Codeforces Beta Round #33 (Codeforces format)

A. What is for dinner?

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

In one little known, but very beautiful country called Waterland, lives a lovely shark Valerie. Like all the sharks, she has several rows of teeth, and feeds on crucians. One of Valerie's distinguishing features is that while eating one crucian she uses only one row of her teeth, the rest of the teeth are "relaxing".

For a long time our heroine had been searching the sea for crucians, but a great misfortune happened. Her teeth started to ache, and she had to see the local dentist, lobster Ashot. As a professional, Ashot quickly relieved Valerie from her toothache. Moreover, he managed to determine the cause of Valerie's developing caries (for what he was later nicknamed Cap).

It turned that Valerie eats too many crucians. To help Valerie avoid further reoccurrence of toothache, Ashot found for each Valerie's tooth its residual viability. Residual viability of a tooth is a value equal to the amount of crucians that Valerie can eat with this tooth. Every time Valerie eats a crucian, viability of all the teeth used for it will decrease by one. When the viability of at least one tooth becomes negative, the shark will have to see the dentist again.

Unhappy, Valerie came back home, where a portion of crucians was waiting for her. For sure, the shark couldn't say no to her favourite meal, but she had no desire to go back to the dentist. That's why she decided to eat the maximum amount of crucians from the portion but so that the viability of no tooth becomes negative.

As Valerie is not good at mathematics, she asked you to help her to find out the total amount of crucians that she can consume for dinner.

We should remind you that while eating one crucian Valerie uses exactly one row of teeth and the viability of each tooth from this row decreases by one.

Input

The first line contains three integers n, m, k ($1 \le m \le n \le 1000$, $0 \le k \le 10^6$) — total amount of Valerie's teeth, amount of tooth rows and amount of crucians in Valerie's portion for dinner. Then follow n lines, each containing two integers: r ($1 \le r \le m$) — index of the row, where belongs the corresponding tooth, and c ($0 \le c \le 10^6$) — its residual viability.

It's guaranteed that each tooth row has positive amount of teeth.

Output

In the first line output the maximum amount of crucians that Valerie can consume for dinner.

Sample test(s)

input
4 3 18 2 3 1 2 3 6 2 3
output
11

input
2 2 13 1 13 2 12
output
13

B. String Problem

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Boy Valera likes strings. And even more he likes them, when they are identical. That's why in his spare time Valera plays the following game. He takes any two strings, consisting of lower case Latin letters, and tries to make them identical. According to the game rules, with each move Valera can change **one** arbitrary character A_i in one of the strings into arbitrary character B_i, but he has to pay for every move a particular sum of money, equal to W_i. He is allowed to make as many moves as he needs. Since Valera is a very economical boy and never wastes his money, he asked you, an experienced programmer, to help him answer the question: what minimum amount of money should Valera have to get identical strings.

Input

The first input line contains two initial non-empty strings s and t, consisting of lower case Latin letters. The length of each string doesn't exceed 10^5. The following line contains integer n ($0 \le n \le 500$) — amount of possible changings. Then follow n lines, each containing characters A_i and B_i (lower case Latin letters) and integer W_i ($0 \le W_i \le 100$), saying that it's allowed to change character A_i into character B_i in any of the strings and spend sum of money W_i.

Output

If the answer exists, output the answer to the problem, and the resulting string. Otherwise output `-1` in the only line. If the answer is not unique, output any.

Sample test(s)

input

```
uayd
uxxd
3
a x 8
x y 13
d c 3
```

output

```
21
uxyd
```

input

```
a
b
3
a b 2
a b 3
b a 5
```

output

```
2
b
```

input

```
abc
ab
6
a b 4
a b 7
b a 8
c b 11
c a 3
a c 0
```

output

```
-1
```

C. Wonderful Randomized Sum

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Learn, learn and learn again — Valera has to do this every day. He is studying at mathematical school, where math is the main discipline. The mathematics teacher loves her discipline very much and tries to cultivate this love in children. That's why she always gives her students large and difficult homework. Despite that Valera is one of the best students, he failed to manage with the new homework. That's why he asks for your help. He has the following task. A sequence of n numbers is given. A prefix of a sequence is the part of the sequence (possibly empty), taken from the start of the sequence. A suffix of a sequence is the part of the sequence (possibly empty), taken from the end of the sequence. It is allowed to sequentially make two operations with the sequence. The first operation is to take some prefix of the sequence and multiply all numbers in this prefix by -1. The second operation is to take some suffix and multiply all numbers in it by -1. The chosen prefix and suffix may intersect. What is the maximum total sum of the sequence that can be obtained by applying the described operations?

Input

The first line contains integer n ($1 \le n \le 10^5$) — amount of elements in the sequence. The second line contains n integers a_i ($-10^4 \le a_i \le 10^4$) — the sequence itself.

Output

The first and the only line of the output should contain the answer to the problem.

Sample test(s)

input
3 -1 -2 -3
output
6

input
5 -4 2 0 5 0
output
11

input
5 -1 10 -5 10 -2
output
18

D. Knights

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Berland is facing dark times again. The army of evil lord Van de Mart is going to conquer the whole kingdom. To the council of war called by the Berland's king Valery the Severe came n knights. After long discussions it became clear that the kingdom has exactly n control points (if the enemy conquers at least one of these points, the war is lost) and each knight will occupy one of these points.

Berland is divided into $m + 1$ regions with m fences, and the only way to get from one region to another is to climb over the fence. Each fence is a circle on a plane, no two fences have common points, and no control point is on the fence. You are given k pairs of numbers a_i, b_i. For each pair you have to find out: how many fences a knight from control point with index a_i has to climb over to reach control point b_i (in case when Van de Mart attacks control point b_i first). As each knight rides a horse (it is very difficult to throw a horse over a fence), you are to find out for each pair the minimum amount of fences to climb over.

