УПУТСТВО

Пред вама је 6 задатака чији је преглед дат у табели на следећој страни. **Ученици Б категорије решавају само прва 3 задатка а ученици А категорије само последња 3 задатка**. Уколико сте ученик Б категорије а желите да пређете у А категорију, **неопходно је да то наведете у коментарима кода који предајете и да о томе обавестите дежурног професора пре завршетка такмичења**. Израда задатака траје 5 сати.

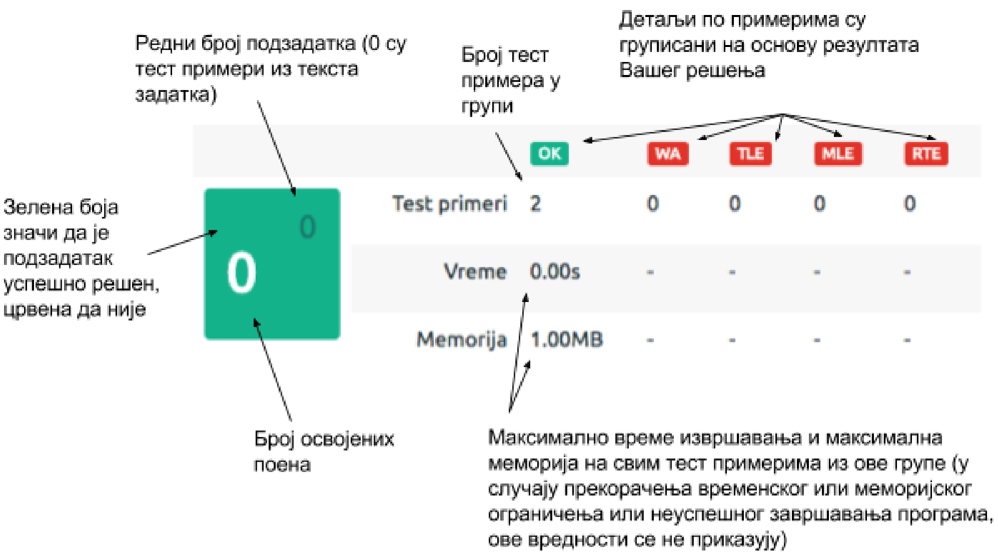
Задатке шаљете кроз такмичарско окружење коме приступате на `http://drzavno.petlja.org`. Потребно је да се улогујете са истим Петља налогом који сте користили до сада (рецимо, на квалификацијама). **Резултат сваког послатог решења добијате одмах (full feedback)** - детаљни приказ добијате кликом на лупу у делу "Pregled". Ако сте послали више решења за исти задатак, рачуна се оно са којим сте остварили највећи број поена. Питања у вези са формулацијом задатака можете упутити кроз такмичарско окружење у делу "Pitanja". Пре постављања питања, молимо вас да погледате део "Obaveštenja" јер је могуће да ћете одговор пронаћи тамо.

Тестирање задатака се обавља под оперативним системом Windows, на истој машини и (64-битном) систему као и за квалификације/окружно такмичење, уз коришћење званичних компајлера **FreePascal 2.6.4** и **GCC 5.3.0**. За компајлирање се користе следеће команде:

- За `C++` кодове: `g++ -O2 -std=c++14 -Wl,-stack -Wl,268435456 naziv_zadatka.cpp`
- За `Pascal` кодове: `fpc -O2 -Cs67107839 naziv_zadatka.pas`

Подзадаци

Подзадаци су групе тест примера које се оцењују као целина. Сваки подзадатак носи одређен број поена и неопходно је да ваше решење прође све тест примере из подзадатка да би сте добили те поене. Ако ваше решење макар у једном тест примеру да погрешан излаз, прекорачи временско или меморијско ограничење или се не заврши успешно, цео подзадатак вредеће 0 поена. Временско и меморијско ограничење се задаје на нивоу целог задатка, тј. исто је за све подзадатке. Систем приказује резултате подзадатака на следећи начин:



Преглед проблема

Б КАТЕГОРИЈА			
Задатак	**БОНД**	**ХАРИ**	**КЛАДИОНИЦА**
Улаз	СТАНДАРДНИ УЛАЗ (stdin)		
Излаз	СТАНДАРДНИ ИЗЛАЗ (stdout)		
Временско ограничење	1.0 sec	0.4 sec	0.7 sec
Меморијско ограничење	256 MB	64 MB	256 MB
Број поена	100	100	100

А КАТЕГОРИЈА			
Задатак	**КАРИБИ**	**ПРОРОК**	**КАРТЕ**
Улаз	СТАНДАРДНИ УЛАЗ (stdin)		
Излаз	СТАНДАРДНИ ИЗЛАЗ (stdout)		
Временско ограничење	1.5 sec	0.3 sec	0.6 sec
Меморијско ограничење	256 MB	64 MB	64 MB
Број поена	100	100	100

Напомена: У случају неслагања временских/меморијских окружења у овом прегледу и на окружењу преко кога шаљете задатке, валидна су ограничења на окружењу.

