Thuật toán Sàng Eratosthenes

Nguyễn Đức Hùng

Ngày 26 tháng 5 năm 2023

Mục lục

1. Giới thiệu
2. Thuật toán Sàng Eratosthenes
3. Mã C++ cho Thuật toán Sàng Eratosthenes
4. Giải thích mã

Thuật toán Sàng Eratosthenes

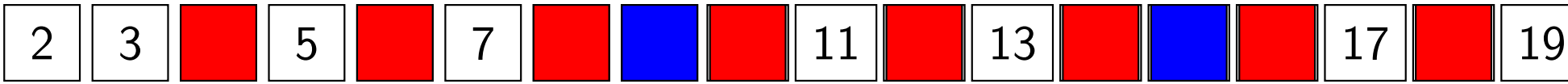
Nguyễn Đức Hùng

Ngày 26 tháng 5 năm 2023

Giới thiệu

Trong bài trình diễn này, chúng tôi sẽ minh họa cách Thuật toán Sàng Eratosthenes hoạt động để tìm tất cả các số nguyên tố không vượt quá một giới hạn cho trước.

Thuật toán Sàng Eratosthenes



- Chúng ta bắt đầu từ số nguyên tố đầu tiên, đó là 2, và đánh dấu tất cả các bội số của nó bằng màu đỏ.
- Sau đó, chúng ta di chuyển đến số tiếp theo chưa được đánh dấu, đó là 3, và đánh dấu tất cả các bội số của nó bằng màu xanh lam.
- Chúng ta làm tương tự với số tiếp theo chưa được đánh dấu, đó là 5, đánh dấu tất cả các bội số của nó bằng màu xanh lá cây.
- Tất cả các số chưa được đánh dấu còn lại đều là số nguyên tố.

Mã C++ cho Thuật toán Sàng Eratosthenes

```cpp
#include<iostream>
#define N 100
using namespace std;

int main() {
    int prime[N+1];
    for (int i=0; i<=N; i++)
        prime[i] = 1;

    for (int p=2; p*p<=N; p++) {
        if (prime[p] == 1) {
            for (int i=p*p; i<=N; i += p)
                prime[i] = 0;
        }
    }

    for (int p=2; p<=N; p++)
```

Giải thích mã - Dòng 1

Nhập thư viện iostream

```
#include<iostream>
```

Nhập thư viện iostream, cung cấp các chức năng nhập/xuất.

Giải thích mã - Dòng 2

Định nghĩa N

```
#define N 100
```

Xác định N là 100, đây là giới hạn số nguyên tố mà chúng ta muốn tìm.

Giải thích mã - Dòng 3

Using namespace std

`using namespace std;`

Định nghĩa không gian tên mặc định, cho phép chúng ta sử dụng các thành phần của thư viện chuẩn mà không cần phải viết "std::" trước mỗi lần sử dụng.

Giải thích mã - Dòng 4

int main()

`int main()`

Đây là hàm main, nơi chương trình bắt đầu.

Giải thích mã - Dòng 5

int prime[N+1]

`int prime[N+1]`
Khai báo một mảng gồm N+1 phần tử, mỗi phần tử tương ứng với một số nguyên từ 0 đến N.

Giải thích mã - Dòng 6

for (int i=0; i<=N; i++) prime[i] = 1;

```
for (int i=0; i<=N; i++) prime[i] = 1;
```

Khởi tạo mọi phần tử trong mảng là 1, tức là chúng ta giả định tất cả các số từ 0 đến N đều là số nguyên tố.

Giải thích mã - Dòng 7

for (int p=2; p*p<=N; p++)

```
for (int p=2; p*p<=N; p++)
```

Duyệt qua tất cả các số nguyên từ 2 đến căn bậc hai của N.

Giải thích mã - Dòng 8

if (prime[p] == 1)

```
if (prime[p] == 1)
```

Nếu phần tử p của mảng vẫn là 1 (nghĩa là p vẫn được cho là số nguyên tố).

Giải thích mã - Dòng 9

for (int i=p*p; i<=N; i += p) prime[i] = 0;

```
for (int i=p*p; i<=N; i += p) prime[i] = 0;
```

Đánh dấu tất cả các bội số của p là không phải số nguyên tố bằng cách đặt phần tử tương ứng của mảng về 0.

Giải thích mã - Dòng 10

for (int p=2; p<=N; p++) if (prime[p] == 1) cout « p « " ";

```
for (int p=2; p<=N; p++) if (prime[p] == 1) cout « p « " ";
```

Duyệt qua tất cả các số từ 2 đến N, và nếu phần tử tương ứng trong mảng vẫn là 1 (tức là nó là số nguyên tố), in ra số đó.

Giải thích mã - Dòng 11

return 0;

`return 0;`

Kết thúc hàm main, trả về giá trị 0, cho biết chương trình đã chạy thành công.