Piotr Stańczyk **Arkadiusz Paterek**
Treść zadania, Opracowanie Program

Punkty

Dany jest zbiór punktów na płaszczyźnie o współrzędnych całkowitych, który będziemy nazywać **wzorem***, oraz zestaw innych zbiorów punktów na płaszczyźnie (również o współrzędnych całkowitych). Interesuje nas, które z podanych zestawów są podobne do wzoru, tzn. można je tak przekształcić za pomocą obrotów, przesunięć, symetrii osiowej i jednokładności, aby były identyczne ze wzorem. Przykładowo: zbiór punktów* $\{(0,0),(2,0),(2,1)\}$ *jest podobny do zbioru* $\{(6,1),(6,5),(4,5)\}$*, ale nie do zbioru* $\{(4,0),(6,0),(5,-1)\}$*.*

Zadanie

Napisz program, który:

- *wczyta ze standardowego wejścia opisy: wzoru oraz zestawu badanych zbiorów punktów,*
- *wyznaczy, które z badanych zbiorów punktów są podobne do wzoru,*
- *wypisze wynik na standardowe wyjście.*

Wejście

W pierwszym wierszu standardowego wejścia zapisana jest jedna liczba całkowita k $(1 \leqslant k \leqslant 25\,000)$ *— liczba punktów tworzących wzór. W kolejnych* k *wierszach zapisane są pary liczb całkowitych pooddzielanych pojedynczymi odstępami. W* $i+1$*-ym wierszu są współrzędne* i*-tego punktu należącego do wzoru:* x_i *i* y_i $(-20\,000 \leqslant x_i, y_i \leqslant 20\,000)$*. Punkty tworzące wzór są (parami) różne.*

W kolejnym wierszu zapisana jest liczba zbiorów do zbadania n $(1 \leqslant n \leqslant 20)$*. Dalej następuje* n *opisów zbiorów punktów. Opis każdego zbioru rozpoczyna się od wiersza zawierającego jedną liczbę całkowitą* l *— liczbę punktów w danym zbiorze* $(1 \leqslant l \leqslant 25\,000)$*. Punkty te są opisane w kolejnych wierszach, po jednym w wierszu. Opis punktu to dwie liczby całkowite oddzielone pojedynczym odstępem oznaczające jego współrzędne* x *i* y $(-20\,000 \leqslant x, y \leqslant 20\,000)$*. Punkty tworzące jeden zbiór są parami różne.*

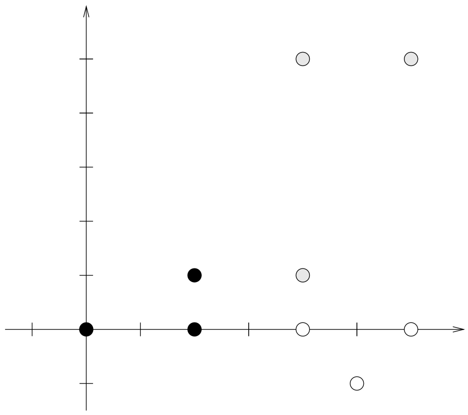
Wyjście

Twój program powinien wypisać na standardowe wyjście n *wierszy — po jednym dla każdego badanego zbioru punktów. Wiersz* i*-ty powinien zawierać słowo* `TAK`*, gdy* i*-ty z podanych zbiorów punktów jest podobny do podanego wzoru, lub słowo* `NIE` *w przeciwnym przypadku.*

Przykład

Dla danych wejściowych:

```
3
0 0
2 0
2 1
2
3
4 1
6 5
4 5
3
4 0
6 0
5 -1
```



poprawnym wynikiem jest:

```
TAK
NIE
```

Rozwiązanie

Analiza problemu

Na wstępie zauważmy, że aby zbiory były podobne, muszą zawierać po tyle samo punktów. Załóżmy więc, że mamy dwa równoliczne zbiory punktów S_1 i S_2. Chcąc sprawdzić, czy są one do siebie podobne, możemy każdy z nich przekształcać za pomocą operacji:

- symetrii osiowej,
- przesunięcia,
- jednokładności (o skali większej od 0),
- obrotu.

Jeżeli w wyniku otrzymamy dwa identyczne zbiory, to mamy twierdzącą odpowiedź na pytanie.

W rozwiązaniu wzorcowym korzystamy z faktu, iż złożenie dowolnego ciągu wymienionych wyżej przekształceń można sprowadzić do złożenia najwyżej czterech operacji — symetrii osiowej, przesunięcia, jednokładności i obrotu. W algorytmie staramy się wyznaczyć te operacje — jeśli nam się to uda, to oznacza, że testowane zbiory punktów są podobne. Co więcej, poszukiwane operacje można wyznaczać niezależnie od siebie. Jest to możliwe dzięki określeniu pewnych niezmienników, które muszą być zachowane przez operacje pozostałe do wykonania.

