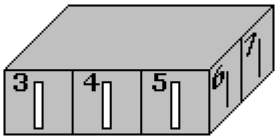
VII Bałtycka Olimpiada Informatyczna, Sopot 2001

VII Bałtycka Olimpiada Informatyczna — treści zadań

Box of Mirrors

Profesor Andrus uwielbia rozwiązywać przeróżne łamigłówki. Jedną z jego ulubionych jest "Lustrzana Skrzynka". Konstrukcję skrzynki najłatwiej opisać patrząc na nią z góry. Załóżmy więc, że widzimy poziomy przekrój skrzynki narysowany w prostokątnym układzie współrzędnych. Jest to prostokąt o bokach równoległych do osi układu podzielony na $n \times m$ *kwadratowych pól (ułożonych w* n *wierszy i* m *kolumn). W każdym polu może być umieszczone lustro. Lustro jest ustawione pionowo po przekątnej pola biegnącej od lewego–dolnego do prawego–górnego narożnika (przekroju) pola. Obie strony lustra odbijają światło.*



W zewnętrznych ścianach skrzynki, pośrodku każdego wiersza i każdej kolumny, znajdują się otwory, przez które może wpadać do wnętrza lub wychodzić na zewnątrz skrzynki wiązka światła. Przez każdy otwór można wpuścić do wnętrza skrzynki wiązkę światła jedynie w kierunku prostopadłym do ściany, w której znajduje się otwór. Taka wiązka odbijając się od lustra zmienia kierunek o 90 stopni. Gdy wiązka przechodzi przez puste pole (takie, na którym nie ma lustra), wówczas jej kierunek nie ulega zmianie. Otwory w ścianach skrzynki są ponumerowane od 1 *do* $2 \cdot (n+m)$*. Numery są nadawane otworom zgodnie z kolejnością ich występowania na obwodzie skrzynki, począwszy od otworu w lewej ścianie górnego–lewego pola (na przekroju) i następnie w kierunku przeciwnym do ruchu wskazówek zegara (czyli idąc najpierw w dół lewej ściany). Ponieważ z zewnątrz nie widać układu luster, więc jedynym sposobem, by wywnioskować, jaki jest ten układ, jest wpuszczanie wiązek światła przez wybrane otwory i obserwowanie, przez które otwory takie wiązki wychodzą.*

Zadanie

Napisz program, który:

- *wczyta z pliku tekstowego* `box.in` *rozmiar skrzynki i numery otworów, przez które wpadają i wychodzą wiązki światła,*
- *obliczy, na których polach znajdują się lustra, a które pola są puste,*
- *zapisze wynik w pliku tekstowym* `box.out`.

Jeżeli istnieje więcej niż jedno rozwiązanie, to program powinien podać dowolne z nich.

Wejście

W pierwszym wierszu pliku `box.in` *znajdują się dwie liczby naturalne:* n *(liczba wierszy pól,* $1 \le n \le 100$*) oraz* m *(liczba kolumn pól,* $1 \le m \le 100$*) oddzielone pojedynczym odstępem. Każdy z kolejnych* $2 \cdot (n+m)$ *wierszy zawiera po jednej liczbie naturalnej. Liczba w* $(i+1)$*-szym wierszu oznacza numer otworu, przez który wyjdzie wiązka światła, która wpada do skrzynki przez otwór o numerze* i*.*

Wyjście

Twój program powinien zapisać w pliku wynikowym `box.out` n *wierszy, z których każdy powinien zawierać* m *liczb oddzielonych pojedynczymi odstępami. Liczba* j*-ta w* i*-ym wierszu powinna być równa* `1`*, jeżeli na polu w* i*-tym wierszu i* j*-tej kolumnie znajduje się lustro; w przeciwnym razie (gdy pole to jest puste) liczba ta powinna być równa* `0`*.*

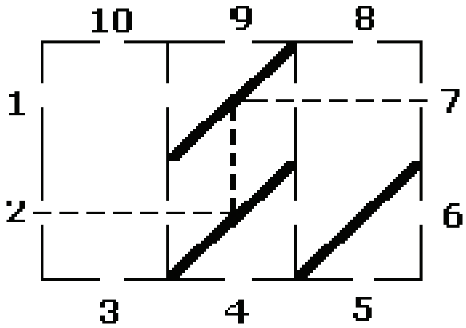
Przykład

Dla pliku wejściowego `box.in`*:*

```
2 3
9
7
10
8
6
5
2
4
1
3
```

poprawną odpowiedzią jest plik wyjściowy `box.out`*:*

```
0 1 0
0 1 1
```



Crack the Code

Kryptografia zajmuje się kodowaniem informacji w taki sposób, że tylko uprawniony odbiorca jest w stanie odczytać zakodowany tekst. Z kolei kryptoanaliza zajmuje się łamaniem kodów.

