Gas Station

Michael the Explorer is on holiday. He now wants to drive his car from his home to Singapore. To reach Singapore, Michael needs to stop at **N** cities along the way. From city `i`, Michael can only go to city `i+1`. In each city, he can fill the fuel of his car in the gas station. If Michael decides to refill gas at that city, he will always fill his car until full. However, each city has different cost of filling fuel.

Michael's car has a fuel capacity of 200 units. He will start his journey with a full-tank. To simplify this problem, if the distance of two cities is X, Michael needs X units of fuel to travel between the two cities.

Your job is to tell Michael the minimum cost he can spend on fuel for his journey. You also need to tell Michael if his journey is futile: i.e. if Michael cannot reach the destination, no matter how often he fills the fuel of his car.

Input

The first line contains an integer **N (0 <= N <= 22)**, denoting the number of cities in between Michael's home and Singapore.

N rows follow.

Each row consists of two integers, **X (1 <= X <= 500)** and **P (1 <= P <= 10,000)**, denoting the distance from the previous city to that city and the price (per unit) of filling the fuel at that city.

After that, one row consisting of a single integer follows. This integer denotes the distance from the last city to Singapore.

Output

Print the minimum cost of refueling that Michael can spend. If it is impossible for Michael to reach the destination, print "can meh?" (without the quotes). Your output should contain a newline character.

Sample Input 1

```
3
150 100
50 1000
30 100
20
```

Sample Output 1

```
15000
```

Sample Input 2

```
0
500
```

Sample Output 2

```
can meh?
```

Explanation

In the first sample input, you can go to the first city and have 50 units of fuel left. Now, technically, you can still reach the next city without refueling. But then, you will spend more on gas in the next city since you need to refuel. Your best option is to refuel at the first city (for 150 units, at the price of 100 per unit) so that you will have enough fuel for the rest off your trip.

In the second sample input, Michael cannot reach his destination.

Skeleton

You are given the skeleton file **GasStation.java**. You should see the contents below when you open the file, otherwise you might be working in the wrong directory.

```
/**
 * Name          :
 * Matric No.    :
 * PLab Acct.    :
 */

public class GasStation {

    public void run() {
        // treat this as your "main" method
    }

    public static void main(String[] args) {
        GasStation gasStation = new GasStation();
        gasStation.run();
    }
}
```

Notes

1. You should develop your program in the subdirectory **ex2** and use the skeleton file provided.
2. You must use **recursion** to solve this problem. Otherwise, you will get 0.
3. Please be reminded that the marking scheme is:

   Input : 10%
   Output : 10%
   Correctness : 50%

   Programming Style : 30% (awarded if you score **at least 20% from the above**):

   - Meaningful comments (pre- and post- conditions, comments inside the code): 10%
   - Modularity (modular programming, proper modifiers [public / private]): 10%
   - Proper Indentation: 5%
   - Meaningful Identifiers (for both method and variable names): 5%

   Compilation Error : Deduction of **50% of the total marks obtained**.