**Problem: Store & Search String**

**Description**

A database contains a sequence of key k1, k2, ..., kn which are strings (1<=n<=100000). Perform a sequence of actions of two kinds:

· find k: find and return 1 if k exists in the database, and return 0, otherwise

· insert k: insert a key k into the database and return 1 if the insertion is successful (k does not exist in the database) and return 0 if the insertion is failed (k exists in the database)

Note that the length of any key is greater than 0 and less than or equal to 50.

**Input**

Two blocks of information. The first block contains a key of (k1,k2,...,kn) in each line. The first block is terminated with a line containing \*. The second block is a sequence of actions of two finds described above: each line contains 2 string: cmd and k in which cmd = find or insert and k is the key (parameter of the action). The second block is terminated with a line containing \*\*\*. Note that the number of actions can be up to 100000.

**Output**

Each line contains the result (0 or 1) of the corresponding action.

**Example**

**Input**

computer

university

school

technology

phone

\*

find school

find book

insert book

find algorithm

find book

insert book

\*\*\*

**Output**

1

0

1

0

1

0

**Problem: Parentheses**

**Description**

Given a string containing only characters (, ), [, ] {, }. Write a program that check whether the string is correct in expression.

Example:

([]{()}()[]): correct

([]{()]()[]): incorrect

**Input**

One line contains the string (the length of the string is less than or equal to 10^6)

**Output**

Write 1 if the sequence is correct, and write 0, otherwise

**Example:**

**input**

(()[][]{}){}{}[][]({[]()})

**output**

1

**Problem: Connected Components of undirected graphs**

**Description**

Given a undirected graph G=(V,E) where V={1,. . .,N} is the number of nodes and the set E has M edges. Compute the number of connected components of G.

**Input**

* Line 1: two positive integers N and M (1 <= N <= 105, 1 <= M <= 105)
* Line i+1 (i=1, . . .,M): contains two positive integers u and v which are endpoints of ith edge

**Output**

Write the number of connected components of G.

**Example**

**Input**

8 8

1 2

1 7

1 8

2 7

4 5

4 6

5 6

7 8

**Output**

3

**Problem: A Taxi Routing**

**Description**

There are n passengers 1, 2, …, n. The passenger i want to travel from point i to point i + n (i = 1,2,…,n). There is a taxi located at point 0 for transporting the passengers. Given the distance matrix c(2n+1)\*(2n+1) in which c(i,j) is the traveling distance from point i to point j (i, j = 0,1,…,2n) Compute the shortest route for the taxi, serving n passengers and coming back to point 0 such that at any moment, there is no more than one passenger in the taxi, and no point is visited more than once (except for point 0, which can be visited up to twice).

**Input**

* Line 1: contains n (1 ≤ n ≤ 11).
* Line i+1 (i = 1, 2,…, 2n+1) contains the ith line of the matrix c.

**Output**

* Unique line contains the length of the shortest route.

**Example**

**Input**

2

0 8 5 1 10

5 0 9 3 5

6 6 0 8 2

2 6 3 0 7

2 5 3 4 0

**Output**

17

**Problem: Make Span Schedule**

**Description**

A project has n tasks 1,. . ., n. Task i has duration d(i) to be completed (i=1,. . ., n). There are precedence constraints between tasks represented by a set Q of pairs: for each (i,j) in Q, task j cannot be started before the completion of task i. Compute the earliest completion time of the project.

**Input**

* Line 1: contains n and m (1 <= n <= 104, 1 <= m <= 200000)
* Line 2: contains d(1),. . ., d(n) (1 <= d(i) <= 1000)
* Line i+3 (i=1,. . ., m) : contains i and j : task j cannot be started to execute before the completion of task i

**Output**

* Write the earliest completion time of the project.

**Example**

**Input**

9 13

5 3 1 2 6 4 3 1 4

1 3

1 5

1 6

2 1

2 3

3 5

4 1

4 2

4 6

5 8

7 9

9 5

9 8

**Output**

18

**Problem: Shortest Path between 2 nodes on a directed graph with non-negative weights**

**Description**

Given a directed graph G = (V,E) in which V = {1,2,...,n) is the set of nodes. Each arc (u,v) has a non-negative weight w(u,v). Given two nodes s and t of G. Find the shortest path from s to t on G.

**Input**

* Line 1: contains two integers n and m which are the number of nodes and the number of arcs of G (1 <= n <= 100000)
* Line i + 1(i = 1,2,...,m): contains 3 integers u, v, w in which w is the weight of arc(u,v) (0 <= w <= 100000)
* Line m+2: contains two integers s and t

**Output**

Write the weight of the shortest path found or write -1 if no path from s to t was found

**Example**

**Input**

5 7

2 5 87

1 2 97

4 5 78

3 1 72

1 4 19

2 3 63

5 1 18

1 5

**Output**

97

**Problem: Bus inter-city**

**Description**

Có n thành phố 1, 2, ..., n. Giữa 2 thành phố i và j có thể có 1 con đường (2 chiều) kết nối giữa chúng.