Załóż, że jesteś właśnie kryptoanalitykiem, a Twoim zadaniem jest odczytanie kilku zaszyfrowanych wiadomości przechwyconych przez policję w lokalu mafii.

Twoi koledzy już uzyskali program szyfrujący używany przez mafię. Jego tekst znajduje się w plikach `crack.pas` *i* `crack.c`*. To co zostało do zrobienia, to odwrócić algorytm szyfrujący i odgadnąć klucze użyte do zakodowania danych.*

Wraz z zaszyfrowanymi wiadomościami masz dostęp do kilku próbek tekstu niezaszyfrowanego, pochodzącego z tego samego źródła, co zakodowane wiadomości i — jak można przypuszczać — mającego podobną strukturę co do użytego języka, zasobu słów itp.

Zadanie

Twoje zadanie polega na odkodowaniu zaszyfrowanych wiadomości i zapamiętaniu ich w określonych plikach. Nie musisz dostarczać żadnego programu. Wystarczy, jeśli zapiszesz odkodowane teksty.

Wejście

Dysponujesz kilkoma zestawami danych. Jeden zestaw składa się z plików `cra`n`.*`*, gdzie* n *jest numerem zestawu. Każdy zestaw składa się z plików:*

- `cra*.in`*, zaszyfrowana wiadomość,*
- `cra*.txt`*, pliki tekstowe pochodzące z tego samego źródła, co zaszyfrowana wiadomość.*

Wyjście

Dla każdej zaszyfrowanej wiadomości `cra*.in`*, powienieneś utworzyć plik* `cra*.out` *z odszyfrowaną wiadomością.*

Excursion

Grupa podróżników zamierza odwiedzić wybrane miasta. Każdy podróżnik określa dwa życzenia dotyczące odwiedzenia bądź nieodwiedzenia danego miasta. Jedno życzenie to stwierdzenie, że się chce odwiedzić dane miasto, albo że się nie chce odwiedzać danego miasta. Jedno życzenie dotyczy tylko jednego miasta. Można w obu życzeniach odnieść się do tego samego miasta; zarówno zgodnie — wtedy oba życzenia są identyczne — jak i przeciwstawnie, czyli np. „Ja chcę odwiedzić miasto A" i „Ja nie chcę odwiedzać miasta A".

Zadanie

Twoje zadanie polega na napisaniu programu, który

- *wczyta życzenia podróżnika z pliku* `exc.in`*,*
- *określi, czy można wybrać taką listę (być może pustą) odwiedzanych miast, aby spełnić przynajmniej jedno życzenie każdego podróżnika,*
- *zapisze listę odwiedzanych miast w pliku wyjściowym* `exc.out`*.*

Jeżeli jest kilka możliwych rozwiązań, Twój program powinien zapisać dowolne z nich.

Wejście

Pierwszy wiersz pliku `exc.in` *zawiera dwie dodatnie liczby całkowite n oraz m ($1 \leq n \leq 20\,000$, $1 \leq m \leq 8\,000$); n jest liczbą podróżników, zaś m jest liczbą miast. Podróżnicy są ponumerowani od 1 do n, a miasta od 1 do m. Każdy z kolejnych wierszy pliku zawiera dwie różne od zera liczby całkowite oddzielone pojedynczym odstępem. Wiersz $i+1$ zawiera liczby w_i oraz w'_i oznaczające życzenia i-tego podróżnika, $-m \leq w_i, w'_i \leq m$, $w_i, w'_i \neq 0$. Liczba dodatnia oznacza, że podróżnik chce odwiedzić, a ujemna, że podróżnik nie chce odwiedzać miasta, którego numer jest równy wartości bezwzględnej tej liczby.*

Wyjście

W pierwszym wierszu pliku wyjściowego `exc.out`*, Twój program powinien zapisać jedną nieujemną liczbę l określającą proponowaną liczbę miast do odwiedzenia. W drugim wierszu tego pliku powinno się znaleźć dokładnie l dodatnich liczb całkowitych, określających miasta, które mają być odwiedzone. Liczby te muszą być podane w porządku rosnącym.*

