IV Olimpiada Informatyczna Centralnej Europy Nowy SĆącz, lipiec 1997

teksty zadaĆń

Krzysztof Diks
autor zadania

Przemysława Kanarek, Marcin Kubica
przekład

Na drugi brzeg

Mieszkańcy Bajtocji uwielbiają wszelkie sporty, które oprócz sprawności fizycznej wymagają również sprytu i inteligencji. Jedną z takich dyscyplin jest przeprawianie się przez Hex-Dunajec — najszerszą rzekę Bajtocji. W poprzek Hex-Dunajca, od lewego brzegu do prawego, ustawionych jest n pali oznaczonych kolejno liczbami $1, \ldots, n$. Najbliżej lewego brzegu jest pal nr 1, a najbliżej prawego pal nr n. Zawodnik musi przeprawić się z lewego brzegu na prawy skacząc po palach.

Przeprawa rozpoczyna się w chwili 0. Zawodnik startuje z lewego brzegu rzeki i ma za zadanie osiągnąć prawy brzeg w jak najkrótszym czasie. W każdej chwili, każdy z pali jest wynurzony albo zanurzony. W każdej chwili zawodnik może stać na brzegu lub na palu. Na palu można stać jedynie wówczas, gdy jest on wynurzony.

W chwili 0 wszystkie pale są zanurzone. Dalsze zachowanie każdego pola opisują związane z nim dwie liczby całkowite a i b. Począwszy od chwili 1 przez a jednostek czasu pal pozostaje wynurzony. Przez kolejnych b jednostek czasu pal jest zanurzony. Następnie ponownie wynurza się na a jednostek czasu, potem zanurza się na b jednostek, itd. Np. jeżeli dla pewnego pala $a = 2$ i $b = 3$, to pal jest zanurzony w chwili 0, wynurza się na chwile 1 i 2, zanurza się na chwile 3, 4 i 5 itd.

Jeżeli w chwili t, zawodnik znajduje się w pewnym miejscu (na palu wynurzonym w chwili t lub na brzegu), to w chwili $t + 1$ może znaleźć się w dowolnym miejscu (na palu wynurzonym w chwili $t + 1$ lub na brzegu) odległym o co najwyżej 5 pali. Np. stojąc w chwili t na palu nr 5, w chwili $t + 1$ zawodnik może znaleźć się na lewym brzegu lub na dowolnym z pali 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 (oczywiście, o ile pal jest wynurzony).

Zadanie

Napisz program, który:

- *wczytuje z pliku tekstowego RIV.IN liczbę bloków danych (każdy blok zawiera kompletny opis jednego przykładu zadania);*
- *dla każdego bloku:*
  - *wczytuje liczbę pali oraz opis ich zachowania;*
  - *oblicza najkrótszy możliwy czas potrzebny do przeprawienia się na prawy brzeg, o ile przeprawa jest możliwa;*
  - *zapisuje wynik w pliku tekstowym RIV.OUT.*

Wejście

Pierwszy wiersz pliku RIV.IN zawiera liczbę bloków danych x, $1 \leqslant x \leqslant 5$. Kolejne wiersze zawierają x bloków danych. Pierwszy blok rozpoczyna się w drugiej linii pliku wejściowego, a kolejne bloki następują bezpośrednio po sobie.

Pierwszy wiersz każdego bloku zawiera liczbę pali n, $5 < n \leqslant 1000$. W każdym z kolejnych n wierszy bloku znajdują się dwie liczby całkowite a i b ($1 \leqslant a, b \leqslant 5$) oddzielone pojedynczą spacją. Zachowanie i-tego pala opisują liczby z $(i + 1)$-go wiersza bloku, $1 \leqslant i \leqslant n$.

Wyjście

Dla k-tego bloku danych, $1 \leqslant k \leqslant x$, Twój program powinien zapisywać wynik w k-tym wierszu pliku tekstowego RIV.OUT. Jeśli można przeprawić się przez rzekę, to wynikiem jest liczba całkowita będąca minimalną liczbą jednostek czasu koniecznych do przeprawy. Jeśli nie można przeprawić się przez rzekę, to wynikiem jest słowo NO.

