

Żart permutacyjny (A)

Limit pamięci: 32 MB

Limit czasu: 1.00 s

Andrzej ma permutację. A dokładniej to miał, bo żartowniś Jasio zmazał mu spacje pomiędzy liczbami. Pomóż Andrzejowi odtworzyć permutację.

Napisz program, który: wczyta permutację bez spacji, wyznaczy gdzie powinny pojawić się spacje i wypisze wynik na standardowe wyjście.

Wejście

W pierwszym (i jedynym) wierszu wejścia znajduje się ciąg cyfr – permutacja Andrzeja bez spacji.

Wyjście

W pierwszym (i jedynym) wierszu wyjścia powinna się znaleźć permutacja Andrzeja ze spacjami (pojedynczymi odstępami). Jeśli istnieje wiele rozwiązań, wypisz dowolne z nich.

Ograniczenia

Długość ciągu wejściowego nie przekracza 256 znaków.

Przykład

Wejście

4111109876532

Wyjście

4 1 11 10 9 8 7 6 5 3 2

Podział na grupy (B)

Limit pamięci: 512 MB

Limit czasu: 2.00 s



A kto nie trafi do żadnej z grup to zmarznie.

Ach tak, to właśnie początek kolejnego obozu informatycznego przygotowującego spragnionych wiedzy licealistów do Olimpiady Informatycznej. Tym razem odbywa się on w miejscowości Lewin Kłodzki i otacza go zimowa aura napędzana grudniowym, zimnym górskim powietrzem.

Na obozach zwykle bywało tak, że gdy tylko zaczynały się, to Kadra miała w zwyczaju od razu wpadać w rozmaite tarapaty. Tak stało się i tym razem – Kadra nie za bardzo potrafi sobie poradzić z podzieleniem uczestników obozu na odpowiednie grupy.

Na obóz przyjechało N uczestników, każdemu z nich Kadra przypisała poziom umiejętności A_i , zgodnie z liczbą wbitych do tej pory zadań na serwisie Solve. Uczestników obozu należy podzielić na K grup, w taki sposób, aby różnica poziomu umiejętności dowolnych dwóch uczestników obozu z tej samej grupy różniła się o nie więcej niż D . Niestety liczba grup K jest ograniczona przez liczbę sal na terenie ośrodka, a liczba D jest uwarunkowana wybranymi na obóz zadaniami, może się więc okazać, że nie wszystkich uczestników obozu uda się przypisać do jakiegokolwiek grupy. I tutaj właśnie pojawia się problem Kadry: chcieliby podzielić uczestników na grupy w taki sposób, aby jak najwięcej z nich trafiło do jakiegokolwiek grupy.

Twoim zadaniem będzie napisanie programu, który na podstawie listy poziomów umiejętności uczestników, maksymalnej liczby grup i maksymalnej różnicy poziomów umiejętności wyznaczy maksymalną liczbę osób, które można podzielić na grupy.

Wejście

W pierwszym wierszu wejścia znajduje się trzy liczby naturalne N , D i K , pooddzielane pojedynczymi odstępami i oznaczające odpowiednio liczbę uczestników obozu, maksymalną różnicę poziomów umiejętności oraz maksymalną liczbę grup. W drugim wierszu wejścia znajduje się ciąg N liczb naturalnych A_i , pooddzielanych pojedynczymi odstępami i oznaczającymi poziomy umiejętności kolejnych uczestników.

Wyjście

W pierwszym wierszu wyjścia powinna się znaleźć jedna liczba naturalna – maksymalna liczba uczestników, którą można podzielić na grupy zgodnie z zasadami opisanymi w treści zadania.

Ograniczenia

$$1 \leq N \leq 500\,000, 1 \leq D, A_i \leq 10^9, 1 \leq K \leq 10.$$

Podzadanie	Warunki	Punkty
1	$K = 1$	20
2	$K = 2$	20
3	$N \leq 20, K \leq 5$	20
4	$N \leq 1000$	20
5	brak dodatkowych ograniczeń	20

Przykład

Wejście

5 1 2
6 1 2 4 6

Wyjście

4

Wyjaśnienie

Uczestnicy z poziomami umiejętności 1 i 2 mogą być w jednej grupie, a dwójka uczestników z poziomami umiejętności 6 w drugiej grupie.

Wejście

2 2 3
3 1

Wyjście

2

Wyjaśnienie

Uczestnicy mogą być razem w grupie, a dwie mogą zostać niewykorzystane.

Wejście

5 1 2
6 1 2 3 6

Wyjście

4

Wyjaśnienie

Uczestnicy z poziomami umiejętności 2 i 3 mogą być w jednej grupie, a dwójka uczestników z poziomami umiejętności 6 w drugiej grupie.

Zgadywanie permutacji (c)

Limit pamięci: 128 MB

Limit czasu: 5.00 s

To zadanie jest interaktywne. Po wypisaniu każdego wiersza należy opróżnić bufor. W C++ możesz użyć `cout « flush`, w Java – `System.out.flush()`, w Python – `sys.stdout.flush()`. Należy ściśle trzymać się protokołu opisanego w rozdziale *Protokół interakcji* – niewczytanie wiadomości wysłanej przez interaktor skutkuje werdyktem *wrong answer*.