Gdyby takiej listy miast nie dało się stworzyć, Twój program powinien umieścić w pierwszym i jedynym wierszu pliku wyjściowego jedno słowo `NO`*.*

Przykład

Dla pliku wejściowego `exc.in`*:*

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

poprawną odpowiedzią jest plik wyjściowy `exc.out`*:*

```
2
3 4
```

Knights

Dana jest szachownica o wymiarach $n \times n$, z której usunięto pewną liczbę pól. Należy wyznaczyć maksymalną liczbę skoczków (koników) szachowych, które można ustawić na pozostałych polach szachownicy tak, żeby żadne dwa skoczki nie atakowały się nawzajem.

	x		x	
x				x
		S		
x				x
	x		x	

Rysunek 1: Skoczek umieszczony w polu S atakuje pola oznaczone przez x.

Zadanie

Napisz program, który:

- *wczyta opis szachownicy z usuniętymi polami z pliku tekstowego* `kni.in`*,*
- *wyznaczy maksymalną liczbę wzajemnie nie atakujących się skoczków szachowych, które można ustawić na tej szachownicy,*
- *zapisze wynik w pliku tekstowym* `kni.out`*.*

Wejście

W pierwszym wierszu pliku tekstowego `kni.in` *znajdują się dwie liczby całkowite n i m, gdzie $1 \le n \le 200$, $0 \le m < n^2$. Liczba n oznacza rozmiar szachownicy, a m oznacza liczbę usuniętych pól.*

W każdym z kolejnych m wierszy jest zapisana para liczb naturalnych x i y, gdzie $1 \le x, y \le n$, oddzielonych pojedynczym odstępem. Są to współrzędne usuniętych pól. Lewy górny róg szachownicy ma współrzędne (1,1), natomiast prawy dolny róg ma współrzędne (n,n). Pola nie powtarzają się.

Wyjście

Plik tekstowy `kni.out` *powinien zawierać dokładnie jeden wiersz, zawierający pojedynczą liczbę całkowitą równą maksymalnej liczbie wzajemnie nie atakujących się skoczków, które można ustawić na zadanej szachownicy.*

Przykład

Dla pliku wejściowego `kni.in`*:*

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

poprawną odpowiedzią jest plik wyjściowy `kni.out`*:*

```
5
```

Marcin Kubica
Tłumaczenie

Mars maps

W roku 2051 kilka ekspedycji marsjańskich wybrało się w różne rejony Czerwonej Planety i wykonało mapy tych terenów. BAK (Bałtycka Agencja Kosmiczna) ma ambitne plany: zamierza stworzyć mapę całej planety. Aby przewidzieć skalę zadania pracownicy agencji muszą znać całkowitą powierzchnię skartowanego dotychczas terenu. Twoje zadanie polega na napisaniu programu, który tę powierzchnię obliczy.

Zadanie

Napisz program, który:

- *wczyta z pliku wejściowego* `mar.in` *opis obszarów pokrytych przez mapy,*
- *obliczy całkowitą powierzchnię obszaru pokrytego przez mapy,*
- *zapisze wynik w pliku wyjściowym* `mar.out`*.*

Wejście

Pierwszy wiersz pliku wejściowego `mar.in` *zawiera jedną liczbę całkowitą* N *(*$1 \leq N \leq 10\,000$*). Jest to liczba dostępnych map. Każdy z następnych* N *wierszy zawiera cztery liczby całkowite* x_1, y_1, x_2 *oraz* y_2 *(*$0 \leq x_1 < x_2 \leq 30\,000$, $0 \leq y_1 < y_2 \leq 30\,000$*). Wartości* (x_1, y_1) *oraz* (x_2, y_2) *to współrzędne odpowiednio lewego–dolnego i prawego–górnego rogu opisywanej mapy. Każda z map ma kształt prostokąta o bokach równoległych do osi układu współrzędnych.*

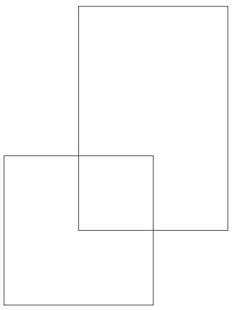
Wyjście

Plik wyjściowy `mar.out` *powinien zawierać jedną liczbę całkowitą* A *— całkowite pole powierzchni skartowanego obszaru (czyli pole powierzchni sumy wszystkich prostokątów).*

Przykład

Dla pliku wejściowego `mar.in`*:*

```
2
10 10 20 20
15 15 25 30
```



poprawną odpowiedzią jest plik wyjściowy `mar.out`*:*

```
225
```

Marcin Kubica
Tłumaczenie

Postman

Wiejski listonosz musi dostarczać pocztę wszystkim mieszkańcom okolicy, którzy zamieszkują w wioskach i przy drogach łączących wioski.