1. Бонд

Пошто је добио дозволу за убијање, добро познати агент Бонд, Џејмс Бонд, се упутио у, нама суседну, Црну Гору како би спасио свет. Он је сазнао да се озлоглашени Ле Шифр упутио у казино како би учествовао у турниру са великим улогом у игри Ројал. Како би осујетили планове Ле Шифра и натерали га да се преда, МИ6 планира да убаци Бонда у турнир, верујући да баш он може да победи. Као што то обично бива, до финала су стигли баш Бонд и Ле Шифр.

Ројал је веома једноставна игра. На почетку игре се одреди број k. Потом први играч баца шестострану коцкицу (свака страна коцкице има урезан број од 1 до 6) k пута и забележе се бројеви на којима коцкица стане приликом сваког бацања. Означимо добијене бројеве са $A = [a_1, a_2, ..., a_k]$. Затим други играч баца исту коцкицу k пута и забележи бројеве које добије. Означимо добијене бројеве са $B = [b_1, b_2, ..., b_k]$. После бацања коцкица следи упоређивање низова A и B и то се ради на следећи начин:

- Најпре се бројеви у низу A сортирају у неопадајући поредак и тако се добије нови низ $A' = [a'_1, a'_2, ..., a'_k]$.
- Слично се бројеви у низу B сортирају у неопадајући поредак како би се добио нови низ $B' = [b'_1, b'_2, ..., b'_k]$.
- На крају се бројеви у низовима упоређују један по један, тј. број a'_i се упоређује са b'_i (где $1 \leq i \leq k$). Са R_1 означимо број пута где је $a'_i > b'_i$, а са R_2 број пута где је $a'_i < b'_i$. Број поена које први играч освоји је баш R_1, а R_2 представља број поена другог играча.

Нпр. претпоставимо да је $k = 5$ и да први играч приликом бацања коцкице добија бројеве $A = [6, 4, 1, 4, 3]$, а други играч добија бројеве $B = [1, 5, 3, 2, 5]$. По сортирању ових низова добијамо низове $A' = [1, 3, 4, 4, 6]$ и $B' = [1, 2, 3, 5, 5]$. Упоређивањем бројева ова два низа имамо: $1 = 1$, $3 > 2$, $4 > 3$, $4 < 5$ и $6 > 5$, тј. $R_1 = 3$ и $R_2 = 1$.

Бонд је сазнао да ће у финалу да се користи специјална коцкица за коју може да предвиди на којим бројевима ће да стане за првих $2 \cdot N$ бацања, тј. у првих $2 \cdot N$ бацања, пашће се бројеви $X = [x_1, x_2, ..., x_{2 \cdot N}]$ редом. Зна се да ће Бонд бити први играч и још има право да одабере број k који представља број бацања коцкице сваког играча у игри. Међутим, мора да важи да укупан број бацања коцкице не сме бити већи од $2 \cdot N$, тј. $1 \leq k \leq N$.

Ваш задатак је да помогнете Бонду да одабере њему најповољније k, тј. такво k где ће освојити највише поена.

Опис улаза

У првој линији стандардног улаза налази се природан број N који означава максималан број бацања коцкице. У следећем реду се налази $2 \cdot N$ целих бројева $x_1, x_2, ..., x_{2 \cdot N}$ који представљају вредности које ће се пасти у првих $2 \cdot N$ бацања, редом (у првом бацању коцкица пашће се број x_1, у другом x_2, итд.).

Опис излаза

На стандарни излаз је потребно исписати један цео број, R_1, који представља максималан број поена које Бонд може да освоји.

Пример 1

Улаз	Излаз
`4` `2 4 6 1 5 3 5 2`	`3`

Пример 2

Улаз	Излаз
`2` `1 1 2 2`	`0`

Објашњење примера

У првом тест примеру, могуће вредности за k су 1, 2, 3 и 4.

