‹Primos› e ‹Crivo de Eratóstenes›

Kaio Christaldo
Fabricio Matsunaga

Numeros Primos

Apresentação Problema Motivador

beecrowd | 1165

Número Primo

Adaptado por Neilor Tonin, URI Brasil

Timelimit: 1

Na matemática, um Número Primo é aquele que pode ser dividido somente por 1 (um) e por ele mesmo. Por exemplo, o número 7 é primo, pois pode ser dividido apenas pelo número 1 e pelo número 7.

Entrada

A entrada contém vários casos de teste. A primeira linha da entrada contém um inteiro **N** ($1 \leq N \leq 100$), indicando o número de casos de teste da entrada. Cada uma das **N** linhas seguintes contém um valor inteiro **X** ($1 < X \leq 10^7$), que pode ser ou não, um número primo.

Saída

Para cada caso de teste de entrada, imprima a mensagem "**X** eh primo" ou "**X** nao eh primo", de acordo com a especificação fornecida.

Exemplo de Entrada	Exemplo de Saída
3 8 51 7	8 nao eh primo 51 nao eh primo 7 eh primo

1165 - Número Primo

‹Primos e Fatores›

Um numero A é chamado de fator ou divisor de um numero B se A divide B.
Um numero $N > 1$ será primo se seus unicos fatores forem 1 e N.

Exemplo:
7, 19 e 41 são primos, mas 35 não é, porque $5 \cdot 7 = 35$.

‹Fatoração em Primos›

Para todo numero N > 1 existe uma fatoração de primos unica

$$n = p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_k^{\alpha_k},$$

Onde p1, p2, p3, ... pK, são numeros primos distintos, e α1, α2, α3, ... , αK são numeros positivos. Por exemplo a fatoração de Primos de 84 é:

$$84 = 2^2 \cdot 3^1 \cdot 7^1.$$

Calculo do Numero de Fatores

$$n = p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_k^{\alpha_k},$$

$$84 = 2^2 \cdot 3^1 \cdot 7^1.$$

ow many times it

$$\tau(84) = 3 \cdot 2 \cdot 2 = 12.$$

$$\tau(n) = \prod_{i=1}^{k} (\alpha_i + 1),$$

Os fatores de 84 são 1,2,3,4,6,7,12,14,21,28,42,84

Soma dos Fatores de N

$$\sigma(n) = \prod_{i=1}^{k}(1 + p_i + \ldots + p_i^{\alpha_i}) = \prod_{i=1}^{k} \frac{p_i^{a_i+1} - 1}{p_i - 1},$$

$$\sigma(84) = \frac{2^3 - 1}{2 - 1} \cdot \frac{3^2 - 1}{3 - 1} \cdot \frac{7^2 - 1}{7 - 1} = 7 \cdot 4 \cdot 8 = 224.$$

<Algoritmo Basico>

Se um numero N, nao é primo, ele pode ser representado como um produto A*B, onde A ou B é menor igual a raiz quadrada de N, então certamente ele possui um fator entre 2 e raiz de N.

```
bool primo(int n){
    if( n < 2) return false;
    for (int i = 2; i*i <= n; i++)
    {
        if (n%i == 0)
        {
            return false;
        }
    }
    return true;

}
```

Resolução do Problema Motivador

A resolução estará disponível no Drive. Tente resolver por conta própria e, se precisar, compare com a solução! 😊

Crivo de Eratóstenes

Algorithm <crivo de erastóstenes>

- O Crivo de Eratóstenes (Sieve of Eratosthenes) é um dos algoritmos mais antigos e eficientes para encontrar todos os números primos até um limite n.
- Criado pelo matemático grego Eratóstenes de Cirene por volta de 240 a.C.
- Complexidade **O(n log log n)**

Algorithm <crivo de erastóstenes>

Ideia principal

1. Começamos com uma lista de números de 2 até n.
2. Repetidamente:
    a. Pegamos o menor número não marcado (ele é primo?).
    b. Marcamos todos os seus múltiplos como não primos.
3. O processo continua até o menor número não marcado ser maior que $\sqrt{n}$

Implementação em C++

Algorithm ‹crivo de erastóstenes›

```cpp
vector<int> crivo_erastotenes(int n) {
  // inicializa como a ideia de que todos os numeros são primos
  vector<bool> isPrime(n + 1, true);

  isPrime[0] = isPrime[1] = false;

  for (int i = 2; i <= n; i++) { // percorre 2 até n

    if (isPrime[i]) { // se o número é primo
      for (int j = i*i; j <= n; j += i) // percorre todos multiplos de i*i até n
        isPrime[j] = false; // marca múltiplo como não primo (false)
    }
  }

  vector<int> primes; // vetor de resposta

  for (int i = 2; i <= n; i++) { // percorre vetor com números primos e não primos
    if (isPrime[i]) primes.push_back(i); // adiciona em "prime" apenas numeros verdadeiramente primos
  }
  return primes;
};
```

Algorithm ‹crivo de erastóstenes›

Exemplo: Vamos buscar todos numéros primos de 1 até 100:

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Algorithm ‹crivo de erastóstenes›

Resposta: 2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Algorithm ‹crivo de erastóstenes›

Vantagens do Crivo de Eratóstenes:

- Rápido para gerar todos os primos até n com complexidade $O(n \log \log n)$
- Simples de implementar e entender.
- Pode ser facilmente adaptado para crivo segmentado, bitset, etc.
- Gera a lista completa de primos, útil para diversos problemas (fatoração, teste rápido, contagem, etc).
- Tamanho de entrada aceitável em 1 segundo - 10^7

Algorithm <crivo de erastóstenes>

Algoritmos Relacionados:

- Crivo Linear (Euler's Sieve)
- Crivo Segmentado (Segmented Sieve)
- Teste de primalidade probabilístico (ex: Miller–Rabin)
- Algoritmos de contagem de primos (ex: Meissel-Lehmer)
- Fatoração com crivo (usando menor fator primo)
- Crivo de Atkin

Apresentação Problema Motivador

[2589 - Maior Distância Entre Primos Consecutivos]

Resolução do Problema Motivador

2589 - Maior Distância Entre Primos Consecutivos

A resolução estará disponível no Drive. Tente resolver por conta própria e, se precisar, compare com a solução! 😊

Lista de Exercícios

1022 - TDA Racional
2514 - Alinhamento Lunar
2063 - Caçando Digletts
1028 - Figurinhas

beecrowd

Se tiver alguma dúvida ou dificuldade na resolução de algum exercício, sinta-se à vontade para perguntar! 😊