. Планина је представљена матрицом надморских висина, означеном са A, димензија $N \times N$. Свака врста и свака колона матрице A је неопадајућа.

Да би пронашао локацију, може да пита туристичког водича за надморску висину у тачки (i, j). Пошто пати од социјалне анксиозности, жели да сведе на минимум број питања која поставља водичу.

Овај задатак има два независна подзадатка који вреде 30 и 70 поена. Потребно је предати једну датотеку, `sketch.cpp`, која имплементира функције `findVenue`, `init` и `query`. И делимично (парцијално) решење мора садржати све три функције како би могло да се компајлира.

Подзадаци

Подзадатак 1 (30 поена)

Фарук жели да сазна да ли се надморска висина x појављује у матрици A. На почетку не зна ниједну вредност у матрици, али може да позива:

```
int ask(int i, int j);
```

Ова функција враћа $A[i][j]$. Индекси морају задовољавати $0 \leq i, j < N$.

Имплементирајте:

```
bool findVenue(int N, int x);
```

Функција треба да врати `true` ако и само ако је вредност неког поља матрице једнака x.

Ограничења

- $2 \leq N \leq 100$
- $1 \leq x \leq 10^9$
- $1 \leq A[i][j] \leq 10^9$
- $A[i][j] \leq A[i][j+1]$ за све $0 \leq i < N, 0 \leq j < N-1$.
- $A[i][j] \leq A[i+1][j]$ за све $0 \leq i < N-1, 0 \leq j < N$.
- Функцију `ask` можете позвати највише 10^4 пута.

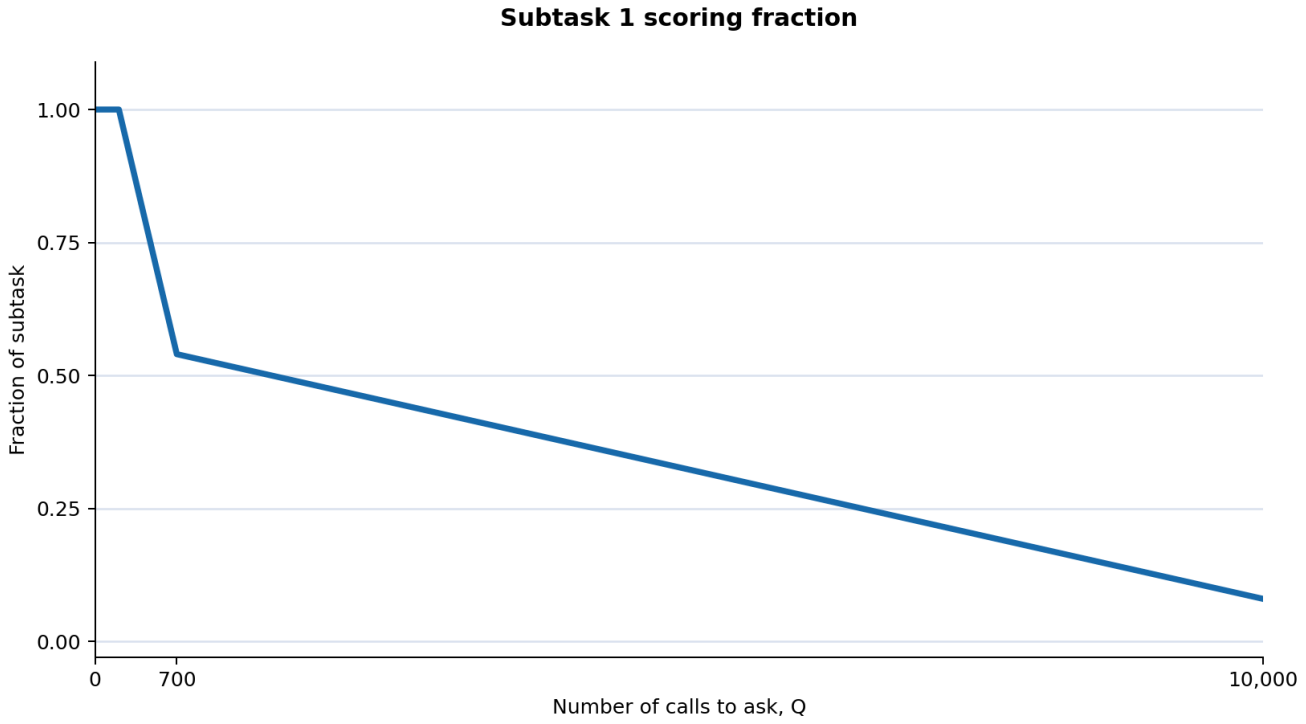
Бодовање

Нека је Q највећи број позива функције `ask` међу свим тест примерима. Ако `findVenue` врати нетачну вредност на било ком тест примеру или се изврши недозвољен позив функције `ask`, добијате 0 поена за овај подзадатак.