Mỗi thành phố i có tuyến buýt i với C[i] là giá vé mỗi khi lên xe và D[i] là số thành phố tối đa mà buýt i có thể đi đến trên 1 hành trình đi qua các con đường kết nối.

Hãy tìm cách đi từ thành phố 1 đến thành phố n với số tiền phải trả là ít nhất

**Input**

* Dòng 1: chứa 2 số nguyên dương n và m trong đó n là số thành phố và m là số con đường kết nối các thành phố (1 <= n <= 5000, 1 <= m <= 10000)
* Dòng i+1 (i = 1,2,...,n): chứa 2 số nguyên dương C[i] và D[i] (1 <= C[i] <= 10000, 1 <= D[i] <= 100)
* Dòng n+1+i (i = 1, 2, ..., m): chứa 2 số nguyên dương i và j trong đó giữa thành phố i và j có con đường kết nối

**Output**

* Số tiền tối thiểu phải bỏ ra để đi buýt từ thành phố 1 đến thành phố n

**Example**

**Input**

6 6

10 2

30 1

50 1

20 3

30 1

20 1

1 2

1 3

1 5

2 4

2 5

4 6

**Output**

30

**Giải thích:**

-Lên buýt 1 từ thành phố 1 đến thành phố 4 mất 10 đồng

-Lên buýt 4 từ thành phố 4 đến thành phố 6 mất 20 đồng

Tổng cộng mất 10 + 20 = 30 đồng

**Problem: Minimum Spanning Tree - Kruskal**

**Description**

Given a undirected connected graph G=(V,E) where V={1,…,N}. Each edge (u,v)∈E(u,v)∈E has weight w(u,v)w(u,v). Compute minimum spanning tree of G.

**Input**

* Line 1: N and M (1≤N,M≤105) in which NN is the number of nodes and MM is the number of edges.
* Line i+1 (i=1,…,M): contains 3 positive integers u, v, and w where w is the weight of edge (u,v)

**Output**

Write the weight of the minimum spanning tree found.

**Example**

**Input**

5 8 1 2 1 1 3 4 1 5 1 2 4 2 2 5 1 3 4 3 3 5 3 4 5 2

**Output**

7

**Problem: Code generation and standardization in DBMS**

**Description**

In DataBase Management Systems, we need to generate the keys for records created. In many cases, a key is generated by generating a non-negative integer and is standardized under a string with fixed length L (by filling characters 0 at the beginning of the integer util the length is equal to L). For example, if the fixed-length L is 5, the key generated from 123 is the string 00123.

Given a positive integer n and L, and a sequence of integers a1, a2, ..., an. Write a program for generating corresponding keys k1, k2, ..., kn.

**Input**

* Line 1: contains n and L (1 <= n <= 1000000, 2 <= L <= 50)
* Line i+1 (i = 1,...,n): contains ai (1 <= ai <= 2000000), ai < 10L

**Output**

* Line i (i = 1,..., n): contains the key ki

**Example**

**Input**

5 6

54

39

40

78

1

**Output**

000054

000039

000040

000078

000001

**Problem: Max Matching on Bipartie Graph**

**Description**

There are $n$ tasks $1,\dots,n$ and $m$ staffs $1,\dots,m$. $T(i)$ is the set of staffs that can perform the task $i$ ($i=1\dots,n$). Compute an assignment of staffs to tasks such that each task is assigned to at most one staff and each staff cannot be assigned to more than one task and the number of tasks assigned is maximal.

**Input**

* Line 1: contains 2 positive integer $n$ và $m$ ($1\leq n,m\leq 10000$)
* Line $i+1$ ($i=1,\dots,n$) contains a positive integer $k$ and $k$ integer of $T(i)$

**Output**

* Write the maximum number of tasks that are assigned to staffs.

**Example**

**Input**

3 4

2 1 4

2 1 3

1 2

**Output**

3

**Problem: Check TSP with Precedence Constraint**

**Description**

A shipper visits all points 1, 2, 3, ..., n (each point is visited exactly once) for delivering packages and comes back to the starting point. There is a precedence constraint between points on the route of shipper which is represented by a list of m pairs (i,j): point i must be located before point j on the route (i, j = 1, 2, ..., n). A route of shipper is feasible if it is a permutation of 1, 2, ..., n and satisfies the precedence constraint. The travel distance from point i to point j is d(i,j) (i,j = 1,...,n). Write a program that checks if a sequence x1, x2, ..., xn is a feasible route of the shipper and computes the total travel distance of that route.