- Уколико је $k = 1$, онда имамо $A = [x_1] = [2]$, $B = [x_2] = [4]$, тј. $A' = [2]$, $B' = [4]$ и овде имамо да је $R_1 = 0$.
- Уколико је $k = 2$, онда имамо $A = [x_1, x_2] = [2, 4]$, $B = [x_3, x_4] = [6, 1]$, tj. $A' = [2, 4]$, $B' = [1, 6]$ и овде имамо да је $R_1 = 1$.
- Уколико је $k = 3$, онда имамо $A = [x_1, x_2, x_3] = [2, 4, 6]$, $B = [x_4, x_5, x_6] = [1, 5, 3]$, tj. $A' = [2, 4, 6]$, $B' = [1, 3, 5]$ и овде имамо да је $R_1 = 3$.
- Уколико је $k = 4$, онда имамо $A = [x_1, x_2, x_3, x_4] = [2, 4, 6, 1]$, $B = [x_5, x_6, x_7, x_8] = [5, 3, 5, 2]$, tj. $A' = [1, 2, 4, 6]$, $B' = [2, 3, 5, 5]$ и овде имамо да је $R_1 = 1$.

Дакле, највећи број поена које Бонд може да освоји је 3. У другом тест примеру, које год k Бонд да одабере, имаће 0 поена.

Ограничења и подзадаци

- $1 \leq x_1, x_2, ..., x_{2 \cdot N} \leq 6$

Постоје 3 подзадатка у којима додатно важи::

- Подзадатак 1 [**21 поен**]: $1 \leq N \leq 100$.
- Подзадатак 2 [**31 поен**]: $1 \leq N \leq 5.000$.
- Подзадатак 3 [**48 поена**]: $1 \leq N \leq 1.000.000$.

2. Хари

Хогвортс, школа магије и чаробњаштва, припрема се за ускршњи распуст када сви ученици иду кући на заслужени одмор. Како је магија забрањена у свету обичних људи, чаробњаци не носе своје чаробне штапове, већ их остављају Хагриду на чување. Чаробни штапови се чувају у посебним магичним кутијама. Хагрид је врло сенилан, па је заборавио где је оставио магичне кутије, те мора да направи нове.

Да се штапови не би помешали, па да након распуста чаробњаци не могу да нађу свој штап, постоји посебна процедура. Најпре сви чаробњаци, њих N, оставе своје штапове на дугачком столу, а онда их Хагрид редом пакује у кутије. Хагрид мора да их пакује редом, тј свака кутија треба садржати **само узастопне штапове**.

Мора се водити рачуна и о стабилности кутија. У свакој кутији мора бити **исти број штапова**. Такође, штапови имају старост, изражену у годинама. Уколико се у једној кутији нађе више од K чаробних штапова исте старости, може доћи до велике магичне експлозије.

Хагрида занима на колико начина може да упакује штапове у кутије, а да испоштује сва правила.

Опис улаза

У првом реду стандардног улаза налазе се два природна броја, N - број чаробњака и K - највећи дозвољени број штапова исте старости који се могу наћи у једној кутији. У следећем реду налази се N целих бројева, $A_1, A_2, ..., A_N$ који представљају старости штапова.

Опис излаза

На стандардни излаз потребно је исписати један број, који представља број начина на који Хагрид може поделити штапове у кутије.

Пример 1

Улаз	Излаз
`6 1` `1 5 3 3 7 4`	`2`

Пример 2

Улаз	Излаз
`12 2` `1 2 3 4 2 3 4 2 1 2 3 4`	`5`

Објашњење примера

У првом примеру K је 1, што значи да сви штапови у једној кутији морају имати различиту старост. Могуће је ставити сваки штап у посебну кутију, или прва три у једну кутију, а следећа три у другу. Ако се стављају по два, друга кутија ће садржати штапове старости

3 и 3, што није стабилно. У другом примеру валидне су поделе на по 1, 2, 3, 4 и 6 штапова по кутији.

Ограничења и подзадаци

- $1 \leq K \leq 200.000$
- $1 \leq A_i \leq 10^9$

Постоје 4 подзадатка у којима додатно важи:

- Подзадатак 1 [**16 поена**]: $1 \leq N \leq 100$.
- Подзадатак 2 [**24 поена**]: $1 \leq N \leq 20.000$.
- Подзадатак 3 [**29 поена**]: $1 \leq N \leq 200.000$ и $1 \leq A_i \leq 10^6$.
- Подзадатак 4 [**31 поен**]: $1 \leq N \leq 200.000$.

