Problems overview

Problem	BUSES	SOLDIERS	ROADS
Program name	BUSES.EXE	SOLDIERS.EXE	ROADS.EXE
Source name	BUSES.PAS BUSES.C BUSES.CPP	SOLDIERS.PAS SOLDIERS.C SOLDIERS.CPP	ROADS.PAS ROADS.C ROADS.CPP
Input file	BUSES.IN	SOLDIERS.IN	ROADS.IN
Output file	BUSES.OUT	SOLDIERS.OUT	ROADS.OUT
Time limit per test	10 seconds	10 seconds	10 seconds
Number of tests	10	10	10
Points per test	3	3	4
Total points	**30**	**30**	**40**

The maximum total score for Day 2 is 100 points.

BUSES

Problem description

Description

A particular one-way street has a bus stop every kilometre. A passenger is charged according to the number of kilometres they ride on the bus. An example list of fares is shown in the table below.

Kms	1	2	3	4	5	6	7	8	9	10
Price	12	21	31	40	49	58	69	79	90	101

No bus travels for more than 10 kilometres. A passenger wants to travel n ($1 \le n \le 100$) kilometres. If the passenger can ride any number of times to complete the trip, which selection of rides will ensure that the total price is a minimum.

Input

The input file **BUSES.IN** has as its first line the 10 integers which are the fares for the 10 trips (i.e. the fare to travel 1, 2, 3 ... 10 kilometres). The integers are separated from each other by a single blank space. Each fare will be in the range 1 to 500 inclusive. The next line contains n, ($1 \le n \le 100$), the number of kilometres to be travelled.
Note that the fares do not necessarily make economic sense. For example, it is possible that the fare for 10 kilometres is cheaper than that for 1 kilometre.

Output

The output file **BUSES.OUT** contains a single integer, the cost of the cheapest way to travel the n kilometres.

Example 1

Input : BUSES.IN

```
12 21 31 40 49 58 69 79 90 101
15
```

Output: BUSES.OUT

```
147
```

Note that the cheapest cost is 147. This could be achieved using 3 bus rides of 3, 6 and 6 kilometres or alternatively 3 rides of 5 kilometres each.

Example 2

Input : BUSES.IN

```
12 20 30 40 25 60 70 80 90 11
21
```

Output: BUSES.OUT

```
34
```

Again by making 3 bus trips (of 1, 10 and 10 kilometres) we travel the required 21 kilometres at a cost of 12+11+11 = 34.

SOLDIERS

Problem description

N soldiers of the land Gridland are randomly scattered around the country.
A position in Gridland is given by a pair (x, y) of integer coordinates. Soldiers can move - in one move, one soldier can go one unit up, down, left or right (hence, he can change either his x or his y coordinate by 1 or -1).
The soldiers want to get into a horizontal line next to each other (so that their final positions are (x, y), (x+1, y), ..., (x+N-1, y), for some x and y). Integers x and y, as well as the final order of soldiers along the horizontal line is arbitrary.
The goal is to minimise the total number of moves of all the soldiers that takes them into such configuration. Two or more soldiers must never occupy the same position at the same time.

Input data

The first line of the input file **SOLDIERS.IN** contains the integer N, $1 \le N \le 10000$, the number of soldiers. The following N lines of the input file contain initial positions of the soldiers: for each i, $1 \le i \le N$, the $(i+1)^{st}$ line of the input file contains a pair of integers x[i] and y[i] separated by a single blank character, representing the coordinates of the i^{th} soldier, $-10000 \le x[i], y[i] \le 10000$.

Output data

The first and the only line of the output file **SOLDIERS.OUT** should contain the minimum total number of moves that takes the soldiers into a horizontal line next to each other.

Examples

SOLDIERS.IN

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

SOLDIERS.OUT

```
4
```

SOLDIERS.IN

```
5
1 2
2 2
1 3
3 -2
3 3
```

SOLDIERS.OUT

8

ROADS

Problem description

N cities named with numbers 1 ... N are connected with one-way roads. Each road has two parameters associated with it: the road length and the toll that needs to be paid for the road (expressed in the number of coins).
Bob and Alice used to live in the city 1. After noticing that Alice was cheating in the card game they liked to play, Bob broke up with her and decided to move away - to the city N. He wants to get there as quickly as possible, but he is short on cash.
We want to help Bob to find **the shortest path** from the city 1 to the city N **that he can afford** with the amount of money he has.

Input data

The first line of the input file **ROADS.IN** contains the integer K, $0 \le K \le 10000$, maximum number of coins that Bob can spend on his way.
The second line contains the integer N, $2 \le N \le 100$, the total number of cities.
The third line contains the integer R, $1 \le R \le 10000$, the total number of roads.
Each of the following R lines describes one road by specifying integers S, D, L and T separated by single blank characters :
S is the source city, $1 \le S \le N$
D is the destination city, $1 \le D \le N$
L is the road length, $1 \le L \le 100$
T is the toll (expressed in the number of coins), $0 \le T \le 100$
Notice that different roads may have the same source and destination cities.

Output data

The first and the only line of the output file **ROADS.OUT** should contain the total length of the shortest path from the city 1 to the city N whose total toll is less than or equal K coins.
If such path does not exist, only number -1 should be written to the output file.

Examples

ROADS.IN

```
5
6
7
1 2 2 3
2 4 3 3
3 4 2 4
1 3 4 1
4 6 2 1
3 5 2 0
5 4 3 2
```

ROADS.OUT

```
11
```

ROADS.IN

```
0
4
4
1 4 5 2
1 2 1 0
2 3 1 1
3 4 1 0
```

ROADS.OUT

```
-1
```