У супротном, број поена рачуна се као $S_1(Q) \cdot 30$, где је функција S_1 дефинисана на следећи начин:

- $S_1(Q) = 0$ ако је $Q > 10^4$.
- $S_1(Q) = \frac{27 - \frac{23(Q-700)}{9300}}{50}$ ако је $700 < Q \leq 10^4$.
- $S_1(Q) = \frac{50 - \frac{23(Q-204)}{496}}{50}$ ако је $204 < Q \leq 700$.
- $S_1(Q) = 1$ ако је $Q \leq 204$.

Имајте у виду да је грејдер адаптиван.



Подзадатак 2 (70 поена)

У овом подзадатку имплементирате туристичког водича. Водич ужива у разговору са Фаруком и жели да разговор траје што дуже. Да га не би разочарао, водич мора да обезбеди да матрица увек садржи надморску висину x; Фарук престаје да поставља питања тек када је пронађе.

Водич може да бира одговоре о надморским висинама у зависности од дотадашњег тока разговора, уместо да унапред одреди целу матрицу. Међутим, после сваког одговора мора постојати бар једна матрица са неопадајућим врстама и колонама која садржи x и која је у складу са свим дотадашњим одговорима.

Детаљи имплементације

Имплементирајте:

```
void init(int N, int x);
int query(int i, int j);
```

Грејдер позива `init(N, x)` на почетку сваке интеракције. Може позвати `init` више пута током једног тест примера. Сваки позив започиње нову интеракцију. Након тога, `query(i, j)` треба да врати водичев одговор за поље (i, j).

У оквиру једне интеракције, водичеви одговори морају задовољавати сва следећа правила:

- Поновљени упити за исто поље добијају исти одговор.
- Сваки одговор је цео број у интервалу $[1, 10^9]$.
- Сви одговори су у складу са неопадајућим поретком врста и колона.
- Постоји исправна матрица која је у складу са свим одговорима и која садржи x.