**Input**

* Line 1: contains a positive integer n (1 <= n <= 1000)
* Line 2: contains n positive integers x1, x2, ..., xn
* Line i + 2 (i = 1,...,n): contains the ith row of the distance matrix d
* Line n+3: contains a positive integer m (1 <= m <= 1000000)
* Line k + n + 3 (k = 1,...,m): contains two positive integers i and j (1 <= i, j <= n): point i must be located before point j on the route

**Output**

* Write the total travel distance of the route x1, ..., xn if it is feasible, or write -1 if the sequence x1, ..., xn is not feasible.

**Example**

**Input**

4

1 2 3 4

0 2 4 3

3 0 1 1

2 3 0 5

1 3 2 0

3

1 2

3 4

1 4

**Output**

9

**Input**

4

4 2 3 1

0 2 4 3

3 0 1 1

2 3 0 5

1 3 2 0

3

1 2

3 4

1 4

**Output**

-1

**Problem: CBUS**

**Description**

There are n passengers 1, 2, …, n. The passenger i want to travel from point i to point i + n (i = 1,2,…,n). There is a bus located at point 0 and has k places for transporting the passengers (it means at any time, there are at most k passengers on the bus). You are given the distance matrix c in which c(i,j) is the traveling distance from point i to point j (i, j = 0,1,…, 2n). Compute the shortest route for the bus, serving n passengers and coming back to point 0.

**Input**

* Line 1 contains n and k (1≤n≤11,1≤k≤10)
* Line i+1 (i=1,2,…,2n+1) contains the (i−1)th line of the matrix c (rows and columns are indexed from 0,1,2,..,2n).

**Output**

* Unique line contains the length of the shortest route.

**Example**

**Input**

3 2 0 8 5 1 10 5 9 9 0 5 6 6 2 8 2 2 0 3 8 7 2 5 3 4 0 3 2 7 9 6 8 7 0 9 10 3 8 10 6 5 0 2 3 4 4 5 2 2 0

**Output**

25

**Problem: Starting date-time plus duration equal to Finishing date-time**

**Description**

Given a sequence of n tasks 1, 2, ..., n. The task i has starting time is ti which is under the format YYYY-MM-DD hh:mm:ss (for example: 2021-06-03 10:34:21) and the duration Di (in seconds). Compute the finising time of n tasks.

**Input**

* The first block is the sequence of continuous days, each day is in a line under the format YYYY-MM-DD. The first block is terminated by a line containing the \* character
* The second block contains n lines, each line i (i = 1,...,n) contains ti and Di

**Output**

Each line i (i = 1, ..., n) contains the finishing time of the task i under the format YYYY-MM-DD hh:mm:ss

**Example**

**Input**

2020-12-29

2020-12-30

2020-12-31

2021-01-01

2021-01-02

2021-01-03

\*

2020-12-30 21:23:02 1

2020-12-30 21:33:02 86400

2020-12-30 21:33:02 172800

\*\*\*

**Output**

2020-12-30 21:23:03

2020-12-31 21:33:02

2021-01-01 21:33:02

**Problem: Gen k-combinations of 1 2 - n**

**Description**

Given 2 positive integers k and n. Write a program tat generates all k-combinations of 1, 2, ..., n in a lexicographic order.

**Example**

**Input**

2 4

**Output**

1 2

1 3

1 4

2 3

2 4

3 4

**Problem: Warehouse**

**Description**

A truck is planned to arrive at some stations among N stations 1, 2, . . ., N located on a line. Station i (i = 1,…,N) has coordinate i and has following information

* ai : amount of goods
* ti : pickup time duration for taking goods

The route of the truck is a sequence of stations x1 < x2 < . . . < xk (1 ≤ xj ≤ N, j = 1,…, k). Due to technical constraints, the distance between two consecutive stations that the truck arrives xi and xi+1 is less than or equal to D and the total pickup time duration cannot exceed T. Find a route for the truck such that total amount of goods picked up is maximal.

**Input**

* Line 1: N, T, D (1 <= N <= 1000, 1 <= T <= 100, 1 <= D <= 10)
* Line 2: a1,. . ., aN (1 <= ai <= 10)
* Line 3: t1, . . ., tN (1 <= ti <= 10)

**Output**

* Write the total amount of goods that the truck picks up in the route.

