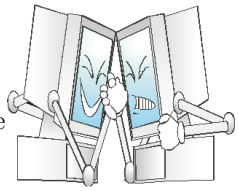


Проблем 1. Рачуни

Ђурица воли да обилази ресторане. Током године је обишао пуно ресторана, и сад је баш огладнео и занима га у који следећи да оде. У свим ресторанима које је Ђурица посетио укупно је наручио k различитих ставки, те се сваки рачун може описати као

$$id\ s_1\ s_2\ \ldots\ s_k.$$

Вредност id је ознака ресторана. Свака ознака јединствено одређује један ресторан, и сваки ресторан је јединствено одређен једном ознаком. Вредност s_i је 0 ако на том рачуну не постоји i-та ставка, иначе је природан број који представља колико је плаћена i-та ставка. Ђурица је сакупио тачно n рачуна, и њега занима збирни рачун за сваки ресторан. Имајући све рачуне за ресторане, збирни рачун за ресторан id се дефинише као рачун са k ставки где вредност за i-ту ставку представља суму цена i-те ставке на свим рачунима ресторана id.

Једном кад Ђурица има све збирне рачуне, он сматра да је најбољи ресторан у који следећи треба отићи онај који има лексикографски најмањи збирни рачун ставки. Ђурица је гладан и не може да размишља, те помозите Ђурици и за дати списак рачуна испишите лекикографски најмањи рачун.

Током одређивања и исписивања лексикографски најмањег збирног рачуна не треба узимати у обзир id ресторана. За два збирна рачуна $a_1\ a_2\ \ldots\ a_k$ и $b_1\ b_2\ \ldots\ b_k$ кажемо да је први рачун лексикографски мањи од другог ако постоји j такво да $a_i = b_i, i < j$, и $a_j < b_j$.

Улаз. (Улазни подаци се учитавају из датотеке `racuni.in`.) У првом реду се налазе два природна броја n и k ($1 \leqslant n \leqslant 30.000$, $1 \leqslant k \leqslant 30$) У наредних n редова су дати описи рачуна у формату

$$id\ s_1\ s_2\ \ldots\ s_k.$$

Одговарајућа ограничења су: $1 \leqslant id \leqslant 1.000.000.000$, $0 \leqslant s_j \leqslant 100.000$.

Излаз. (Излазни подаци се исписују у датотеку `racuni.out`.) У првом и једином реду исписати k бројева који представљају лексикографски најмањи збирни рачун.

Пример 1.

`racuni.in`
```
4 2
1 3 4
2 5 1
1 0 8
7 3 20
```

`racuni.out`
```
3 12
```

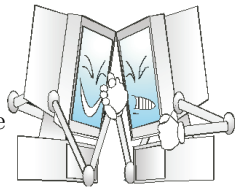
Објашњење. Збирни рачуни за ресторане 1, 2 и 7 су редом 3 12, 5 1 и 3 20.

Пример 2.

`racuni.in`
```
7 4
11 303 0 0 0
2 10 2 8 0
13 20 3 8 1
11 9 4 0 0
2 9 4 0 1
2 3 0 0 5
13 2 3 0 0
```

`racuni.out`
```
22 6 8 1
```

Објашњење. Ресторан са бројем 13 има лексикографски најмањи збирни рачун.



Проблем 2. Сумхем

У теорији информација, Хемингово растојање (енг. *Hamming distance*) између два стринга једнаких дужина је број позиција на којима се одговарајући симболи разликују. Другим речима, ово растојање мери минимални број замена потребних за трансформацију једног стринга у други.

У задатку ћемо посматрати 32-битне бројеве, где се Хемингово растојање $\mathrm{ham}(a, b)$ рачуна као број битова на којима се бројеви a и b разликују. На пример, Хемингово растојање за бројеве $93 = 00...0010\mathbf{11}\mathbf{10}1$ и $75 = 00...0010\mathbf{01}\mathbf{01}1$ је 3, пошто се бројеви разликују на другом, трећем и петом биту с десна.