Musisz pomóc listonoszowi wytyczyć trasę, która pozwoli mu przejechać wzdłuż każdej drogi i odwiedzić każdą wioskę w okolicy przynajmniej raz. Tak się szczęśliwie składa, że w rozważanych przykładach taka trasa zawsze istnieje. Jednak wytyczone trasy mogą się różnić jakością, tzn. listonosz może otrzymywać różną zapłatę za swą pracę w zależności od wybranej trasy (jak się za chwilę przekonamy, to nie zysk listonosza jest najważniejszy, a zysk jego firmy, czyli poczty). Mieszkańcy każdej wioski chcieliby, by listonosz docierał do nich jak najwcześniej. Każda wioska zawarła więc z pocztą następującą umowę: jeżeli i-ta wioska jest odwiedzana przez listonosza jako k-ta w kolejności (tzn. listonosz odwiedził $k-1$ **różnych** *wiosek, zanim po raz pierwszy dotarł do wioski i) oraz $k \leq w(i)$, to wioska płaci poczcie $w(i) - k$ euro. Jeśli jednak $k > w(i)$, to wówczas poczta płaci wiosce $k - w(i)$ euro. Ponadto poczta płaci listonoszowi jedno euro za każdy przejazd między dwiema kolejnymi wioskami na jego trasie.*

W rozważanej okolicy jest n wiosek, które oznaczamy liczbami naturalnymi od 1 do n. Poczta znajduje się w wiosce oznaczonej numerem 1, a więc trasa listonosza musi rozpoczynać się w tej wiosce. W każdej wiosce zbiega się 2, 4 lub 8 dróg. Pomiędzy dwiema wioskami może istnieć kilka różnych dróg; droga może także powracać do tej samej wioski, z której wyszła.

Zadanie

Twoim zadaniem jest napisanie programu, który:

- *wczyta opis wiosek i łączących je dróg z pliku tekstowego* `pos.in`,
- *znajdzie trasę, która prowadzi przez każdą wioskę i wzdłuż każdej drogi, i która pozwala osiągnąć poczcie maksymalny zysk (ewentualnie ponieść minimalną stratę),*
- *zapisze wynik w pliku tekstowym* `pos.out`.

Jeżeli istnieje więcej niż jedno rozwiązanie, to Twój program powinien obliczyć jedno z nich.

Wejście

W pierwszym wierszu pliku tekstowego `pos.in` *zapisane są dwie liczby naturalne n i m oddzielone pojedynczym odstępem; liczba n ($1 \leq n \leq 200$) oznacza liczbę wiosek, a m jest liczbą dróg. W każdym z kolejnych n wierszy znajduje się jedna liczba naturalna (dodatnia). Liczba w $(i+1)$-szym wierszu oznacza $w(i)$ ($1 < w(i) \leq 1\,000$), czyli wstępną kwotę płaconą poczcie przez wioskę numer i (kwota ta jest oczywiście modyfikowana w opisany na początku zadania sposób). W każdym z kolejnych m wierszy znajdują się po dwie liczby naturalne oddzielone pojedynczym odstępem — oznaczają one numery wiosek, które łączy kolejna droga.*

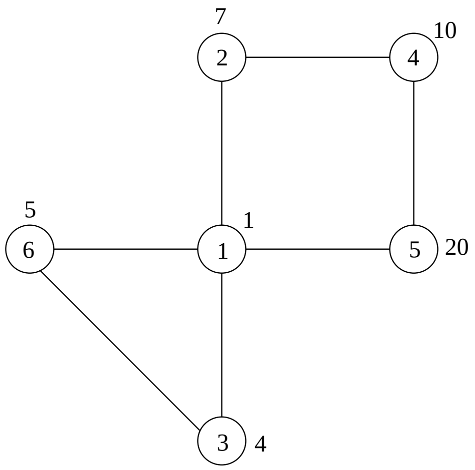
Wyjście

Twój program powinien zapisać jedną dodatnią liczbę naturalną k w pierwszym wierszu pliku tekstowego `pos.out`. *W kolejnym wierszu powinno znaleźć się $k+1$ liczb oznaczających numery wiosek odwiedzanych kolejno przez listonosza w ramach optymalnej trasy.*