Przykład

Dla pliku tekstowego RIV.IN:

```
2
```

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

prawidłowym rozwiązaniem jest plik tekstowy RIV.OUT;

```
NO
4
```

Krzysztof Diks
autor zadania

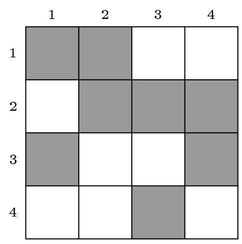
Przemysława Kanarek, Marcin Kubica
przekład

Zawody Strzeleckie

Witamy na Dorocznych Zawodach Strzeleckich Bajtocji. Każdy z zawodników strzela do tarczy w kształcie prostokąta. Tarcza jest podzielona na r wierszy i c kolumn, czyli składa się z $r \times c$ kwadratowych pól. Każde z pól tarczy jest zamalowane na czarno lub biało. Wiadomo, że w każdej kolumnie tarczy są dokładnie dwa pola białe i $r-2$ czarne. Wiersze tarczy numerujemy od góry w dół kolejnymi liczbami naturalnymi $1,\ldots,r$. Kolumny tarczy numerujemy od lewej do prawej liczbami $1,\ldots,c$. Każdy z zawodników strzela serię złożoną z c strzałów. Seria c strzałów jest poprawna jeżeli w jej trakcie trafiono dokładnie po jednym białym polu w każdej kolumnie i nie pozostał żaden wiersz, w którym nie byłoby trafionego białego pola. Pomóż zawodnikowi znaleźć poprawną serię strzałów, o ile taka seria istnieje.

Przykład

Przyjmijmy, że tarcza wygląda jak na poniższym rysunku:



Poprawna seria strzałów, to np. trafienia w pola znajdujące się w wierszach 2, 3, 1, 4 odpowiednio w kolejnych kolumnach 1, 2, 3, 4.

Zadanie

Napisz program, który:

- *wczytuje liczbę bloków danych z pliku tekstowego SHO.IN; każdy blok zawiera kompletny opis jednej tarczy;*
- *następnie dla każdego bloku*
  - *wczytuje rozmiar tarczy i rozkład białych pól na tarczy;*
  - *sprawdza, czy istnieje poprawna seria strzałów dla tej tarczy; jeśli tak, to znajduje jedną z nich;*
  - *zapisuje wynik w pliku tekstowym SHO.OUT;*

Wejście

W pierwszym wierszu pliku danych SHO.IN podana jest liczba bloków danych (opisów tarcz) x, $1 \leqslant x \leqslant 5$. W kolejnych wierszach opisanych jest x bloków. Opis pierwszego bloku rozpoczyna się w drugim wierszu pliku wejściowego, a opis każdego kolejnego bloku rozpoczyna się w wierszu bezpośrednio następującym po zakończeniu opisu poprzedniego bloku.

Opis każdego bloku (tarczy) rozpoczyna się od wiersza zawierającego dwie liczby całkowite r i c ($2 \leqslant r \leqslant c \leqslant 1000$) oddzielone pojedynczym odstępem. Liczby te oznaczają odpowiednio liczbę wierszy i kolumn tarczy. Każdy z kolejnych c wierszy bloku (opisu tarczy) zawiera po dwie liczby oddzielone pojedynczym odstępem. Liczby w $(i+1)$-szym wierszu bloku ($1 \leqslant i \leqslant c$) oznaczają pozycje (tzn. numery wierszy na planszy) białych pól w i-tej kolumnie tarczy.