3. Кладионица

Паметни кодер Аки је синоћ гледао филм *Ocean's eleven (Играј своју игру)*. Како му је филм био досадан, негде на половини се успавао и почео да сања о томе како може преварити кладионицу као актери филма. Сањао је да има приступ бази података кладионице и да може наместити резултат било које утакмице.

Одлучио је да се клади на фудбалске мечеве. Постоји N различитих фудбалских лига, а у свакој лиги по M утакмица за које је могуће уложити опкладу. За сваку утакмицу коју је одабрао је позната **квота** - природан број који представља коефицијент добитка који носи та утакмица уколико је резултат погођен.

Аки ће **из сваке лиге** изабрати бар једну утакмицу (могуће и више), и тако саставити **тикет**. За сваку лигу је познат број утакмица које је могуће изабрати из те лиге на истом тикету - нпр. постоји случај где је за неку лигу могуће изабрати тачно 2 или 4 утакмице, али није могуће изабрати 3.

Добитак тикета се одређује тако што се **помноже квоте** свих одабраних утакмица, а затим се добијени производ **помножи са бонусом** који зависи од броја одиграних утакмица. Бонуси за сваки могући број одиграних утакмица су дати од стране кладионице.

Како Аки у свом сну може наместити резултате свих утакмица, и жели да заради што више пара, он ће **саставити све могуће тикете**. Два тикета се разликују уколико бар једна утакмица постоји на једном тикету, а не постоји на другом. На крају, Аки ће зарадити онолико пара колико је **збир добитака са свих тикета**.

Израчунајте и испишите колико ће пара Аки зарадити у свом сну (рачунајући по модулу $10^9 + 7 = 1.000.000.007$).

Опис улаза

У првом реду се налазе два природна броја: N - број лига, и M - број утакмица у свакој лиги. У наредних N редова се налази по M природних бројева који представљају квоте за сваку утакмицу.

У следећих N редова се налази по M бројева који представљају матрицу C. Елементи те матрице могу имати вредност нула или један. Ако је j-ти број у i-том од тих редова једнак један ($C[i][j] = 1$), онда то значи да се из i-те лиге може изабрати j утакмица, а ако је једнак нули, онда се из i-те лиге не може изабрати j утакмица.

У последњем реду улаза има укупно $N \times M$ природних бројева где i-ти број у низу предстаља бонус за укупно одиграних i утакмица.

Опис излаза

У првом реду излаза исписати колико ће пара Аки зарадити, рачунајући по модулу $10^9+7 = 1.000.000.007$.

Пример

Улаз	Излаз

`2 3`
`2 3 4`
`5 6 7`
`1 0 1`
`0 1 1`
`10 20 30 40 50 60`
	`535290`

Објашњење примера

Постоје две лиге са по три утакмице. Квоте утакмица у првој лиги су 2, 3 и 4, док у другој лиги постоје утакмице са квотама 5, 6 и 7. Из прве лиге је могуће изабрати тачно једну или тачно три утакмице, док није могуће изабрати тачно две утакмице. Из друге лиге је могуће изабрати две или три утакмице. Неки од могућих тикета су:

- Из прве лиге бирамо прву утакмицу (квота 2), из друге лиге бирамо прву и другу утакмицу (квоте 5 и 6). Укупна квота је $2 \cdot 5 \cdot 6 = 60$. Пошто смо изабрали 3 утакмице, квота се множи са бонусом за 3 утакмице (30), и тако добијамо укупан добитак за овај тикет: $60 \cdot 30 = 2400$

- Из прве лиге бирамо све три утакмице (квоте 2, 3 и 4), из друге лиге бирамо прву и трећу утакмицу (квоте 5 и 7). Укупна квота је $2 \cdot 3 \cdot 4 \cdot 5 \cdot 7 = 840$. Како смо изабрали укупно 5 утакмица, квота се множи са бонусом за 5 утакмица (50) и тако добијамо укупан добитак за овај тикет: $840 \cdot 50 = 42000$

Комбиновањем свих могућих тикета, а онда сабирањем њихових добитака, добијамо укупну зараду од 535290.

Ограничења и подзадаци

- Квоте тикета и бонуси су природни бројеви између 1 и 1000.

