



Manage the check-in queue

by nikouni

Problem

Submissions

Leaderboard

Discussions

Empowering people to experience the world doesn't just mean providing somewhere to stay! We need you to act as the Chief Queue Manager for the B.Awesome Airline at PlanetBooking Airport. Are you up for it?

The airline has **N** check-in desks (c_1, c_2, \dots, c_N). Each of them has a member of check-in staff c_i who has a speed of s_i , meaning they can process s_i customers in one minute.

When you start your shift, check-in hasn't opened yet, so currently there are no staff processing customers. But there are some already waiting...

There are x_i customers waiting in a queue at a desk to be checked in by a member of staff c_i . There are also **M** customers waiting to be assigned to a check-in desk. Your job is to assign each of these **M** customers to a check-in desk in a way that minimizes the total time needed to process all customers (the ones that are already assigned to a member of check-in staff, plus the **M** ones that are not yet assigned).

You are not allowed to reassign any customer that is already waiting in a queue.

You're always looking for ways to improve the customer experience of flying with B.Awesome Airline, so we need you to return the minimum number of **whole** minutes needed to process all customers.

Input Format

The first line contains two integer **N** and **M** indicating the number of check-in desks and the number of customers waiting unassigned. Each of the next **N** lines contains two integers s_i and x_i specifying the speed and number of customers already waiting for each member of check-in staff.

Constraints

- $1 \leq N \leq 1000$
- $0 \leq M \leq 10^9$
- $1 \leq s_i \leq 1000$
- $0 \leq x_i \leq 10000$

Output Format

A single line with the minimum number of whole minutes needed to process all customers

Sample Input 0

```
3 10
3 8
2 4
1 5
```

Sample Output 0

```
5
```

Sample Input 1

```
1 0
1000 1001
```

Sample Output 1

```
2
```

[f](#) [t](#) [in](#)Contest ends in **20 hours**Submissions: **278**



Max Score: 100



Difficulty: Medium

Rate This Challenge:

☆☆☆☆☆

[More](#)

Current Buffer (saved locally, editable)  

Java 8  

```
1 import java.io.*;
2 import java.math.*;
3 import java.security.*;
4 import java.text.*;
5 import java.util.*;
6 import java.util.concurrent.*;
7 import java.util.function.*;
8 import java.util.regex.*;
9 import java.util.stream.*;
10 import static java.util.stream.Collectors.joining;
11 import static java.util.stream.Collectors.toList;
12
13 public class Solution {
14
15     // Complete the solve function below.
16     static int solve(int m, List<List<Integer>> desks) {
17
18     }
19
20
21     public static void main(String[] args) throws IOException {
22         BufferedReader bufferedReader = new BufferedReader(new InputStreamReader(System.in));
23         BufferedWriter bufferedWriter = new BufferedWriter(new FileWriter(System.getenv("OUTPUT_PATH")));
24
25         String[] nm = bufferedReader.readLine().replaceAll("\\s+$", "").split(" ");
26
27         int n = Integer.parseInt(nm[0]);
28
29         int m = Integer.parseInt(nm[1]);
30
31         List<List<Integer>> desks = new ArrayList<>();
32
33         IntStream.range(0, n).forEach(i -> {
34             try {
35                 desks.add(
36                     Stream.of(bufferedReader.readLine().replaceAll("\\s+$", "").split(" "))
37                         .map(Integer::parseInt)
38                         .collect(toList())
39                 );
40             } catch (IOException ex) {
41                 throw new RuntimeException(ex);
42             }
43         });
44
45         int result = solve(m, desks);
46
47         bufferedWriter.write(String.valueOf(result));
48         bufferedWriter.newLine();
49
50         bufferedReader.close();
51         bufferedWriter.close();
52     }
53 }
54
```

Line: 1 Col: 1