Codeforces Beta Round #10

Problema E: *Greedy Change*

Prof. Edson Alves – UnB/FGA

Problema

Billy investigates the question of applying greedy algorithm to different spheres of life. At the moment he is studying the application of greedy algorithm to the problem about change. There is an amount of n coins of different face values, and the coins of each value are not limited in number. The task is to collect the sum x with the minimum amount of coins. Greedy algorithm with each its step takes the coin of the highest face value, not exceeding x. Obviously, if among the coins' face values exists the face value 1, any sum x can be collected with the help of greedy algorithm.

Problema

However, greedy algorithm does not always give the optimal representation of the sum, i.e. the representation with the minimum amount of coins. For example, if there are face values $\{1, 3, 4\}$ and it is asked to collect the sum 6, greedy algorithm will represent the sum as $4 + 1 + 1$, while the optimal representation is $3 + 3$, containing one coin less. By the given set of face values find out if there exist such a sum x that greedy algorithm will collect in a non-optimal way. If such a sum exists, find out the smallest of these sums.

Entrada e saída

Input

The first line contains an integer n ($1 \leq n \leq 400$) – the amount of the coins' face values. The second line contains n integers a_i ($1 \leq a_i \leq 10^9$), describing the face values. It is guaranteed that $a_1 > a_2 > \ldots > a_n$ and $a_n = 1$.

Output

If greedy algorithm collects any sum in an optimal way, output -1. Otherwise output the smallest sum that greedy algorithm collects in a non-optimal way.

Exemplo de entradas e saídas

Sample Input	Sample Output
5 25 10 5 2 1	-1
3 4 3 1	6

Solução $O(N^3)$

- O problema consiste em identificar o menor dentre os contraexemplos, caso a base seja não-canônica
- O algoritmo de Pearson resolve justamente este problema, com complexidade $O(N^3)$
- Em linhas gerais ele avalia $O(N^2)$ candidatos à menor contraexemplo, checando cada um deles em $O(N)$
- Se nenhum deles se confirmar, a base é canônica e a resposta será -1
- Dentre os candidatos que se confirmarem como contraexemplos, o menor deles será a solução do problema

Solução $O(N^3)$

```cpp
#include <bits/stdc++.h>

using namespace std;

const int oo { 2000000007 };

vector<int> greedy(int x, int N, const vector<int>& xs)
{
    vector<int> res(N, 0);

    for (int i = 0; i < N; ++i)
    {
        auto q = x / xs[i];
        x -=  q*xs[i];

        res[i] = q;
    }

    return res;
}
```

Solução $O(N^3)$

```
22 int value(const vector<int>& M, int N, const vector<int>& xs)
23 {
24     int res = 0;
25
26     for (int i = 0; i < N; ++i)
27         res += M[i]*xs[i];
28
29     return res;
30 }
31
32 int solve(int N, const vector<int>& xs)
33 {
34     if (N <= 2)
35         return -1;
36
37     int ans = oo;
38
39     for (int i = N - 2; i >= 0; --i) {
40         auto g = greedy(xs[i] - 1, N, xs);
41         vector<int> M(N, 0);
```

Solução $O(N^3)$

```
43          for (int j = 0; j < N; ++j)
44          {
45              M[j] = g[j] + 1;
46              auto w = value(M, N, xs);
47              auto G = greedy(w, N, xs);
48
49              auto x = accumulate(M.begin(), M.end(), 0);
50              auto y = accumulate(G.begin(), G.end(), 0);
51
52              if (x < y)
53                  ans = min(ans, w);
54
55              M[j]--;
56          }
57      }
58
59      return ans == oo ? -1 : ans;
60  }
```

Solução $O(N^3)$

```
62 int main()
63 {
64     ios::sync_with_stdio(false);
65
66     int N;
67     cin >> N;
68
69     vector<int> xs(N);
70
71     for (int i = 0; i < N; ++i)
72         cin >> xs[i];
73
74     auto ans = solve(N, xs);
75
76     cout << ans << '\n';
77
78     return 0;
79 }
```