Дат је низ целих бројева a дужине n. Одредити збир Хемингових растојања свих парова елемената низа, односно

$$\sum_{i<j} \mathrm{ham}(a[i], a[j]).$$

Улаз. (Улазни подаци се учитавају из датотеке `sumhem.in`.) У првом реду се налази природан број n ($1 \leqslant n \leqslant 100.000$). У следећих n редова се налази по један број и они представљају низ a ($0 \leqslant a[i] \leqslant 2^{31} - 1$).

Излаз. (Излазни подаци се исписују у датотеку `sumhem.out`.) У првом и једином реду исписати суму Хемингових растојања свака два броја у низу.

Пример 1.

`sumhem.in`	`sumhem.out`

`4`
`1`
`9`
`4`
`10`
	`14`

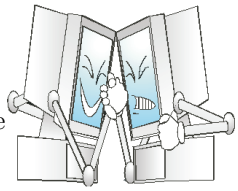
Објашњење. Хемингова растојања за сваки пар бројева из низа су једнака $ham(1, 9) = 1$, $ham(1, 4) = 2$, $ham(1, 10) = 3$, $ham(9, 4) = 3$, $ham(9, 10) = 2$ и $ham(4, 10) = 3$. Сума свих растојања је управо 14.

Пример 2.

`sumhem.in`	`sumhem.out`

`7`
`128`
`7`
`0`
`99`
`18`
`10`
`1984`
	`84`

Напомена. У 40% тест примера, број n ће бити мањи од 3.000.



Проблем 3. Папирићи

Перица је одувек волео игре на срећу, па је смислио једну за свог друга Јовицу. Перица узме n папирића, на сваком папирићу напише по један број и окрене их тако да Јовица не зна који су бројеви написани. Затим Јовица мора да распореди све папириће на произвољан број гомила. Свака гомила мора да има најмање 2, а највише k папирића. На крају Перица открије све папириће, те рачуна број поена који је Јовица освојио.

Перица рачуна поене на следећи начин: Пронађе највећи и најмањи број на једној гомили и израчуна њихову разлику. Када израчуна разлику највећег и најмањег за сваку гомилу, сабере све разлике. Добијена вредност представља освојене поене. Што је број поена мањи, то је Јовица успешнији.

Знајући које бројеве је Перица написао на папириће, одредите колико најмање поена Јовица може да сакупи.

Улаз. (Улазни подаци се учитавају из датотеке `papirici.in`.) У првом реду улаза налазе се два броја n и k ($2 \leqslant k \leqslant n \leqslant 100.000$) који представљају, редом, број папирића и максималну величину гомила. У другом реду се налази n целих бројева, који представљају бројеве написане на папирићима. Сви бројеви ће бити у интервалу $[-2^{31}, 2^{31} - 1]$.

Излаз. (Излазни подаци се исписују у датотеку `papirici.out`.) У првом и једином реду излаза исписати најмањи број поена који може да се сакупи. Решење ће увек постојати.

Пример 1.

papirici.in	papirici.out
7 3 6 0 5 2 8 0 -2	7

Објашњење. На прву гомилу се ставе папирићи са бројевима 0 и 2, на другу гомилу папирићи са 6, 5 и 8, а на трећу папирићи са 0 и -2. Укупан број поена је $(2 - 0) + (8 - 5) + (0 - (-2)) = 7$

Напомена. У 30% тест примера биће $k \leqslant 100$.

Проблем 4. Мастило

Набавили смо једну од супер-упијајућих крпи са једног од *tv shop*-ова и желимо да је тестирамо. Крпа је правоугаоног облика и има шаре у облику вертикалних и хоризонталних линија које је деле на $n \times m$ подударних квадратића. Тест се састоји од проливања мастила на неке од $n \times m$ квадратића крпе (сваки квадратић је или упрљан мастилом или потпуно чист на почетку) и мерења времена за које ће крпа упити сво мастило.

Наравно, испоставља се да је крпа скупа и бескорисна. Осим што не упија мастило како треба, она доприноси његовом разливању. Прецизније, сваке секунде мастило се разлива на суседне, чисте квадратиће (тј. сваки чист квадратић који у тој секунди има **бар једног упрљаног суседа** постаје упрљан мастилом) а **у исто време** крпа упија оно што је до тада било упрљано мастилом (тј. **сваки** упрљани квадратић постаје чист). Два квадратића су суседна ако деле заједничку страницу.