Ограничења

- $2 \leq N \leq 10$
- $2 \leq x \leq 10^9 - 1$

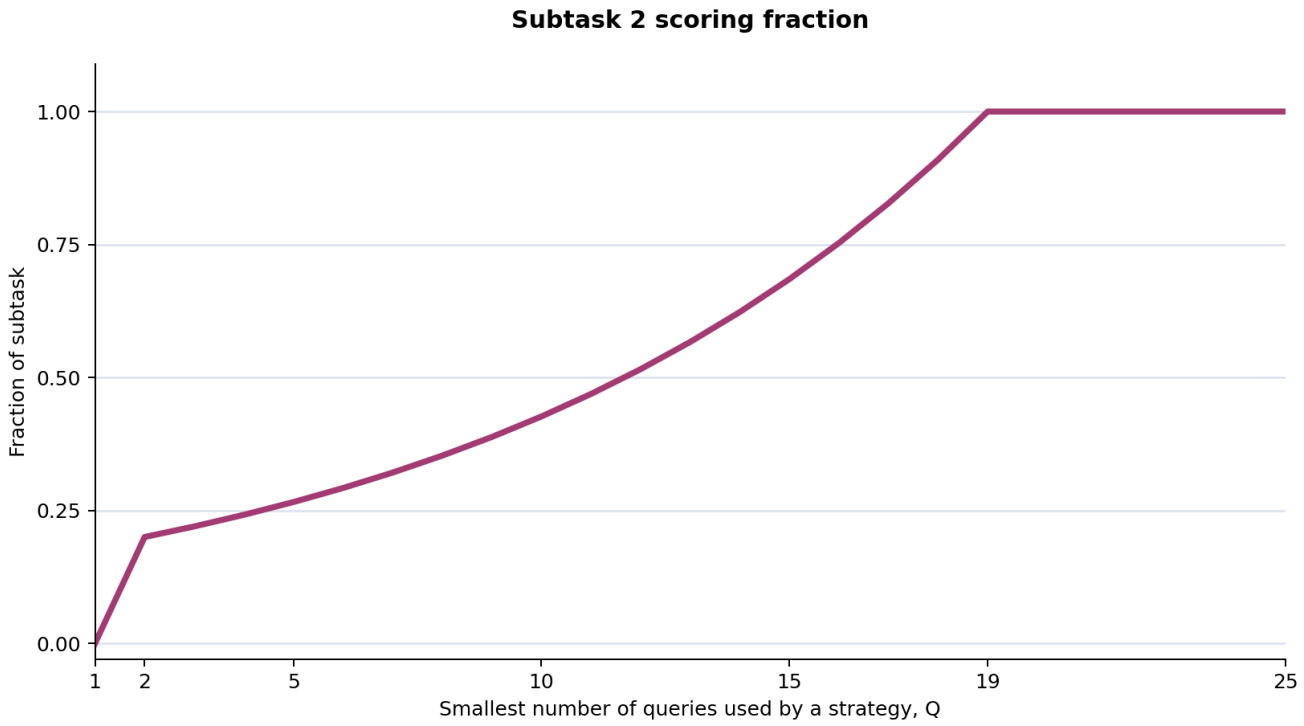
Бодовање

Грејдер покреће неколико Фарукових стратегија. Стратегије могу бити адаптивне. Свако покретање стратегије представља једну интеракцију. Ако се током било које интеракције прекрши неко од наведених правила, добијате 0 поена за овај подзадатак. У супротном, нека је Q најмањи број позива функције `query` међу свим интеракцијама.

Број поена рачуна се као $S_2(Q) \cdot 70$, где је функција S_2 дефинисана на следећи начин:

- $S_2(Q) = 0$ ако је $Q = 1$.

- $S_2(Q) = \min\left(1, \frac{1}{5}e^{\frac{\ln 5}{17}(Q-2)}\right)$ ако је $Q \geq 2$.



Примери

Подзадатак 1

Претпоставимо да је $N = 4$, $x = 7$ и да је скривена матрица:

$$\begin{matrix} 1 & 3 & 5 & 8 \\ 2 & 4 & 7 & 9 \\ 3 & 6 & 10 & 12 \\ 5 & 8 & 11 & 15 \end{matrix}$$

Једна могућа интеракција је:

Позив функције	Повратна вредност	Објашњење
`ask(2, 0)`	`3`	Ова вредност је мања од x.
`ask(0, 3)`	`8`	Ова вредност је већа од x.
`ask(1, 2)`	`7`	Пронађена је тражена надморска висина.

Дакле, `findVenue(4, 7)` враћа `true`. За исту матрицу и $x = 13$, функција мора да врати `false`.

Подзадатак 2

Поново претпоставимо да је $N = 4$ и $x = 7$. Водич би могао да бира одговоре који су у складу са следећом матрицом:

$$\begin{matrix} 1 & 3 & 5 & 8 \\ 2 & 4 & 7 & 9 \\ 3 & 6 & 10 & 12 \\ 5 & 8 & 11 & 15 \end{matrix}$$

На пример, водич може да одговара на следећи начин:

Фарук позива	**Водич враћа**
`query(2, 0)`	3
`query(0, 3)`	8
`query(1, 2)`	7

Нека друга стратегија могла би одмах да пронађе x:

Фарук позива	**Водич враћа**
`query(1, 2)`	7

Ова стратегија проналази x након једног упита. Број поена за ову интеракцију био би 0, јер је $S_2(1) = 0$.

Предаја решења

Почните од приложене датотеке `att/sketch.cpp`. Она садржи празне дефиниције све три функције. Допуните их и задржите следећу линију на почетку датотеке:

```
#include "grader.h"
```

У свом решењу немојте дефинисати функцију `main`, читати са стандардног улаза нити писати на стандардни излаз.

Програм за локално тестирање

Директоријум `att` садржи почетну датотеку `sketch.cpp` и програм за локално тестирање. Копирајте своје решење у `sketch.cpp` или измените ту датотеку, па из тог директоријума покрените компајлирање:

```
g++ -std=c++20 -o sketch sketch.cpp grader.cpp
```

Програм за локално тестирање има два режима рада. Прво учитава `N x`, а затим проверава следећи улазни податак:

- Ако је то број, покреће се режим за подзадатак 1, јер тај број представља прву вредност матрице.
- Ако је то `?` или `!`, покреће се режим за подзадатак 2, јер су то команде за интеракцију са водичем.