Wynik

Dla i-tego bloku, $1 \leqslant i \leqslant x$, Twój program powinien zapisać w i-tym wierszu pliku wynikowego SHO.OUT ciąg c numerów wierszy (oddzielonych pojedynczymi spacjami) tworzących opis poprawnej serii strzałów w białe pola *kolejnych kolumn $1, 2, \ldots, c$. Jeżeli dla podanej tarczy poprawna seria nie istnieje, w i-tym wierszu pliku SHO.OUT należy wpisać słowo NO.*

Przykład

Dla pliku tekstowego SHO.IN:

```
2
4 4
2 4
3 4
1 3
1 4
5 5
1 5
2 4
3 4
2 4
2 3
```

jedną z poprawnych odpowiedzi jest plik tekstowy SHO.OUT:

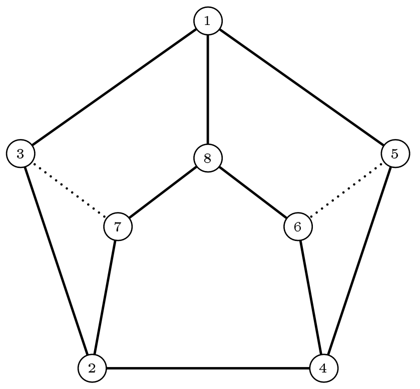
```
2 3 1 4
NO
```

Krzysztof Diks
autor zadania

Przemysława Kanarek, Marcin Kubica
przekład

Jaskinie

W Bajtocji jest wiele jaskiń. Oto mapa jednej z nich:



W Bajtocji każda z jaskiń ma następujące cechy:

- *wszystkie komnaty i korytarze leżą na tym samym poziomie,*
- *korytarze nie przecinają się,*
- *część komnat leży na obwodzie jaskini — nazywamy je* komnatami zewnętrznymi*,*
- *wszystkie pozostałe komnaty, leżące wewnątrz, nazywamy* komnatami wewnętrznymi*,*
- *wejście do jaskini prowadzi do jednej z zewnętrznych komnat,*
- *z każdej komnaty wychodzą dokładnie trzy korytarze, prowadzące do trzech różnych innych komnat; jeśli komnata jest zewnętrzna, to dwa spośród korytarzy prowadzą do sąsiednich komnat zewnętrznych na obwodzie jaskini, a jeden do komnaty wewnętrznej,*
- *korytarze łączące zewnętrzne komnaty nazywamy* korytarzami zewnętrznymi*, a wszystkie pozostałe* korytarzami wewnętrznymi*,*
- *z każdej komnaty można dojść do każdej innej komnaty korzystając jedynie z wewnętrznych korytarzy, ale jeśli dodatkowo wymagamy by przez każdy korytarz przechodzić co najwyżej raz, to można to zrobić tylko w jeden sposób,*
- *nie wszystkie korytarze są jednakowo łatwe do pokonania — dzielimy je na dwie kategorie: łatwe i trudne.*

Postanowiono udostępnić wszystkie jaskinie dla zwiedzających. Aby zapewnić płynny i bezpieczny przepływ zwiedzających, w każdej z jaskiń należy ustalić trasę zwiedzania, która przechodzi dokładnie raz przez każdą komnatę.

Jedynym wyjątkiem od tej zasady jest komnata wejściowa, od której zwiedzanie się zaczyna i na której się kończy (tzn. zwiedzający przechodzą przez tę komnatę dokładnie dwa razy). Trasa zwiedzania powinna być przeznaczona dla przeciętnego turysty i zawierać jak najmniej trudnych korytarzy.

Przykład

Przyjrzyjmy się jaskini przedstawionej na rysunku. Komnata wejściowa ma numer 1. Trudne korytarze są zaznaczone linią przerywaną. Trasa 1, 5, 4, 6, 8, 7, 2, 3 nie przechodzi przez żaden z trudnych korytarzy. Trasa zwiedzania kończy się oczywiście w komnacie nr 1, chociaż pomijamy tę jedynkę na końcu opisu trasy.