Пошто смо разочарани крпом, а не желимо да нам експеримент пропадне, одлучили смо да предвидимо како ће крпа изгледати после t секунди.

Улаз. (Улазни подаци се учитавају из датотеке `mastilo.in`.) У првом реду улазне датотеке налазе се 3 природна броја n, m и t који представљају, редом, димензије крпе и време (у секундама) после ког нас интересује изглед крпе ($1 \leqslant n, m \leqslant 10^3$, $1 \leqslant t \leqslant 10^6$). Следећих n редова садрже низ од m цифара из скупа $\{0, 1\}$ **без размака** - изглед крпе након почетног проливања мастила. 1 означава да се на одговарајућем месту налази упрљан квадратић а 0 - чист квадратић.

Излаз. (Излазни подаци се исписују у датотеку `mastilo.out`.) У првих n редова излазне датотеке исписати по m цифара из скупа $\{0, 1\}$ без размака - изглед крпе након t секунди. 1 и 0 имају исто значење као на улазу.

Пример 1.

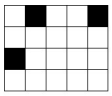
`mastilo.in`

```
4 5 2
01001
00000
10000
00000
```

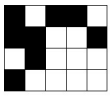
`mastilo.out`

```
01001
00110
10101
01000
```

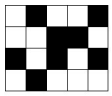
Објашњење. За дати тест пример, промена крпе је приказана на сликама:



Почетак



После 1 секунде



После 2 секунде

Напомена. У 25% тест примера биће $1 \leqslant n, m, t \leqslant 200$. Због величине улаза, `C/C++` кодови би требало да учитавају целе редове.

Проблем 5. Лето

Катарина обожава да путује и већ је почела да прави планове за ово лето. Направила је списак са n егзотичних места која би волела да посети. За свако место познато је колико дана би она тамо остала, то су вредности t_i, и при томе за свако $i \neq j$ важи $t_i \neq t_j$.

Катарина има p пријатеља са којима може да путује. Међутим, не желе сви пријатељи да иду на сва места, већ постоји p парова бројева b_i и e_i $(1 \leqslant b_i \leqslant e_i \leqslant n)$. Они нам говоре да i-ти пријатељ жели да иде само на места са редним бројем j таквим да важи $b_i \leqslant j \leqslant e_i$.

Потребно је пронаћи колико највише дана Катарина може да проведе на путовањима, ако морају да буду задовољени следећи услови:

- Катарина нигде не може да путује сама.
- Са сваким пријатељем она може да иде на највише једно место.
- Свако место она може посетити највише једном.

Улаз. (Улазни подаци се учитавају из датотеке `leto.in`.) У првом реду улазне датотеке се налази број n $(n \leqslant 5.000)$, број места. У следећем реду налази се n бројева раздвојених размацима, то су вредности t_i $(1 \leqslant t_i \leqslant 100.000)$. Следи ред у коме се налази број p $(p \leqslant 5.000)$, број пријатеља. У сваком од наредних p редова налазе се по два броја, b_i и e_i.

Излаз. (Излазни подаци се исписују у датотеку `leto.out`.) У излазну датотеку треба уписати само један број, колико највише дана могу трајати сва Катаринина путовања заједно.

Пример 1.

`leto.in`	`leto.out`

`8`
`2 3 1 4 6 5 20 9`
`5`
`5 6`
`2 5`
`1 2`
`8 8`
`8 8`
	`23`

Објашњење. Постоји 8 места, на првом се остаје 2 дана, на другом 3 дана, итд. Катарина има 5 пријатеља. Први пријатељ жели да иде на место 5 или 6, други на неко од места 2, 3, 4 и 5, итд. Оптимално решење добија се на пример ако са првим пријатељем оде на шесто место (5 дана), са другим на пето место (6 дана), са трећим на друго (3 дана), и са петим на осмо место (9 дана). То укупно даје $5 + 6 + 3 + 9 = 23$ дана.

Напомена. У 60% тест примера биће $n, p \leqslant 1.000$, а у 30% $n, p \leqslant 200$.