Режим за подзадатак 1

Унесите `N x`, а затим свих N^2 вредности матрице. Програм за локално тестирање једном позива `findVenue(N, x)`.

```
4 7
1 3 5 8
2 4 7 9
3 6 10 12
5 8 11 15
```

За тачан одговор исписује:

```
1
Q
```

где је Q број позива функције `ask`. У супротном исписује `0` и објашњење.

Режим за подзадатак 2

Прво унесите `N x`. Затим уносите команде, сваку у посебном реду:

- `? i j` позива `query(i, j)` и исписује враћену надморску висину.
- `!` завршава интеракцију. Унесите ову команду тек након што једна од исписаних надморских висина буде једнака x.

На пример, започните интеракцију овако:

```
4 7
? 0 0
```

Наставите да уносите команде `? i j` по жељи. Када програм за локално тестирање испише x, унесите `!`.

Програм за локално тестирање одмах проверава сваки одговор. Након успешног завршетка исписује:

```
1
Q
P
```

где је Q број упита, а $P = S_2(Q)$ удео поена освојен за ту интеракцију, изражен бројем из интервала $[0, 1]$. Исписује `0` и објашњење ако је неки одговор неисправан или ако се `!` унесе пре него што је пронађено x.

Team Parade

На церемонији отварања JBOI, N такмичара стоји у једном дугачком реду, на позицијама $1, 2, \ldots, N$. Сваки такмичар припада тачно једном од K тимова, нумерисаних од 1 до K, и сваком тиму припада бар један такмичар.

Организатори желе да формирају свечану параду тако што ће изабрати тачно једног такмичара из сваког тима. Изабрани такмичари морају се у реду појављивати по редоследу бројева својих тимова. Другим речима, организатори морају да изаберу такмичара из тима 1, затим негде десно од њега такмичара из тима 2, затим негде десно од претходно изабраног такмичара из тима 3, и тако даље, све до тима K.

За два узастопно изабрана такмичара i и $i+1$, који се налазе на позицијама p_i и p_{i+1}, редом, њихово растојање дефинишемо као $p_{i+1} - p_i$.

Нека је D највеће растојање између два узастопно изабрана такмичара у паради. Изаберите такмичаре тако да вредност D буде што мања и испишите ту вредност.

Ако није могуће изабрати по једног такмичара из сваког тима у захтеваном редоследу, испишите -1.

Улаз

У првом реду налазе се два цела броја N и K.

У другом реду налази се N целих бројева $a_1, a_2, \ldots, a_N$, где је a_i број тима ком припада i-ти такмичар у реду.

Излаз

Испишите један цео број: најмању могућу вредност D, или -1 ако није могуће формирати параду.

Ограничења

- $2 \leq K \leq N \leq 2 \cdot 10^5$
- $1 \leq a_i \leq K$

- Гарантује се да сваки тим има барем једног такмичара.

Подзадаци

Подзадатак 1 (4 поена)

$N = K$

Подзадатак 2 (6 поена)

$K = 2$

Подзадатак 3 (11 поена)

$K = 3$

Подзадатак 4 (15 поена)

Низ a је сортиран у неопадајућем поретку.

Подзадатак 5 (28 поена)

$N \leq 2000$

Подзадатак 6 (17 поена)

Сваки тим има највише 50 такмичара.

Подзадатак 7 (19 поена)

Без додатних ограничења.

Примери

Улаз 1

```
12 4
2 1 3 1 2 2 3 4 2 3 4 4
```

Излаз 1

```
2
```

Објашњење 1

Један могући избор су такмичари на позицијама $4, 6, 7, 8$. Они припадају тимовима $1, 2, 3, 4$, редом, а растојања између узастопно изабраних такмичара су $2, 1, 1$. Дакле, $D = 2$.

Није могуће постићи $D = 1$, па је одговор 2.

Улаз 2

```
4 3
3 2 1 3
```

Излаз 2

```
-1
```

Објашњење 2

Једини такмичар из тима 2 налази се на позицији 2, лево од јединог такмичара из тима 1, који се налази на позицији 3, па не постоји исправна парада.

Lines

Након што је на Фејсбуку прочитао да је Claude Code окончао програмирање, Фуки је одлучио да почне да се бави физиком.

