SPOJ SUBSUMS

Subset Sums

Prof. Edson Alves – UnB/FGA

Problema

Given a sequence of N ($1 \leq N \leq 34$) numbers $S_1, \ldots, S_N$ ($-20,000,000 \leq S_i \leq 20,000,000$), determine how many subsets of S (including the empty one) have a sum between A and B ($-500,000,000 \leq A \leq B \leq 500,000,000$), inclusive.

Entrada e saída

Input

The first line of standard input contains the three integers N, A, and B. The following N lines contain S_1 through S_N, in order.

Output

Print a single integer to standard output representing the number of subsets satisfying the above property. Note that the answer may overflow a 32-bit integer.

Exemplo de entradas e saídas

Sample Input

```
3 -1 2
1
-2
3
```

Sample Output

```
5
```

Solução $O(2^{N/2} \log N)$

- No pior caso, há 2^{34} subconjuntos a serem avaliados, de modo que uma solução que olhe todos eles individualmente resultará em um TLE
- A técnica do encontro no meio pode ser usada para dividir a entrada em dois grupos de aproximadamente $N/2$ elementos
- Para cada um destes grupos, é preciso computar as somas dos elementos de seus subconjuntos (listas S_1 e S_2, respectivamente)
- Para cada elemento $s \in S_1$, é preciso identificar todos os elementos $r \in S_2$ tais que $A \leq s + r \leq B$
- Se S_2 estiver ordenado, este intervalo de valores pode ser computado por meio de duas buscas binárias, ou através das funções `lower_bound` e `upper_bound` da STL do C++

Solução $O(2^{N/2} \log N)$

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

using namespace std;
using ll = long long;

vector<ll> subset_sum(const vector<ll>& xs) {
    vector<ll> s;

    for (size_t i = 0; i < (1ul << xs.size()); ++i) {
        ll sum = 0;

        for (size_t j = 0; j < xs.size(); ++j)
            if ((1 << j) & i)
                sum += xs[j];

       s.push_back(sum);
    }

    return s;
}
```

Solução $O(2^{N/2} \log N)$

```
ll solve(ll N, ll A, ll B, const vector<ll>& xs)
{
    vector<ll> g1(xs.begin(), xs.begin() + N/2);
    vector<ll> g2(xs.begin() + N/2, xs.end());

    auto s1 = subset_sum(g1), s2 = subset_sum(g2);
    sort(s2.begin(), s2.end());

    ll ans = 0;

    for (auto s : s1)
    {
        auto it = lower_bound(s2.begin(), s2.end(), A - s);
        auto jt = upper_bound(s2.begin(), s2.end(), B - s);
        ans += (jt - it);
    }

    return ans;
}
```

Solução $O(2^{N/2} \log N)$

```
42 int main()
43 {
44     ios::sync_with_stdio(false);
45
46     ll N, A, B;
47     cin >> N >> A >> B;
48
49     vector<ll> xs(N);
50
51     for (ll i = 0; i < N; ++i)
52         cin >> xs[i];
53
54     cout << solve(N, A, B, xs) << endl;
55
56     return 0;
57 }
```