АНАЛИЗ НА РЕШЕНИЕТО НА ЗАДАЧА ПРОИЗВЕДЕНИЯ

Без ограничение може да считаме, че $a_1 \le a_2 \le \ldots \le a_k$. Равенството $(a_1+1)(a_2+1)\ldots(a_k+1) = na_1a_2\ldots a_k$ може да се запише във вида:

$$\left(1+\frac{1}{a_1}\right)\left(1+\frac{1}{a_2}\right)\ldots\left(1+\frac{1}{a_k}\right) = n.$$

Тъй като

$$n = \left(1+\frac{1}{a_1}\right)\left(1+\frac{1}{a_2}\right)\ldots\left(1+\frac{1}{a_k}\right) \le \underbrace{\left(1+\frac{1}{1}\right)\left(1+\frac{1}{1}\right)\ldots\left(1+\frac{1}{1}\right)}_{k \text{ пъти}} = 2^k,$$

то $k \ge \log_2 n$. Нека k е първото цяло число, което не е по-малко от $\log_2 n$. За тази стойност на k проверяваме дали има цели положителни числа a_1, a_2, …, a_k, за които $a_1 \le a_2 \le \ldots \le a_k$ и $(a_1+1)(a_2+1)\ldots(a_k+1) = na_1a_2\ldots a_k$. Ако такива числа съществуват, то търсената най-малка стойност на k е намерена. В противен случай увеличаваме стойността на k с 1 до намирането на първото k, за което съществуват числа, изпълняващи горните условия.

Остава да уточним как при фиксирана стойност на k може да проверим дали съществуват цели положителни числа a_1, a_2, …, a_k, за които $a_1 \le a_2 \le \ldots \le a_k$ и $(a_1+1)(a_2+1)\ldots(a_k+1) = na_1a_2\ldots a_k$. Проверката може да се направи чрез пълно изчерпване. Очевидно $a_1 \ge 1$. Горна граница за a_1 може да се получи от следното неравенство:

$$n = \left(1+\frac{1}{a_1}\right)\left(1+\frac{1}{a_2}\right)\ldots\left(1+\frac{1}{a_k}\right) \le \underbrace{\left(1+\frac{1}{a_1}\right)\left(1+\frac{1}{a_1}\right)\ldots\left(1+\frac{1}{a_1}\right)}_{k \text{ пъти}},$$

което може да се запише във вида

$$\underbrace{(1+a_1)(1+a_1)\ldots(1+a_1)}_{k \text{ пъти}} \ge n\underbrace{a_1a_1\ldots a_1}_{k \text{ пъти}}.$$

От него се получава ограничението $a_1 \le \dfrac{1}{\sqrt[k]{n}-1}$. Аналогично могат да се получат долна и горна граници за a_i (при $i = 2,3\ldots,k-1$). Нека са известни стойностите на a_1, a_2, …, a_{i-1}. Ясно е, че $a_i \ge a_{i-1}$ (долна граница за a_i). Горна граница за a_i се получава от неравенството:

$$n = \left(1+\frac{1}{a_1}\right)\left(1+\frac{1}{a_2}\right)\left(1+\frac{1}{a_{i-1}}\right)\left(1+\frac{1}{a_i}\right)\ldots\left(1+\frac{1}{a_k}\right) \le$$

$$\left(1+\frac{1}{a_1}\right)\left(1+\frac{1}{a_2}\right)\left(1+\frac{1}{a_{i-1}}\right)\underbrace{\left(1+\frac{1}{a_i}\right)\ldots\left(1+\frac{1}{a_i}\right)}_{k-i+1 \text{ пъти}},$$

което може да се запише във вида

$$(1+a_1)(1+a_2)\ldots(1+a_{i-1})\underbrace{(1+a_i)\ldots(1+a_i)}_{k-i+1 \text{ пъти}} \ge na_1a_2\ldots a_{i-1}\underbrace{a_i\ldots a_i}_{k-i+1 \text{ пъти}}.$$

От него се получава

ограничението $a_i \le \dfrac{1}{\sqrt[k-i+1]{\dfrac{na_1 \ldots a_{i-1}}{(1+a_1)\ldots(1+a_{i-1})}}-1}$. Накрая, ако са известни стойностите на a_1, a_2, …, a_{k-1}, напълно достатъчно е да се провери дали решението на уравнението $(a_1+1)(a_2+1)\ldots(a_k+1) = na_1a_2\ldots a_k$ (неизвестното е a_k) е цяло число.

Реализация на програмата:

```
#include<iostream>
using namespace std;

int a[30];
int n, k;
bool okk;

void next(int i, int k)
{
  bool ok;
  long long left = 1, right = n;
  for(int j = 1; j < i; j++)
  {
    left *= 1+a[j];
    right *= a[j];
  }
  if (left >= right) return;
  if (i == k)
  { if (left == (left/(right-left)) * (right-left))
      {
        a[k] = left/(right-left);
        okk = true;
      }
    return;
  }
  a[i] = a[i-1];
  do
  {
    ok = false;
    for (int j = i; j <= k; j++)
     { left *= 1 + a[i];
       right *= a[i];
     }
    if (left >= right)
      { ok = true;
        next(i+1, k);
        if (okk) return;
      }
    for (int j=i; j<=k; j++)
     { left /= 1 + a[i];
       right /= a[i];
     }
    a[i]++;
  } while(ok);
}
```

```
int main()
{
  cin >> n;
  k = 1;
  while (n > (1<<k)) k++;
  okk = false;
  while (!okk)
  {
    a[0] = 1;
    for (int i = 1; i < 30; i++) a[i] = 0;
    next(1, k);
    k++;
  }
  cout << k - 1 << endl;
  return 0;
}
```

Автор: Младен Манев