Постоји 5 подзадатака у којима додатно важи:

- Подзадатак 1 [**7 поена**]: $N \cdot M \le 20$.
- Подзадатак 2 [**25 поена**]: $N = 1$, и $M \le 100$.
- Подзадатак 3 [**12 поена**]: $N, M \le 100$ и **из сваке лиге се може играти тачно једна утакмица** ($C[i][1] = 1$, $C[i][j] = 0$ ако је $j > 1$).
- Подзадатак 4 [**25 поена**]: $N, M \le 100$ и **сви бонси су тачно 1**.
- Подзадатак 5 [**31 поен**]: $N, M \le 100$.

4. Кариби

Некада давно, било је време програмера који нису тестирали своје кодове, време Програмера са Кариба. Ови програмери пловили су морем и сејали страх и неквалитетне програме дуж карипских острва; за собом су остављали уплакане кориснике, незавршене апликације и чланове посаде који би користили `C` уместо `C++`-а. Сваки брод који би им се супротставио, добио би флоатинг поинт ексепшн и престао би да флоат...

Популарно место за окупљање Програмера са Кариба било је Карипско острво Тортуга. Залив овог острва можемо представити **бинарном матрицом димензије** $n \times m$ **чије је свако поље копно (означено јединицом) или вода (означено нулом)**. Програмерски бродови долазе у разним дужинама и уколико је дужина неког брода x, то значи да тај брод заузима тачно x узастопних поља залива, било вертикално или хоризонтално (ширина брода је тачно једно поље). Према томе, **брод дужине** x **се може паркирати у залив ако и само ако постоји** x **узастопних поља у истој врсти или истој колони матрице залива при чему сва поља представљају воду и на ниједном од њих се не налази део неког другог брода**. У општем случају, може постојати више начина за паркирање.

Једног дана, у залив Тортуге стигао је програмер Цек Врабац на свом броду *Black Perl* **дужине** a и решио да га упаркира. Цек зна да после њега у залив стиже програмер Барбоса на свом броду *Blue Pascal* **дужине** b, као и да Барбоса највише на свету мрзи три ствари: програмера Цека, дебуг мод и недовољно места за паркирање брода. **Зато је Цек одлучио да паркира свој брод тако да после његовог паркирања буде најмање могуће начина да се у залив упаркира Барбосин брод**. Како програмер Цек не зна да програмира, затражио је помоћ од вас и као награду неће вам хаковати рачунар.

Опис улаза

У првом реду стандардног улаза налазе се четири природна броја n, m, a и b, раздвојена размаком, која, редом, представљају димензије залива, дужину Цековог брода и дужину Барбосиног брода. Затим следи опис залива - у наредних n редова налази се по m карактера (без размака) од којих је сваки '0' или '1'.

Опис излаза

У првом и јединoм реду стандардног излаза исписати један природан број - најмањи могући број начина за паркирање Барбосиног брода након паркирања Цековог брода.

Пример 1

Улаз	Излаз

`5 6 3 3`
`110111`
`000001`
`000001`
`111101`
`100010`
	`2`

Пример 2

Улаз	Излаз
2 5 2 5 00000 00000	0

Објашњење примера

У првом тест примеру су дужине Цековог и Барбосиног брода по 3. Уколико Цек паркира свој брод тако да заузима поља (1,3)(2,3)(3,3), Барбоса ће имати само два начина за паркирање свог брода - (2,5)(3,5)(4,5) и (5,2)(5,3)(5,4) (врсте су нумерисане одозго надоле а колоне с лева удесно). Цек никако не може паркирати свој брод тако да Барбоса има мање од два начина за паркирање.

Ограничења и подзадаци

- $1 \leq a \leq max\{n, m\}$, $2 \leq b \leq max\{n, m\}$.
- Залив ће бити такав да ће постојати бар један начин да Цек упаркира свој брод.

Постоји 5 подзадатака у којима додатно важи:

- Подзадатак 1 [**11 поена**]: $1 \leq n, m \leq 80$.
- Подзадатак 2 [**12 поена**]: $n = 1$ и $1 \leq m \leq 2000$.
- Подзадатак 3 [**13 поена**]: $a = 1$ и $1 \leq n, m \leq 2000$.
- Подзадатак 4 [**22 поена**]: $1 \leq n, m \leq 500$.
- Подзадатак 5 [**42 поена**]: $1 \leq n, m \leq 2000$.