Zadanie

Napisz program, który:

- *wczytuje opis jaskini z pliku tekstowego CAV.IN;*
- *znajduje trasę zwiedzania, która zaczyna i kończy się w komnacie wejściowej, przechodzi przez każdą komnatę dokładnie raz oraz prowadzi przez jak najmniejszą liczbę trudnych korytarzy;*
- *zapisuje wynik w pliku tekstowym CAV.OUT.*

Wejście

W pierwszym wierszu pliku tekstowego CAV.IN znajdują się dwie liczby całkowite n, k (oddzielone pojedynczym odstępem). Liczba n, $3 < n \leqslant 500$, to liczba wszystkich komnat w jaskini, a k, $k \geqslant 3$, to liczba komnat zewnętrznych. Komnaty są ponumerowane od 1 do n. Komnata wejściowa ma numer 1. Komnaty zewnętrzne mają numery $1, 2, \ldots, k$, aczkolwiek nie muszą leżeć na obwodzie jaskini w tej kolejności. Kolejne $3n/2$ wiersze pliku zawierają opisy korytarzy. Opis każdego korytarza składa się z trzech liczb całkowitych a, b, c, (oddzielonych pojedynczymi spacjami). Liczby a i b są numerami komnat, które łączy korytarz. Liczba c jest równa 0 lub 1 — 0 oznacza, że korytarz jest łatwy, a 1, że trudny.

Wyjście

Twój program powinien zapisać w pierwszym wierszu pliku tekstowego CAV.OUT ciąg n liczb całkowitych, pooddzielanych pojedynczymi odstępami. Pierwszą liczbą powinno być 1 (numer komnaty wejściowej), a kolejnymi $n-1$ liczbami powinny być numery kolejnych komnat trasy zwiedzania.

Przykład

Dla tekstowego pliku wejściowego CAV.IN:

```
8 5
1 3 0
3 2 0
7 3 1
7 2 0
8 7 0
1 8 0
6 8 0
6 4 0
6 5 1
5 4 0
2 4 0
5 1 0
```

jednym z poprawnych wyników jest następujący plik tekstowy CAV.OUT:

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

P. Chrząstowski-Wachtel, W. Guzicki
autorzy zadania

P. Kanarek, M. Kubica
przekład

Przedziały liczb całkowitych

Przedział liczb całkowitych $[a..b]$, gdzie $a < b$, to zbiór wszystkich kolejnych liczb całkowitych rozpoczynających się od a i kończących na b.

Zadanie

Napisz program, który:

- *wczytuje liczbę przedziałów i ich opisy z pliku tekstowego INT.IN;*
- *znajduje minimalną liczbę elementów zbioru zawierającego co najmniej dwie różne liczby całkowite z każdego z przedziałów;*
- *zapisuje wynik w pliku tekstowym INT.OUT.*

Wejście

Pierwszy wiersz pliku tekstowego INT.IN zawiera liczbę przedziałów n, $1 \leqslant n \leqslant 10000$. Każdy z kolejnych n wierszy zawiera dwie liczby całkowite a i b, $0 \leqslant a < b \leqslant 10000$, oddzielone pojedynczą spacją. Liczby te oznaczają początek i koniec przedziału.

Wyjście

Twój program powinien zapisać w pierwszym wierszu pliku tekstowego INT.OUT jedną liczbę całkowitą; ma to być minimalna liczba elementów zbioru zawierającego co najmniej dwie różne liczby z każdego z przedziałów.

Przykład

Dla pliku tekstowego INT.IN

```
4
3 6
2 4
0 2
4 7
```

prawidłowym rozwiązaniem jest plik tekstowy INT.OUT:

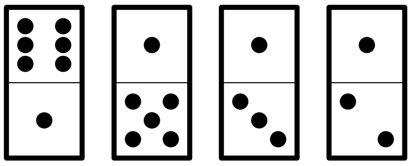
```
4
```

Krzysztof Diks
autor zadania

Marcin Kubica
przekład

Domino

Kostki domino to niewielkie prostokątne klocki. Na każdym klocku są dwa kwadratowe pola, a każde pole jest puste, lub znajduje się na nim od 1 do 6 oczek. Na stole ułożono rząd kostek domino, np.:



Liczba oczek w górnym rzędzie pól wynosi $6 + 1 + 1 + 1 = 9$*, a w dolnym* $1 + 5 + 3 + 2 = 11$*.*

Różnica (wartość bezwzględna z różnicy) liczb oczek w górnym i dolnym rzędzie wynosi 2.