Ukryta jest pewna permutacja liczb od 1 do n . Można, dla dowolnych $1 \leq l \leq r \leq n$ oraz $1 \leq x \leq n$ zadawać pytania postaci: czy pomiędzy l -tym a r -tym elementem (włącznie) tej permutacji jest pewien element podzielny przez x ? Można zadać co najwyżej Q zapytań. W odpowiedzi należy wypisać tę ukrytą permutację.

Interakcja

Na początku należy wczytać 1 liczbę całkowitą n , oznaczającą długość permutacji. Po zadaniu zapytania postaci $? \ l \ r \ x$, należy wczytać liczbę całkowitą, 0, jeżeli odpowiedź na to zapytanie jest negatywna oraz 1 w przeciwnym wypadku.

Na końcu należy wypisać znalezioną permutację a w formacie: `! a_1 a_2 ... a_n`.

Podzadania

We wszystkich podzadaniach zachodzi: $1 \leq n \leq 10\,000$, $Q = 500\,000$.

T	Warunki	Punkty
1	$n = 10$	5
2	$n = 1000$	15
3	Gradacja względem liczby zadanych pytań.	80

Dla podzadania 3 gradacja wygląda w następujący sposób: niech z oznacza liczbę zadanych przez program pytań, wtedy procent punktów przyznanych za dany test wynosi:

- 100, jeżeli $z \leq 135\,000$,
- $40 + 60 \cdot \frac{250\,000 - z}{250\,000 - 135\,000}$, gdy $135\,000 < z \leq 250\,000$
- $40 \cdot \frac{500\,000 - z}{500\,000 - 250\,000}$, gdy $250\,000 < z \leq 500\,000$
- 0 gdy $z > 500\,000$.

Przykładowa interakcja

Wejście Wyjście

```
5
? 5 5 5
1
? 2 3 4
1
! 3 1 4 2 5
```

Narzędzia do testowania lokalnego

Zostaną udostępnione testy przykładowe i programy **zgasoc.cpp**, **run.py** oraz przykładowe błędne rozwiązanie o poprawnym schemacie komunikacji **zga.cpp**. Żeby urochomić rozwiązanie na teście należy użyć komendy:

```
python3 run.py 1 [plik wykonywalny z zgasoc.cpp] [plik wykonywalny z rozwiązaniem]
```

[test]

Na przykład: `python3 run.py 1 ./kulsoc ./kul 0a`

Listowa reprezentacja grafu (G1)

Limit pamięci: 32 MB

Limit czasu: 2.00 s

Dany jest graf nieskierowany. Wypisz jego reprezentację listową.

Wejście

W pierwszym wierszu dane są dwie liczby: N , M , gdzie N oznacza liczbę wierzchołków zaś M – liczbę krawędzi. W następnych M wierszach podane są po dwie liczby: A , B oznaczające krawędź między wierzchołkami A i B .

Wyjście

W i -tym wierszu należy wypisać najpierw " i :", a dalej numery wierzchołków połączonych krawędzią z i -tym wierzchołkiem. Numery wierzchołków należy podać w porządku rosnącym, oddzielając je pojedynczą spacją.

Ograniczenia

$1 \leq N \leq 10^5$, $0 \leq M \leq 10^5$, $1 \leq A, B \leq N$.

Przykład

Wejście

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

Wyjście

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

Macierzowa reprezentacja grafu (G2)

Limit pamięci: 32 MB

Limit czasu: 2.00 s

Dany jest graf nieskierowany. Wypisz jego reprezentację macierzową.

Wejście

W pierwszym wierszu dane są dwie liczby: N , M , gdzie N oznacza liczbę wierzchołków zaś M – liczbę krawędzi. W następnych M wierszach podane są po dwie liczby: A , B oznaczające krawędź między wierzchołkami A i B .

Wyjście

W N wierszach powinno znaleźć się N liczb oddzielonych spacjami – macierz sąsiedztwa danego grafu. W macierzy tej na pozycji i , j powinna znaleźć się 1, jeśli istnieje krawędź między wierzchołkami i oraz j . W przeciwnym wypadku należy wypisać 0.

Ograniczenia

$1 \leq N \leq 10^3$, $0 \leq M \leq 10^5$, $1 \leq A, B \leq N$.

Przykład

Wejście

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

Wyjście

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

Spójne składowe (G3)

Limit pamięci: 128 MB

Limit czasu: 2.00 s

Zadanie jest krótkie. Masz dany graf nieskierowany o N wierzchołkach i M krawędziach. Policz ile ma spójnych składowych.