Он мисли да је открио нову физичку формулу којом се одређује колико килограма бурека ће појести, у зависности од времена протеклог од његовог последњег оброка. Сматра да је формула облика $y = m \cdot x + b$, где је y маса бурека коју ће појести, а x време протекло од његовог последњег оброка. Константе m и b представљају, редом, нагиб праве и y -координату њеног пресека са y-осом.

Прикупио је n мерења како би проверио своје револуционарно откриће. Мерења су дата тројкама (x_i, y_i, d_i), где је x_i време протекло од његовог последњег оброка, y_i маса поједеног бурека коју је показала вага, а d_i непрецизност ваге којом је мерио бурек. Према томе, стварна маса бурека коју је појео припада интервалу $[y_i - d_i, y_i + d_i]$, а тачна маса му није позната. За потребе овог задатка, m може бити негативно, а y_i може бити мање од d_i.

Мерење i можемо представити као вертикалну дуж чије су крајње тачке $(x_i, y_i - d_i)$ и $(x_i, y_i + d_i)$.

Фуки сада жели да зна да ли постоји права која није вертикална и која пролази кроз све ове дужи. Другим речима, права мора имати заједничку тачку са сваком дужи.

Ако таква права не постоји, пријавите да не постоји. У супротном, испишите најмањи и највећи могући нагиб такве праве.

Имајте у виду да праве са најмањим и највећим нагибом не морају имати исту тачку пресека са y-осом.

Улаз

У првом реду налази се позитиван цео број n.

Затим следи n редова, од којих сваки садржи по један цео број x_i и два децимална броја y_i и d_i. Бројеви y_i и d_i имају највише девет цифара после децималног зареза.

Гарантује се да су све вредности x_i међусобно различите.

Излаз

Ако не постоји ниједна одговарајућа права, испишите -1. У супротном, испишите два реална броја: најмањи и највећи могући нагиб.

За оба броја дозвољена је апсолутна или релативна грешка највише 10^{-6}. Формално, ако је ваш исписани број x', а очекивани број x, ваш одговор се сматра тачним ако важи $\frac{|x-x'|}{\max(1,|x|)} \leq 10^{-6}$.

Савети за имплементацију

За прецизније израчунавање користите тип `long double` у језику C++. Такође, при исписивању бројева у покретном зарезу обавезно задајте прецизност. Следи пример кода:

```
long double val = 10;
std::cout << std::fixed << std::setprecision(9) << val << "\n";
```

При коришћењу `setprecision` обавезно укључите заглавље `<iomanip>`.

Ограничења

Гарантује се да, ако постоји одговарајућа права, разлика између највећег и најмањег могућег нагиба износи барем 0.1. Ако не постоји одговарајућа права, она не постоји ни када се свако d_i повећа за 10^{-3}. Такође се гарантује да за сваки могући нагиб m одговарајуће праве важи $|m| \leq 10^3$.

- $2 \leq n \leq 2 \cdot 10^5$
- $0 \leq x_i \leq 10^6$
- $0 \leq y_i \leq 10^9$
- $0 < d_i \leq 10^9$

Подзадаци

Подзадатак 1 (5 поена)

$n = 2$

Подзадатак 2 (7 поена)

Гарантује се да постоји права са нагибом 1 која пролази кроз све дужи.

Подзадатак 3 (19 поена)

$n \leq 100$

Подзадатак 4 (29 поена)

$n \leq 2000$

Подзадатак 5 (40 поена)

Без додатних ограничења.

