Survivors

Problem | Submissions | Discussions

Time Limit: 2s, Memory Limit: 512MB

Problem Statement

You have a garden with N plants. Initially, all plants have dryness level 0. Each plant i has a dryness rate r[i], meaning its dryness increases by r[i] every hour. If any plant's dryness reaches T or more, the garden dies.

You have a robot with an infinite water tank. Each hour, the following happens **in order**:

1. All plants' dryness increases by their respective rates
2. The robot receives exactly **1 unit of water**
3. The robot can choose to water some plants (each plant needs 1 unit of water to reset its dryness to 0)

The robot can store water across hours. For example, if the robot doesn't use water in hour 1, it will have 2 units available in hour 2.

Find the maximum number of complete hours the garden can survive with optimal watering strategy. It is guaranteed that the garden cannot survive indefinitely.

Input Format

- The first line contains two integers N and T.
- The second line contains N integers r[1], r[2], ..., r[N].

Constraints

- $2 \leq N \leq 10^5$
- $1 \leq T \leq min(N, 10^3)$
- $1 \leq r_i \leq 10^3$

Output Format

- Print the maximum number of complete hours the garden can survive.

Sample Input 0

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

Sample Output 0

```
2
```

Explanation 0

Hour 1: Dryness increase:[1, 2, 1, 1, 3], Available water = 1 Unit, The robot waters the plant with dryness 3. Dryness after watering:[1, 2, 1, 1, 0], All plants are still below the threshold.

Hour 2: Dryness after increase:[2, 4, 2, 2, 3], Available water = 1 Unit, The robot waters the plant with dryness 4. Dryness after watering:[2, 0, 2, 2, 3], All plants are still below the threshold.

Hour 3: Dryness after increase:[3, 2, 3, 3, 6], The fifth plant reaches dryness 6, which is greater than T. The garden dies immediately before watering can help.

So,The garden survives for 2 complete hours.

C++

```cpp
#include <map>
#include <set>
#include <list>
#include <cmath>
#include <ctime>
#include <deque>
#include <queue>
#include <stack>
#include <string>
#include <bitset>
#include <cstdio>
#include <limits>
#include <vector>
#include <climits>
#include <cstring>
#include <cstdlib>
#include <fstream>
#include <numeric>
#include <sstream>
#include <iostream>
#include <algorithm>
#include <unordered_map>

using namespace std;
int main() {
    /* Enter your code here. Read input from STDIN. Print output to STDOUT */
    return 0;
}
```

Line: 1 Col: 1

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