Wejście

W pierwszym wierszu wejścia znajdują się dwie liczby naturalne: N , M pooddzielane pojedynczymi odstępami i określające odpowiednio: liczbę wierzchołków i liczbę krawędzi rozpatrywanego grafu. W kolejnych M wierszach znajduje się opis krawędzi grafu. W i -tym wierszu znajdują się dwie liczby całkowite A_i i B_i , oddzielone pojedynczym odstępem, oznaczające, że wierzchołek A_i jest połączony krawędzią z wierzchołkiem B_i . Wszystkie krawędzie w grafie są nieskierowane, wierzchołki numerujemy od 1 do N . Żadna krawędź podana na wejściu nie powtarza się, w grafie nie ma też pętli.

Wyjście

W pierwszej i jedynej linii standardowego wyjścia powinna pojawić się jedna liczba całkowita oznaczająca liczbę spójnych składowych rozpatrywanego grafu.

Ograniczenia

$1 \leq N \leq 1\,000\,000$, $1 \leq M \leq 1\,000\,000$, $1 \leq A_i, B_i \leq N$. W testach wartych łącznie 50% maksymalnej punktacji: $N, M \leq 1000$.

Przykład

Wejście

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

Wyjście

```
2
```


Ucieczka (G4)

Limit pamięci: 32 MB

Limit czasu: 0.50 s

Dla danego opisu labiryntu odpowiedz, czy istnieje droga do wyjścia, gdzie cyfra 1 oznacza przeszkodę, 0 dowolne pole, a cyfra 2 oznacza wyjście z labiryntu, czyli nasz cel.

Startujemy zawsze w lewym górnym rogu, możemy poruszać się jedynie do góry, w prawo, lewo i w dół i nie możemy wejść na pole przeszkody lub wyjść poza labirynt.

Wejście

W pierwszej linii wejścia dane są W i K , oznaczające odpowiednio liczbę wierszy i liczbę kolumn labiryntu. W kolejnych W wierszach dane jest po K liczb pooddzielanych spacjami opisujących labirynt.

Wyjście

Wypisz TAK jeśli istnieje wyjście z labiryntu, albo NIE w przeciwnym przypadku.

Ograniczenia

$$1 \leq W, K \leq 100$$

Przykład

Wejście

```
3 11
0 0 0 1 0 0 1 1 0 0 0
0 0 0 1 0 0 0 0 0 1 0
0 0 0 0 0 0 1 0 0 0 2
```

Wyjście

```
TAK
```

Liczba Erdosa (G5)

Limit pamięci: 64 MB

Limit czasu: 4.00 s

Paul Erdos był jednym z najwybitniejszych matematyków XX w. Był znany nie tylko ze swoich niesamowitych zdolności do rozwiązywania problemów matematycznych, ale także ze swojego specyficznego stylu bycia i poczucia humoru. Jednym z żartów Erdosa, który utrwalił się jako stały element „matematycznego folkloru” jest tzw. **liczba Erdosa**.

Definiujemy ją w następujący sposób: Erdos w swojej własnej osobie ma liczbę Erdosa równą 0. Następnie wszyscy Ci, którzy kiedykolwiek napisali z nim jakąś pracę mają liczbę Erdosa równą 1. Wszyscy Ci, którzy napisali z nimi pracę, ale nie napisali pracy z Erdosem, mają liczbę Erdosa równą 2, itd. Jeśli zdarzy się tak, że dany matematyk nie napisał pracy ani z Erdosem, ani z nikim, komu można przypisać pewną dodatnią liczbę Erdosa, to przyjmujemy, że jego liczba Erdosa wynosi -1 . Można powiedzieć, że w ten sposób definiujemy odległość od Erdosa do innych naukowców.

No dobrze, historyjka ładna, ale pewnie chciałbyś już coś zaimplementować (w końcu po coś jest ten Solve). W takim razie napisz program, który dostając listę prac napisanych przez matematyków, dla każdego z nich obliczy jaka jest jego liczba Erdosa.

Wejście

W pierwszym wierszu wejścia znajdują się trzy liczby naturalne: N , M oraz E pooddzielane pojedynczymi odstępami i określające odpowiednio: liczbę rozpatrywanych matematyków oraz liczbę napisanych prac oraz numer matematyka, który jest Erdosem (wszystkich rozpatrywanych matematyków numerujemy od 1 do N). W kolejnych M wierszach znajduje się opis napisanych prac. W i -tym wierszu znajdują się dwie liczby całkowite A_i i B_i , oddzielone pojedynczym odstępem, oznaczające, że matematyk o numerze A_i napisał pracę wspólnie z matematykiem B_i . Żadna para podana na wejściu nie powtarza się, matematyk nie może również napisać pracy sam ze sobą.

Wyjście

Twój program powinien wypisać dokładnie N linii. W i -tej linii wyjścia powinna znaleźć się liczba Erdosa i -tego matematyka.

Ograniczenia

$1 \leq N \leq 1\,000\,000$, $1 \leq M \leq 1\,000\,000$, $1 \leq A_i, B_i \leq N$.

W testach wartych łącznie 50% maksymalnej punktacji: $N, M \leq 1000$.

Przykład

Wejście

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

Wyjście

```
1
0
1
-1
```