**Example**

**Input**

6 6 2

6 8 5 10 11 6

1 2 2 3 3 2

**Output**

24

**Problem: Telco data check & query**

**Description**

Write a C program to perform some queries on a telco data (comming from stdin) with the following format:

The first block of data consists of lines (terminated by a line containing #), each line (number of lines can be up to 100000) is under the form:

call <from\_number> <to\_number> <date> <from\_time> <end\_time>

which is a call from the phone number <from\_number> to a phone number <to\_number> on <date>, and starting at time-point <from\_time>, terminating at time-point <end\_time>

* <from\_number> and <to\_number> are string of 10 characters (a phone number is correct if it contains only digits 0,1,...,9, otherwise, the phone number is incorrect)
* <date> is under the form YYYY-MM-DD (for example 2022-10-21)
* <from\_time> and <to\_time> are under the form hh:mm:ss (for example, 10:07:23)

The second block consists of queries (terminated by a line containing #), each query in a line (number of lines can be up to 100000) and belongs to one of the following types:

* ?check\_phone\_number: print to stdout (in a new line) value 1 if no phone number is incorrect
* ?number\_calls\_from <phone\_number>: print to stdout (in a new line) the number of times a call is made from <phone\_number>
* ?number\_total\_calls: print to stdout (in a new line) the total number of calls of the data
* ?count\_time\_calls\_from <phone\_number>: print to stdout (in a new line) the total time duration (in seconds) the calls are made from <phone\_number>

**Example**

**Input**

call 0912345678 0132465789 2022-07-12 10:30:23 10:32:00

call 0912345678 0945324545 2022-07-13 11:30:10 11:35:11

call 0132465789 0945324545 2022-07-13 11:30:23 11:32:23

call 0945324545 0912345678 2022-07-13 07:30:23 07:48:30

#

?check\_phone\_number

?number\_calls\_from 0912345678

?number\_total\_calls

?count\_time\_calls\_from 0912345678

?count\_time\_calls\_from 0132465789

#

**Output**

1

2

4

398

120

**Problem: Check Bipartite Graph**

**Description**

Given an undirected graph G=(V, E) in which V={1,…, N} is the set of nodes and |E|=M. You are required to write a program to check if G is a bipartite graph.

**Input**

* Line 1 N and M (1≤N, M≤105)
* Line i+1 (i=1,…, M): u and v which are endpoints of the ith edge

**Output**

Write 1 if G is a bipartite graph and 0, otherwise.

**Example**

**input**

6 6

1 2

1 3

2 5

2 6

4 5

4 6

**output**

1

**Problem: Max-Distance Sub-Sequence**

**Description**

Given N elements (2≤ N ≤100,000) on a straight line at positions x1,…, xN (0≤ xi ≤1,000,000,000).

The distance of a subset of N elements is defined to be the minimum distance between two elements.

Find the subset of N given elements containing exactly C elements such that the distance is maximal.

**Input**

* The first line contains a positive integer T (1 <= T <= 20) which is the number of test cases.
* Subsequent lines are T test cases with the following format:
* Line 1: Two space-separated integers: N and C
* Lines 2: contains x1, x2, . . . , xN

**Output**

For each test case output one integer: the distance of the subset found.

**Example**

**input**

1

5 3

1

2

8

4

9

**output**

3

**Explain**: Jonh can put his 3 cows in the stalls at positions 1, 4 and 8, resulting in a minimum distance of 3.

**Problem: Largest black subrectangle**

**Description**

Một hình chữ nhật kích thước n x m được chia thành các ô vuông con 1 x 1 với 2 màu đen hoặc trắng. Hình chữ nhật được biểu diễn bởi ma trận A(n x m) trong đó A(i, j) = 1 có nghĩa ô hàng i, cột j là ô đen và A(i, j) = 0 có nghĩa ô vuông hàng i cột j là ô trắng.

Hãy xác định hình chữ nhật con của bảng đã cho bao gồm toàn ô đen và có diện tích lớn nhất.

*Dữ liệu*

· Dòng 1: chứa số nguyên dương n và m (1 <= n,m <= 1000)

· Dòng i+1 (i = 1,…, n): chứa hàng thứ i của ma trận A

*Kết quả*

· Ghi ra diện tích của hình chữ nhật lớn nhất tìm được

Ví dụ

*Dữ liệu*

4 4

0 1 1 1

1 1 1 0

1 1 0 0

1 1 1 0

*Kết quả*

6

**Problem: Sum pair of sequence equal to a number**

**Description**

Cho dãy a1, a2, ..., an trong đó các phần tử đôi một khác nhau và 1 giá trị nguyên dương M. Hãy đếm số Q các cặp (i,j) sao cho 1 <= i < j <= n và ai + aj = M.

**Dữ liệu**

* Dòng 1: ghi n và M (1 <= n, M <= 1000000)
* Dòng 2: ghi a1, a2, ..., an

**Kết quả**

Ghi ra giá trị Q

**Ví dụ**

**Dữ liệu**

5 6

5 2 1 4 3

**Kết quả**

2