Minimum Time Required ☆

Your Minimum Time Required submission got 35.00 points. Share Tweet

Try the next challenge

Problem Submissions Leaderboard Editorial

You are planning production for an order. You have a number of machines that each have a fixed number of days to produce an item. Given that all the machines operate simultaneously, determine the minimum number of days to produce the required order.

For example, you have to produce $goal = 10$ items. You have three machines that take $machines = [2, 3, 2]$ days to produce an item. The following is a schedule of items produced:

```
Day Production  Count
2   2           2
3   1           3
4   2           5
6   3           8
8   2          10
```

It takes 8 days to produce 10 items using these machines.

Function Description

Complete the minimumTime function in the editor below. It should return an integer representing the minimum number of days required to complete the order.

minimumTime has the following parameter(s):

- machines: an array of integers representing days to produce one item per machine
- goal: an integer, the number of items required to complete the order

Input Format

The first line consist of two integers n and $goal$, the size of $machines$ and the target production.
The next line contains n space-separated integers, $machines[i]$.

Constraints

- $1 \leq n \leq 10^5$
- $1 \leq goal \leq 10^9$
- $1 \leq machines[i] \leq 10^9$

Output Format

Return the minimum time required to produce $goal$ items considering all machines work simultaneously.

Sample Input 0

```
2 5
2 3
```

Sample Output 0

```
6
```

Explanation 0

In 6 days $machine_0$ can produce 3 items and $machine_1$ can produce 2 items. This totals up to 5.

Sample Input 1

```
3 10
1 3 4
```

Sample Output 1

```
7
```

Explanation 1

In 7 minutes, $machine_0$ can produce 7 items, $machine_1$ can produce 2 items and $machine_2$ can produce 1 item, which totals up to 10.

Sample Input 2

```
3 12
4 5 6
```

Sample Output 2

```
20
```

Explanation 2

In 20 days $machine[0]$ can produce 5 items, $machine[1]$ can produce 4, and $machine[2]$ can produce 3.

Change Theme C++

```
7   long compute(vector<long> machines, long goal, long min_days, long max_days){
8
9       if(min_days == max_days)
10          return min_days;
11      //Binary search algo: assume on mid if production is more than needed then we look
    in low to mid (mid_new =low+mid/2) and vice versa.
12      long mid = (min_days + max_days)/2;
13      long prod = 0;
14
15      for(int i =0; i< machines.size();i++) // calculating amount of items made by
    machines if working till day=mid
16          prod+=floor(mid/machines[i]);
17
18      if(prod >= goal)
19          return compute(machines,goal, min_days, mid);
20      else
21          return compute(machines, goal, mid+1, max_days);
22  }
23  // Complete the minTime function below.
24  long minTime(vector<long> machines, long goal) { // using binary search
25      long n = machines.size();
26      sort(machines.begin(), machines.end());
27      // sorts machine from fastest to slowest in production speed
28      long min_days = (machines[0]*goal)/n; // we can get min/best production time when
    all n machines are fastest (machine[0]).
29      long max_days = (machines[0]*goal); // NOTE: (think a little) the upper bound is
    limited due to the fastest machine not slowest. Assume if fastest machine can alone
```

```
    complete goal in x days, in those x days other slower machines produces something. This
    implies goal was actually achieved before the xth day. Therefore, we can say that upper
    bound is single fastest machine.
30
31      if(machines.size()==1) // if there is only 1 machine, it will time*goal= max_days
32          return max_days;
33
34      return compute(machines, goal, min_days, max_days);
35  }
```

Line: 22 Col: 2

Upload Code as File | Test against custom input | Run Code | Submit Code

Congratulations

You solved this challenge. Would you like to challenge your friends?

Next Challenge

- Test case 0
- Test case 1
- Test case 2
- Test case 3
- Test case 4
- Test case 5
- Test case 6

Compiler Message

Success

Input (stdin) Download

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

Expected Output Download

```
1  6
```