Примери

Улаз 1

```
2
0 0 1
10 10 1
```

Излаз 1

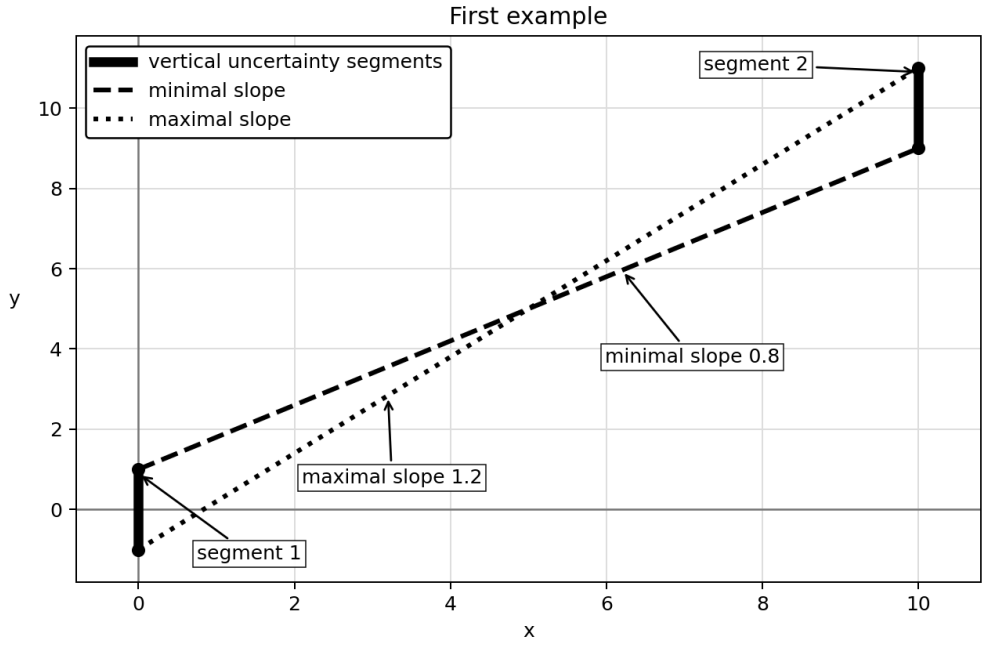
```
0.800000000000 1.200000000000
```

Објашњење 1

Прва дуж се протеже од $(0, -1)$ до $(0, 1)$, а друга од $(10, 9)$ до $(10, 11)$.

Најмањи могући нагиб добија се повезивањем тачака $(0, 1)$ и $(10, 9)$ и износи 0.8. Највећи могући нагиб добија се повезивањем тачака $(0, -1)$ и $(10, 11)$ и износи 1.2.

Илустрација првог примера:



Улаз 2

```
3
0 0 0.7
2 0 1.5
1 99.3 1
```

Излаз 2

```
-1
```

Објашњење 2

Ниједна невертикална права не може да пресече све три дужи, јер је средња дуж превисоко у односу на две крајње дужи.

Christmas Tree

Добро дошли на свемирски брод The Circular! Посада украшава његову контролну таблу: мрежу сијалица димензија $N \times M$, у којој свака сијалица има целобројну јачину светлости.

Мрежа је дата матрицом T. Вредност $T_{i,j}$ представља јачину светлости сијалице у реду (врсти) i и колони j. Редови су нумерисани од 1 до N одозго надоле, а колоне од 1 до M слева надесно.

Новогодишња јелка висине H ($H > 1$) одређује се избором поља (i, j) које представља звезду јелке. Јелка заузима тачно H редова: $i, i+1, \ldots, i+H-1$. У реду $i+d$ ($0 \leq d < H$), јелка садржи сва поља у колонама $j-d, j-d+1, \ldots, j+d$.

Јелка мора у потпуности да се налази унутар мреже. Такође, за свака два узастопна реда јелке, свака јачина светлости у горњем реду јелке мора бити строго мања од сваке јачине светлости у доњем реду јелке.

Формално, звезда јелке висине H мора се налазити на пољу (i, j) таквом да важи $1 \leq i \leq N-H+1$ и $H \leq j \leq M-H+1$.

За свако $d = 0, 1, \ldots, H-1$, нека је

$$S_d = \{T_{i+d,j-d}, T_{i+d,j-d+1}, \ldots, T_{i+d,j+d}\}.$$

Тада за свако $r = 0, 1, \ldots, H-2$, свако $x \in S_r$ и свако $y \in S_{r+1}$ мора да важи $x < y$.

За сваку висину k од 2 до N, одредите највећу могућу вредност звезде $T_{i,j}$ такву да постоји исправна новогодишња јелка висине тачно k чија је звезда на пољу (i, j), или испишите -1 ако таква јелка не постоји.

Улаз

У првом реду налазе се два цела броја N и M.

У наредних N редова налази се по M целих бројева који описују матрицу T.

Напомена: улаз може бити велики, па се препоручује коришћење следећих наредби:

```
ios::sync_with_stdio(false);
cin.tie(0);
```

Излаз

Испишите $N-1$ редова.

У реду који одговара вредности $k = 2, 3, \ldots, N$, испишите одговор за висину k: највећу вредност звезде међу свим исправним новогодишњим јелкама висине тачно k, или -1 ако таква јелка не постоји.