5. Пророк

Након што је провео цео дан гледајући научнофантастичне филмове, Петар је одлучио да искористи идеје и знање које је из њих покупио и направи механичког пророка, односно машину која одговара на било какво да-не питање које јој се постави.

Након што је направио пророка, Петар му је поставио N питања и бележио одговоре које је добио. Да не би трошио превише папира, свако питање је скраћено обележио идентификационим бројем (тако да истим бројевима одговара исто питање – Петар је нека питања поставио више пута). Приметио је да су одговори које је добио контрадикторни, тј. да је на нека питања добио и одговор "да" и одговор "не".

Петар претпоставља да је проблем у неисправној простор-временској ферфуцни, због које пророк на свако K-то постављено питање одговара супротно од тачног одговора (где је $2 \le K \le N$). Да би могао да почне поправке одмах, уместо да чека да сазна праве одговоре, замолио вас је да нађете најмање K које одговара овој претпоставци и добијеним одговорима.

Опис улаза

У првом реду стандардног улаза налази се један природан број N – број питања која је Петар поставио. У наредних N редова налазе се идентификациони број i-тог питања Q_i и одговор који је пророк дао A_i. Све вредности A_i ће бити или "da" или "ne".

Опис излаза

У првом и јединном реду стандардног излаза исписати K – најмању вредност "периода"са којим је могуће да пророк греши. Уколико такво K не постоји, исписати -1.

Пример 1

Улаз	Излаз
5 1 da 2 ne 1 ne 3 da 2 ne	3

Пример 2

Улаз	Излаз
3 1 da 1 ne 1 ne	-1

Објашњење примера

У првом примеру, ако су одговори на питања са идентификационим бројевима $1, 2, 3$ редом "да", "не" и "да", одговори пророка су конзистентни ако је на треће постављено питање дао супротан одговор (тј. $K = 3$). За $K = 2$ се не слажу одговори на прво и треће питање (пророк на њих не би одговорио супротно), као ни одговори на друго и пето (пророк би на друго питање одговорио супротно а не пето не би).

Ограничења и подзадаци

- $1 \leq Q_i \leq 10^9$.

Постоје 3 подзадатка у којима додатно важи:

- Подзадатак 1 [**20 поена**]: $1 \leq N \leq 2.000$.
- Подзадатак 2 [**35 поена**]: $1 \leq N \leq 100.000$, и за све i важи $Q_i \leq 200$.
- Подзадатак 3 [**45 поена**]: $1 \leq N \leq 100.000$.

6. Карте

Вучко је претурајући по соби, иза једног баштенског реквизита који је добио од другара као рођендански поклон, пронашао стари шпил карата. На свакој карти био је исписан један природан број. Он је одмах позвао Коцу да играју следећу игру: Прво Вучко положи све карте лицем надоле а затим Коца бира две карте, чита бројеве са њих и затим тражи природан број већи од 1 који дели оба извучена броја. Ако се испостави да такав број не постоји, Вучко побеђује. У супротном, нико не побеђује (свакако Коца не побеђује).

Вучко спрема испит из вероватноће па жели да као вежбу израчуна вероватноћу да ће победити у овој игри. За то му је потребна ваша помоћ! Реците Вучку на колико начина се могу одабрати две карте тако да он победи.

Опис улаза

У првом реду стандардног улаза налази се један природан број N, број карата у шпилу. У наредних N редова налази се по један природан број A_i, број са i-те карте.

Опис излаза

На стандардни излаз исписати један цео број – број начина да се извуку две карте тако да Вучко побеђује.

Пример 1

Улаз	Излаз
3 2 3 6	1

Пример 2

Улаз	Излаз
4 1 6 10 15	3

Објашњење примера

У првом примеру, једини начин да Вучко победи јесте да Коца извуче прву и другу карту. У другом примеру, постоје три начина – ако Коца извуче прву карту и било коју од преостале три.

Ограничења и подзадаци

- $1 \le A_i \le 10^8$.

Постоје 4 подзадатака у којима додатно важи:

- Подзадатак 1 [**9 поена**]: $1 \le N, A_i \le 100$.
- Подзадатак 2 [**10 поена**]: $1 \le N \le 1.000$
- Подзадатак 3 [**30 поена**]: $1 \le N \le 7.000$.
- Подзадатак 4 [**51 поена**]: $1 \le N \le 30.000$.