Każdą kostkę domino można obrócić o 180 stopni (oczywiście kostka musi pozostać odwrócona oczkami do góry).

Jaka jest najmniejsza liczba kostek, które trzeba obrócić, aby różnica (jw. wartość bezwzględna z różnicy) liczb oczek w górnym i dolnym rzędzie była minimalna?

W sytuacji przedstawionej na powyższym rysunku wystarczy obrócić skrajnie prawą kostkę, aby zmniejszyć różnicę do 0.

Zadanie

Napisz program, który:

- *wczytuje z pliku tekstowego DOM.IN liczbę kostek domino i ich opis,*
- *wylicza najmniejszą liczbę kostek, które trzeba obrócić, aby różnica liczb oczek w górnym i dolnym rzędzie była minimalna,*
- *wypisuje wynik do pliku tekstowego DOM.OUT.*

Wejście

Pierwszy wiersz pliku tekstowego DOM.IN zawiera liczbę całkowitą n, $1 \leqslant n \leqslant 1000$*. Jest to liczba kostek domino ułożonych na stole.*

W każdym z n *kolejnych wierszy zapisane są po dwie liczby całkowite* a *i* b, $0 \leqslant a, b \leqslant 6$*, oddzielone pojedynczym odstępem. Liczby* a *i* b *zapisane w* $(i + 1)$*-szym wierszu pliku wejściowego,* $1 \leqslant i \leqslant n$*, opisują* i*-tą kostkę — odpowiednio liczbę oczek na jego górnym i dolnym polu.*

Wyjście

Twój program powinien wypisać dokładnie jedną liczbę całkowitą, w pierwszym wierszu pliku tekstowego DOM.OUT. Powinna to być najmniejsza liczba kostek, które trzeba obrócić, aby różnica liczb oczek w górnym i dolnym rzędzie była minimalna.

Przykład

Dla pliku tekstowego DOM.IN:

```
4
6 1
1 5
1 3
1 2
```

poprawnym wynikiem jest plik tekstowy DOM.OUT:

```
1
```

Wojciech Rytter
autor zadania

Przemysława Kanarek, Marcin Kubica
przekład

Liczby szesnastkowe

Podstawą systemu szesnastkowego jest liczba 16. Do zapisu liczb w tym systemie używamy 16 cyfr, które oznaczamy $0, 1, \ldots, 9, A, B, C, D, E, F$. *Wielkie litery* $A, \ldots, F$ *oznaczają odpowiednio* $0, \ldots, 15$. *Na przykład wartość liczby szesnastkowej* $CF2$ *wynosi w systemie dziesiętnym* $12 \cdot 16^2 + 15 \cdot 16 + 2 = 3314$. *Niech* X *oznacza zbiór wszystkich dodatnich liczb całkowitych, których szesnastkowa reprezentacja ma najwyżej 8 cyfr. Ponadto żadna z cyfr w reprezentacji nie powtarza się i najbardziej znacząc cyfra jest różna od zera. Największym elementem w* X *jest liczba o reprezentacji szesnastkowej* $FEDCBA98$, *drugą co do wielkości liczbą jest* $FEDCBA97$, *trzecią* $FEDCBA96$, *itd.*

Zadanie

Napisz program, który:

- *czyta dodatnią liczbę całkowitą* n *z pliku tekstowego HEX.IN,* n *jest nie większe niż liczba elementów w zbiorze* X;
- *znajduje* n*-ty co do wielkości (licząc od największego) element zbioru* X;
- *zapisuje wynik w pliku tekstowym HEX.OUT.*

Wejście

Pierwszy wiersz pliku HEX.IN zawiera liczbę n *zapisaną w systemie dziesiętnym.*

Wyjście

Twój program powienien wypisać w systemie szesnastkowym w pierwszym wierszu pliku tekstowego HEX.OUT n*-ty co do wielkości (licząc od największego) element zbioru* X.

Przykład

Dla pliku tekstowego HEX.IN:

```
11
```

prawidłowym rozwiązaniem jest plik tekstowy HEX.OUT:

```
FEDCBA87
```