Input

The first input line contains three integers n, m, k ($1 \le n, m \le 1000$, $0 \le k \le 100000$). Then follow n lines, each containing two integers Kx_i, Ky_i ($-10^9 \le Kx_i, Ky_i \le 10^9$) — coordinates of control point with index i. Control points can coincide.

Each of the following m lines describes fence with index i with three integers r_i, Cx_i, Cy_i ($1 \le r_i \le 10^9$, $-10^9 \le Cx_i, Cy_i \le 10^9$) — radius and center of the circle where the corresponding fence is situated.

Then follow k pairs of integers a_i, b_i ($1 \le a_i, b_i \le n$), each in a separate line — requests that you have to answer. a_i and b_i can coincide.

Output

Output exactly k lines, each containing one integer — the answer to the corresponding request.

Sample test(s)

input

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

output

```
1
```

input

```
2 3 1
0 0
4 4
1 0 0
2 0 0
3 0 0
1 2
```

output

```
3
```

E. Helper

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

It's unbelievable, but an exam period has started at the OhWord University. It's even more unbelievable, that Valera got all the tests before the exam period for excellent work during the term. As now he's free, he wants to earn money by solving problems for his groupmates. He's made a $list$ of subjects that he can help with. Having spoken with n of his groupmates, Valera found out the following information about them: what subject each of them passes, time of the exam and sum of money that each person is ready to pay for Valera's help.

Having this data, Valera's decided to draw up a timetable, according to which he will solve problems for his groupmates. For sure, Valera can't solve problems round the clock, that's why he's found for himself an optimum order of day and plans to stick to it during the whole exam period. Valera assigned time segments for sleep, breakfast, lunch and dinner. The rest of the time he can work.

Obviously, Valera can help a student with some subject, only if this subject is on the $list$. It happened, that all the students, to whom Valera spoke, have different, but one-type problems, that's why Valera can solve any problem of subject $list_i$ in t_i minutes.

Moreover, if Valera starts working at some problem, he can break off only for sleep or meals, but he can't start a new problem, not having finished the current one. Having solved the problem, Valera can send it instantly to the corresponding student via the Internet.

If this student's exam hasn't started yet, he can make a crib, use it to pass the exam successfully, and pay Valera the promised sum. Since Valera has little time, he asks you to write a program that finds the order of solving problems, which can bring Valera maximum profit.

Input

The first line contains integers m, n, k ($1 \le m, n \le 100$, $1 \le k \le 30$) — amount of subjects on the $list$, amount of Valera's potential employers and the duration of the exam period in days.

The following m lines contain the names of subjects $list_i$ ($list_i$ is a non-empty string of at most 32 characters, consisting of lower case Latin letters). It's guaranteed that no two subjects are the same.

The $(m+2)$-th line contains m integers t_i ($1 \le t_i \le 1000$) — time in minutes that Valera spends to solve problems of the i-th subject. Then follow four lines, containing time segments for sleep, breakfast, lunch and dinner correspondingly.

Each line is in format `H1:M1-H2:M2`, where $00 \le$ `H1`, `H2` ≤ 23, $00 \le$ `M1`, `M2` ≤ 59. Time `H1:M1` stands for the first minute of some Valera's action, and time `H2:M2` stands for the last minute of this action. No two time segments cross. It's guaranteed that Valera goes to bed before midnight, gets up earlier than he has breakfast, finishes his breakfast before lunch, finishes his lunch before dinner, and finishes his dinner before midnight. All these actions last less than a day, but not less than one minute. Time of the beginning and time of the ending of each action are within one and the same day. But it's possible that Valera has no time for solving problems.

Then follow n lines, each containing the description of students. For each student the following is known: his exam subject s_i (s_i is a non-empty string of at most 32 characters, consisting of lower case Latin letters), index of the exam day d_i ($1 \le d_i \le k$), the exam time $time_i$, and sum of money c_i ($0 \le c_i \le 10^6$, c_i — integer) that he's ready to pay for Valera's help. Exam time $time_i$ is in the format `HH:MM`, where $00 \le$ `HH` ≤ 23, $00 \le$ `MM` ≤ 59. Valera will get money, if he finishes to solve the problem strictly before the corresponding student's exam begins.

Output

In the first line output the maximum profit that Valera can get. The second line should contain number p — amount of problems that Valera is to solve. In the following p lines output the order of solving problems in chronological order in the following format: index of a student, to whom Valera is to help; index of the time, when Valera should start the problem; time, when Valera should start the problem (the first minute of his work); index of the day, when Valera should finish the problem; time, when Valera should finish the problem (the last minute of his work). To understand the output format better, study the sample tests.

Sample test(s)

input

```
3 3 4
calculus
algebra
history
58 23 15
00:00-08:15
08:20-08:35
09:30-10:25
19:00-19:45
calculus 1 09:36 100
english 4 21:15 5000
history 1 19:50 50
```

output

```
150
2
1 1 08:16 1 09:29
3 1 10:26 1 10:40
```

input

```
2 2 1
matan
```

```
codeforces
1 2
00:00-08:00
09:00-09:00
12:00-12:00
18:00-18:00
codeforces 1 08:04 2
matan 1 08:02 1
```

output

```
3
2
2 1 08:01 1 08:01
1 1 08:02 1 08:03
```

input

```
2 2 1
matan
codeforces
2 2
00:00-08:00
09:00-09:00
12:00-12:00
18:00-18:00
codeforces 1 08:04 2
matan 1 08:03 1
```

output

```
2
1
1 1 08:01 1 08:02
```