Zauważmy, że zadanie istotnie upraszcza się, jeżeli znajdziemy parę punktów $P_1 \in S_1$ oraz $P_2 \in S_2$, o których wiemy, że w wyniku podobieństwa P_1 musi przejść na P_2. Wówczas możemy każdy ze zbiorów przesunąć tak, by P_1 i P_2 znalazły się w środku układu współrzędnych. Potem wystarczy sprawdzić, czy za pomocą symetrii osiowej (według osi *OX* lub *OY*), jednokładności (o środku $(0,0)$) i obrotu (również o środku $(0,0)$) można zbiory sprowadzić do tej samej postaci.

Kluczowym dla rozwiązania zadania jest spostrzeżenie, że rolę takich „punktów zaczepienia" mogą odegrać środki ciężkości zbiorów.

Definicja 1 Dla zbioru punktów $S = \{(x_1,y_1),(x_2,y_2),\ldots(x_k,y_k)\}$ *środek ciężkości* to punkt

$$H(S) = (s_x, s_y) = (\frac{x_1+x_2+\ldots+x_k}{k}, \frac{y_1+y_2+\ldots+y_k}{k})$$

Środek ciężkości ma dwie istotne dla nas własności:

- przy wszystkich rozważanych operacjach środek ciężkości zbioru punktów przed przekształceniem przechodzi na środek ciężkości zbioru po przekształceniu, czyli $R(H(S)) = H(R(S))$, gdzie R oznacza dowolną z wymienionych wcześniej operacji zastosowaną do punktu lub zbioru punktów;
- zbiory punktów S_1 i S_2 (równoliczne) są do siebie podobne wtedy i tylko wtedy, gdy zbiory $S_1 \cup \{H(S_1)\}$ i $S_2 \cup \{H(S_2)\}$ (czyli S_1 i S_2 rozszerzone o swoje środki ciężkości) są do siebie podobne.

Pierwszą z powyższych własności można łatwo sprawdzić analizując kolejno każdą z dopuszczalnych operacji. Druga własność wynika z pierwszej. W ciągu operacji $R_1, R_2, \ldots, R_r$ przekształcających zbiór S_1 w S_2, środek ciężkości $H(S_1)$ przechodzi na $H(S_2)$. Stąd ten sam ciąg operacji przeprowadza zbiór $S_1 \cup \{H(S_1)\}$ w $S_2 \cup \{H(S_2)\}$. Z drugiej strony, jeśli $Q_1, Q_2, \ldots, Q_t$ jest ciągiem przekształcającym zbiór $S_1 \cup \{H(S_1)\}$ w $S_2 \cup \{H(S_2)\}$, to musi on (ponownie z pierwszej własności) przekształcić punkt $H(S_1)$ na $H(S_2)$. W takim razie ciąg $Q_1, Q_2, \ldots, Q_t$ zastosowany do pozostałych punktów zbioru, czyli S_1, przeprowadza je na S_2.

Chcąc wykorzystać własności środka ciężkości, na początku każdy ze zbiorów punktów rozszerzymy o jego środek ciężkości. Następnie sprawdzimy podobieństwo zbiorów przekształcając je przez odpowiednie przesunięcie (sprowadzając środki ciężkości do punktu $(0,0)$) i testując, czy stosując jednokładność i obrót można sprowadzić zbiory do tej samej postaci. Analogiczny test przeprowadzamy po przekształceniu jednego ze zbiorów przez symetrię osiową.

Symetria osiowa

Badając, czy do wykazania podobieństwa zbiorów należy zastosować symetrię osiową, rozważymy dwa przypadki. Oddzielnie sprawdzimy podobieństwo zbiorów S_1 i S_2 oraz $O(S_1)$ i S_2, gdzie O jest symetrią według osi *OY*. Zbiory S_1 i S_2 są podobne tylko wówczas, gdy jeden z testów zakończy się wynikiem pozytywnym. Symetrię osiową wykonujemy zamieniając wszystkie współrzędne x punktów zbioru S_1 na wartości przeciwne.

Dalszą część rozważań prowadzimy oddzielnie dla par zbiorów S_1 i S_2 oraz $O(S_1)$ i S_2.

Przesunięcie

Wykonujemy przesunięcie każdego ze zbiorów odpowiednio o wektor $(-a_x, -a_y)$ oraz $(-b_x, -b_y)$, gdzie $H(S_1) = (a_x, a_y)$ oraz $H(S_2) = (b_x, b_y)$. Po tych operacjach zarówno $H(S_1)$, jak i $H(S_2)$ znajdują się w punkcie $(0,0)$.