Przykład

Dla pliku wejściowego `pos.in`:

```
6 7
1
7
4
10
20
5
2 4
1 5
2 1
4 5
3 6
1 6
1 3
```



poprawną odpowiedzią jest plik wyjściowy `pos.out`*:*

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

Teleports

Wielki Czarodziej Bajtalf stworzył na Bałtyku dwie wyspy: Bornholm i Gotlandię. Na wyspach rozmieścił magiczne teleporty. Teleporty służą do szybkiego "podróżowania" – osoba umieszczona w jednym z teleportów w jednej chwili może się przenieść do innego teleportu. W każdym teleporcie, w trakcie produkcji, wpisuje się identyfikator jego teleportu docelowego, tzn. takiego, do którego może on przenosić "podróżników". Identyfikatora nie można już potem zmienić. Teleporty zostały rozmieszczone tak, by dla każdego teleportu, jego teleport docelowy znajdował się na drugiej wyspie.

Każdy teleport może być nastawiony na:

- *nadawanie — wówczas osoba, która się w nim znajduje zostaje przeniesiona do teleportu docelowego, o ile jest on (teleport docelowy) ustawiony na odbiór (patrz niżej),*
- *odbiór — wówczas może przyjąć podróżnika z innego teleportu.*

Pewnego dnia Wielki Czarodziej Bajtalf nakazał swoim uczniom, by nastawili teleporty tak, aby żaden z nich nie był bezużyteczny, tzn. tak, aby dla każdego teleportu nastawionego na odbiór istniał teleport przenoszący do niego podróżników nastawiony na nadawanie, i na odwrót, dla każdego teleportu nastawionego na nadawanie, jego docelowy teleport był nastawiony na odbiór.

Zadanie

Napisz program, który:

- *wczyta z pliku tekstowego* `tel.in` *opisy teleportów znajdujących się na obu wyspach,*
- *wyznaczy, jak należy nastawić teleporty, by żaden z nich nie był bezużyteczny,*
- *zapisze wynik do pliku tekstowego* `tel.out`.

Jeżeli istnieje wiele rozwiązań, to Twój program powinien wyznaczyć jedno z nich.

Wejście

W pierwszym wierszu pliku tekstowego `tel.in` *znajdują się dwie liczby całkowite m i n, $1 \leq m, n \leq 50\,000$, oddzielone pojedynczym odstępem; m oznacza liczbę teleportów znajdujących się na Bornholmie, a n – liczbę teleportów znajdujących się na Gotlandii. Teleporty na obu wyspach są ponumerowane odpowiednio od 1 do m i od 1 do n. Drugi wiersz pliku wejściowego zawiera m dodatnich liczb całkowitych (nie przekraczających n i oddzielonych pojedynczymi odstępami); k-ta z tych liczb jest numerem teleportu na Gotlandii, który jest teleportem docelowym k-tego teleportu z Bornholmu. Trzeci wiersz zawiera analogiczne dane dla teleportów z Gotlandii, tzn. n dodatnich liczb całkowitych (nie przekraczających m i oddzielonych pojedynczymi odstępami); k-ta z tych liczb jest numerem teleportu na Bornholmie, który jest teleportem docelowym k-tego teleportu z Gotlandii.*

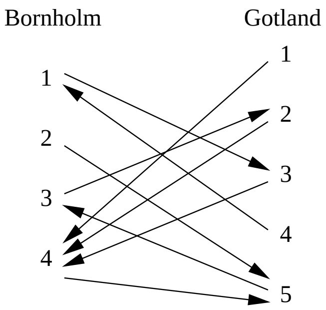
Wyjście

Twój program powinien zapisać w pliku wynikowym `tel.out` *dwa wiersze opisujące, jak należy nastawić teleporty, by żaden z nich nie był bezużyteczny. W pierwszym wierszu powinien znaleźć się opis ustawień teleportów na Bornholmie, a w drugim – opis ustawień teleportów na Gotlandii. Każdy opis, to napis długości równej odpowiednio m i n, złożony z zer lub jedynek. Jeżeli k-ty znak w wierszu jest równy* `1`*, to oznacza, że teleport o numerze k (na danej wyspie) jest ustawiony na nadawanie; jeśli odpowiedni znak jest równy* `0` *– to teleport jest nastawiony na odbiór.*

Przykład

Dla pliku wejściowego `tel.in`*:*

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



poprawną odpowiedzią jest plik wyjściowy `tel.out`*:*

```
0110
10110
```