Ограничења

- $2 \leq N \leq 2500$
- $1 \leq M \leq 2500$
- $1 \leq T_{i,j} \leq 10^9$

Подзадаци

Подзадатак 1 (3 поена)

$M < 3$

Подзадатак 2 (5 поена)

$N = 3$

Подзадатак 3 (12 поена)

$N, M \leq 50$

Подзадатак 4 (15 поена)

$N, M \leq 200$

Подзадатак 5 (19 поена)

$N, M \leq 600$

Подзадатак 6 (38 поена)

$N, M \leq 900$

Подзадатак 7 (8 поена)

Без додатних ограничења.

Примери

Улаз 1

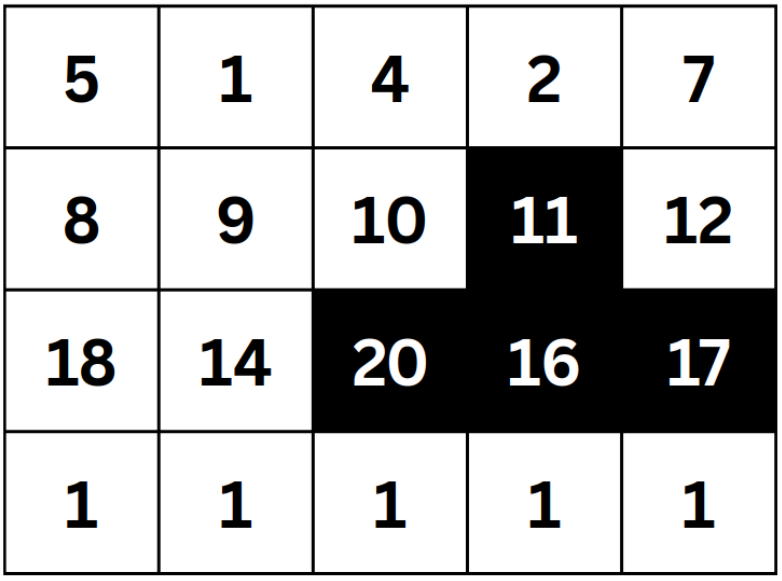
```
4 5
5 1 4 2 7
8 9 10 11 12
18 14 20 16 17
1 1 1 1 1
```

Излаз 1

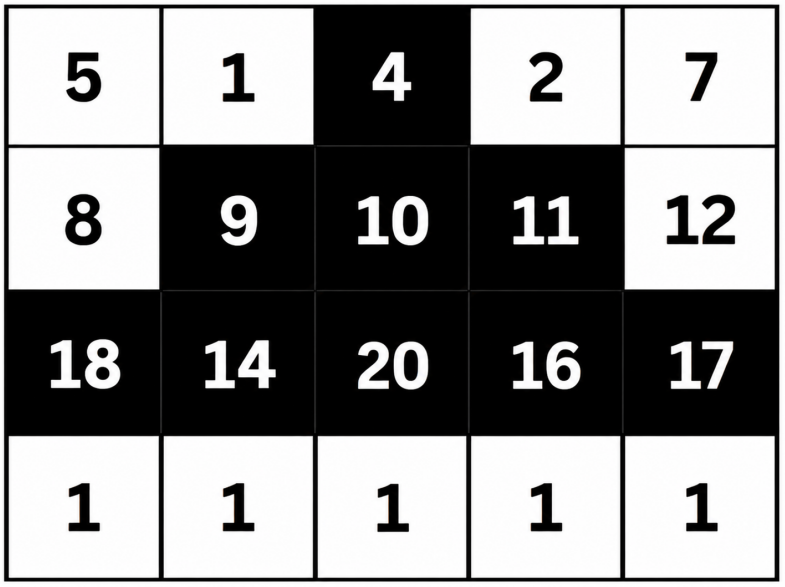
```
11
4
-1
```

Објашњење 1

За висину 2, најбоља могућа звезда је поље са јачином светлости 11 у другом реду и четвртој колони. Ред јелке испод њега садржи јачине светлости 20, 16 и 17, које су све веће. Поље са јачином светлости 12 не може бити звезда, јер јелка тада не би у целости стала у мрежу.



За висину 3, најбоља могућа звезда има јачину светлости 4:



Не постоји исправна јелка висине 4.