Jednokładność

Zdefiniujmy *promień* zbioru jako maksymalną odległość pomiędzy środkiem ciężkości zbioru i jednym z punktów zbioru. Oznaczmy promienie zbiorów S_1 i S_2 odpowiednio przez d_1 i d_2. Starając się sprowadzić zbiory do tej samej postaci należy przekształcić je przez jednokładność w ten sposób, by ich promienie były jednakowe. Aby uzyskać promień d, zbiór S_1 należy przeskalować ze współczynnikiem $\frac{d}{d_1}$, a zbiór S_2 ze współczynnikiem $\frac{d}{d_2}$. Wybór d jest dowolny (ale oczywiście $d \neq 0$).

Obrót

Ostatnią fazą jest sprawdzenie, czy dwa odpowiednio przeskalowane zbiory można sprowadzić do tej samej postaci za pomocą obrotu o środku w punkcie $(0,0)$. Problem ten, choć z natury geometryczny, przedstawimy w języku algorytmów tekstowych i rozwiążemy za pomocą algorytmu sprawdzania *cyklicznej równoważności słów*. Dwa słowa są równoważne cyklicznie, jeśli po przestawieniu pewnej liczby początkowych liter pierwszego słowa na jego koniec otrzymujemy drugie słowo. Przykładowo, słowa *aaababa* i *ababaaa* są równoważne cyklicznie, natomiast słowa *aabaaba* i *ababaaa* nie są. Istnieje prosty, choć nieoczywisty, algorytm sprawdzania równoważności cyklicznej dwóch słów w czasie liniowo zależnym od ich długości (patrz na przykład [14]).

Chcąc zastosować wspomniany algorytm, należy przedstawić zbiory punktów w postaci słów w ten sposób, aby słowa reprezentujące zbiory były równoważne cyklicznie wtedy i tylko wtedy, gdy zbiory można sprowadzić do tej samej postaci za pomocą obrotu o środku w punkcie $(0,0)$. Aby utworzyć taką reprezentację, posortujmy punkty zbioru według współrzędnych kątowych (punkty o jednakowych kątach uporządkujmy według odległości od punktu $(0,0)$). Następnie utwórzmy sekwencję liczb całkowitych: $c_1^2, -e_1^2, c_2^2, -e_2^2, \ldots, c_k^2, -e_k^2$, gdzie c_i oznacza odległość i–tego (według porządku kątowego) punktu zbioru od początku układu współrzędnych, natomiast e_i jest odległością pomiędzy i–tym, a $(i+1)$–szym punktem (e_k jest odległością pomiędzy ostatnim i pierwszym punktem). Sekwencję tę nazwiemy *obwódką*.

Zauważmy, że w obwódce punkty zbioru są opisane poprzez wartości, które nie ulegają zmianie podczas obrotu wokół punktu $(0,0)$. W związku z tym reprezentacja zbioru i jego obrazów poprzez takie obroty różni się jedynie punktem początkowym, od którego rozpoczynamy zapis sekwencji. To pozwala nam sformułować następujący lemat.

Lemat 1 *Dwa zbiory punktów można sprowadzić do tej samej postaci za pomocą obrotu o środku* $(0,0)$ *wtedy i tylko wtedy, gdy ich obwódki są słowami cyklicznie równoważnymi.*

W ten sposób ostatni test równoważności zbiorów możemy przeprowadzić w czasie liniowo zależnym od liczby punktów zbioru, stosując wspomniany algorytm testowania cyklicznej równoważności słów.

Rozwiązanie wzorcowe

Rozwiązanie wzorcowe jest realizacją opisanego powyżej schematu. Poszczególne etapy sprawdzania podobieństwa zbiorów są następujące.

1. Sprawdzamy, czy liczba punktów w porównywanych zbiorach jest różna, jeśli tak, to odpowiadamy, że zbiory nie są podobne.

2. Dla każdego ze zbiorów obliczamy jego środek ciężkości — aby zagwarantować, że punkt ten ma współrzędne całkowite (patrz dyskusja poniżej na temat zakresu liczb i dokładności obliczeń) mnożymy współrzędne wszystkich punktów przez liczność zbioru; dodajemy do zbioru jego środek ciężkości, o ile już do niego nie należy (na tym etapie sprawdzamy, czy nie występuje przypadek, w którym jeden zbiór zawiera już swój środek ciężkości, a drugi nie — w takiej sytuacji odpowiadamy, że zbiory nie są podobne).

3. Przesuwamy oba zbiory tak, by ich środki ciężkości znalazły się w punkcie $(0,0)$.

4. Obliczamy promienie zbiorów, po czym przeskalowujemy oba zbiory tak, by miały takie same promienie.

5. Tworzymy obwódki obu zbiorów. W tym celu sortujemy punkty każdego zbioru według współrzędnych kątowych (punkty o jednakowych współrzędnych kątowych — według odległości od punktu $(0,0)$) i obliczamy odpowiednie odległości pomiędzy punktami.

