

## Незнайка и простые бразильские числа. 7-10 классы

Незнайка бродил по интернету и наткнулся на архив задач Международной иберо-американской олимпиады. Его внимание привлекла задача о простых бразильских числах. Незнайка выяснил, что простое бразильское число – это простое число, которое в некоторой позиционной системе по основанию  $b$ , где  $b > 1$ , записывается тремя или более чем тремя единицами, при этом никаких других цифр кроме единиц в записи числа нет. Например,  $7 = 111_2$ ,  $13 = 111_3$ ,  $31 = 11111_2 = 111_5$ . Но оказалось, что такую запись могут иметь некоторые составные числа. Например,  $21 = 111_4$ ,  $111 = 111_{10}$ . Такие числа к простым бразильским числам не относятся.

Незнайка очень увлёкся простыми бразильскими числами. Буквально про каждое встреченное им число он хотел знать, является ли оно простым бразильским числом. Помогите Незнайке и составьте для него программу.

Программа считывает десятичное натуральное ненулевое число  $N$ , а затем последовательность из  $N$  десятичных натуральных ненулевых чисел  $A[i]$ . Программа находит все простые бразильские числа в последовательности и выводит количество различных простых бразильских чисел в последовательности. Если простых бразильских чисел в последовательности нет, то программа выводит 0.

*Формат ввода:* В первой строке содержится десятичное натуральное число  $N$ :  $0 < N < 1001$ . Во второй строке содержится последовательность из  $N$  десятичных натуральных чисел  $A[i]$ :  $0 < A[i] < 60001$ ,  $i = 1, \dots, N$ .

*Формат вывода:* Выводится десятичная запись без незначащих нулей количества различных простых бразильских чисел в последовательности  $A[i]$ .

Ввод примера №1:		Ввод примера №2:		Ввод примера №3:
4		1		3
2 7 21 7		111		31 13 7
Вывод примера №1:		Вывод примера №2:		Вывод примера №3:
1		0		3

## Решение

В диапазон, заданный в условии задачи, попадает лишь 70 простых бразильских чисел: 7, 13, 31, ..., 55987. Подробнее см. на странице про последовательность A085104 в OEIS (On-Line Encyclopedia of Integer Sequences): <https://oeis.org/A085104>. Для получения эффективной программы следует самостоятельно рассчитать эти числа. Массив из них будет использоваться для быстрой проверки того, является ли число простым бразильским.

В решении используем массив  $BP$  из предварительно рассчитанных простых бразильских чисел. Также используем массив  $C$  из 70 булевых значений. Массив  $C$  инициализируем ложными значениями. В цикле будем считывать очередное число  $A[i]$ . Бинарным поиском по массиву  $BP$  будем проверять, является ли число простым бразильским. При успешной проверке получим индекс числа в массиве  $BP$ . В элемент массива  $C$  с тем же индексом занесём истинное значение. Если ранее там было ложное значение, то увеличим счётчик различных бразильских чисел в последовательности. Если счётчик насчитал 70 чисел, то не нужно дочитывать последовательность, так как ответ уже известен.

## Код возможного решения

```
program BRAZILPRIMES710(input, output);
const NUMBP =70;
    BP: array [1 .. NUMBP] of word = (7, 13, 31, 43, 73, 127, 157, 211,
    241, 307, 421, 463, 601, 757, 1093, 1123, 1483, 1723, 2551, 2801,
    2971, 3307, 3541, 3907, 4423, 4831, 5113, 5701, 6007, 6163, 6481,
    8011, 8191, 9901, 10303, 11131, 12211, 12433, 13807, 14281, 17293,
    19183, 19531, 20023, 20593, 21757, 22621, 22651, 23563, 24181, 26083,
    26407, 27061, 28057, 28393, 30103, 30941, 31153, 35533, 35911, 37057,
    37831, 41413, 42643, 43891, 46441, 47743, 53593, 55933, 55987);
var N, A, J : word;
    I, RESULT : byte;
    C: array [1 .. NUMBP] of boolean;
```

```

    FLAG : boolean;
function binSearch(A : word) : byte;
var L, M, H : byte;
begin
    L := 1;
    H := NUMBP;
    while L <= H do
    begin
        M := (L + H) div 2;
        if BP[M] > A then
        begin
            H := M - 1;
        end
        else if BP[M] < A then
        begin
            L := M + 1;
        end
        else
        begin
            break;
        end;
    end;
    if (M < NUMBP) AND (BP[M + 1] <= A) then binSearch := M + 1
    else if (M > 1) AND (BP[M] > A) then binSearch := M - 1
    else binSearch := M
end;

begin
    RESULT := 0;
    FLAG := false;
    for I := 1 to NUMBP do C[I] := false;
    J := 0;
    readln(N);
    repeat
        read(A);
        J := J + 1;
        I := binsearch(A);
        if A = BP[I] then
            if not C[I] then begin
                C[I] := true;
                RESULT := RESULT + 1;
                if RESULT = NUMBP then FLAG := true
            end;
        until FLAG or (J >= N);
    write(RESULT)
end.

```