6. Stosujemy algorytm testowania cyklicznej równoważności słów do utworzonych obwódek; jeżeli są sobie równoważne, to odpowiadamy, że zbiory są podobne; w przeciwnym razie odpowiadamy, że nie są podobne.

7. Jeżeli powyższy test wypadł pomyślnie, to kończymy algorytm. W przeciwnym razie przekształcamy jeden ze zbiorów przez symetrię według osi OY i ponawiamy kroki od 2 do 6.

Na czas działania powyższej procedury zasadniczy wpływ ma sortowanie wykonywane w trakcie obliczania obwódki. Wszystkie pozostałe operacje potrafimy wykonać w czasie $O(k)$, zakładając, że do testowania równoważności cyklicznej zastosujemy algorytm liniowy. Stąd złożoność całego programu wykonującego opisany test dla wzoru i n zbiorów punktów wynosi $O(n \cdot k \cdot \log k)$.

Algorytm możemy zaimplementować tak, by wszystkie przeprowadzane obliczenia były całkowitoliczbowe, co gwarantuje ich dokładność. Jednak już pobieżna analiza wykonywanych operacji pokazuje, że współrzędne występujące w obliczeniach mogą być zbyt duże dla arytmetyki 64-bitowej. Początkowo współrzędne punktów należą do przedziału $[-2 \cdot 10^4, 2 \cdot 10^4]$. Podczas obliczania środków ciężkości przemnażamy wszystkie współrzędne przez k, co zwiększa ich zakres do przedziału $[-5 \cdot 10^8, 5 \cdot 10^8]$. Po przeskalowaniu wyrównującym promienie zbiorów zakres współrzędnych można ograniczyć do przedziału $[-25 \cdot 10^{16}, 25 \cdot 10^{16}]$. Kwadraty odległości między punktami występujące w obwódkach są już więc liczbami z przedziału $[0, 10^{36}]$. Wymusza to zaimplementowanie operacji arytmetycznych na liczbach 128-bitowych.

Inne rozwiązania

W zadaniu nie widać sensownej alternatywy dla podstawowej idei algorytmu polegającej na oparciu testu podobieństwa zbiorów na zrównaniu ich środków ciężkości. Mogą natomiast istnieć rozwiązania bez dodatkowej arytmetyki i z mniej efektywną procedurą sprawdzania cyklicznej równoważności słów. Testy dla zadania przechodzi na przykład program reprezentujący zmienne w postaci zmiennoprzecinkowej, w którym po przesunięciu środków ciężkości do punktu $(0,0)$ przeskalowujemy zbiory zmieniając ich promienie tak, by wynosiły 20 000. Co prawda w wykonywane obliczenia wkradają się pewne błędy zaokrągleń, ale są one na tyle małe, że program działa poprawnie na wszystkich przygotowanych testach. Również zastosowanie algorytmu naiwnego — o pesymistycznej złożoności $O(k^2)$ — do sprawdzania cyklicznej równoważności obwódek jest dla testów z zadania wystarczająco szybkie.

Niepoprawne rozwiązania

We wspomnianym wyżej algorytmie naiwnym przerywamy sprawdzanie cyklicznej równoważności obwódek natychmiast, gdy znajdziemy odpowiedź. Przy takim podejściu znaczna część sprawdzeń kończy się po porównaniu kilku elementów. Gdybyśmy zawsze kontynuowali porównywanie słów do końca, to otrzymamy procedurę działającą *zawsze* w czasie $\Theta(k^2)$. Algorytm z taką procedurę działa zbyt długo i nie przechodzi testów przy zadanych ograniczeniach czasowych.

Niepoprawne rozwiązania zadania mogą także wynikać z nieuwzględnienia sytuacji, w których środek ciężkości zbioru punktów należy do tego zbioru. Potencjalnym źródłem błędów są także przypadki, gdy w zbiorze znajduje się kilka punktów o tej samej odległości kątowej względem środka ciężkości.

Testy

Zadanie było sprawdzane na 10 testach. Pięć pierwszych testów to testy poprawnościowe. Pozostałe testy miały na celu sprawdzenie zarówno poprawności rozwiązań zawodników, jak i ich efektywności. Poniżej znajduje się krótki opis każdego z testów, zawierający liczbę punktów tworzących wzór *k* oraz liczbę zbiorów do zbadania *n*.

Nazwa	**k**	**n**
pun1.in	5	8
pun2.in	1	12
pun3.in	3	5
pun4.in	3	4
pun5.in	1 301	3

Nazwa	**k**	**n**
pun6.in	6 400	6
pun7.in	9 999	5
pun8.in	16 000	7
pun9.in	10 000	7
pun10.in	25 000	20