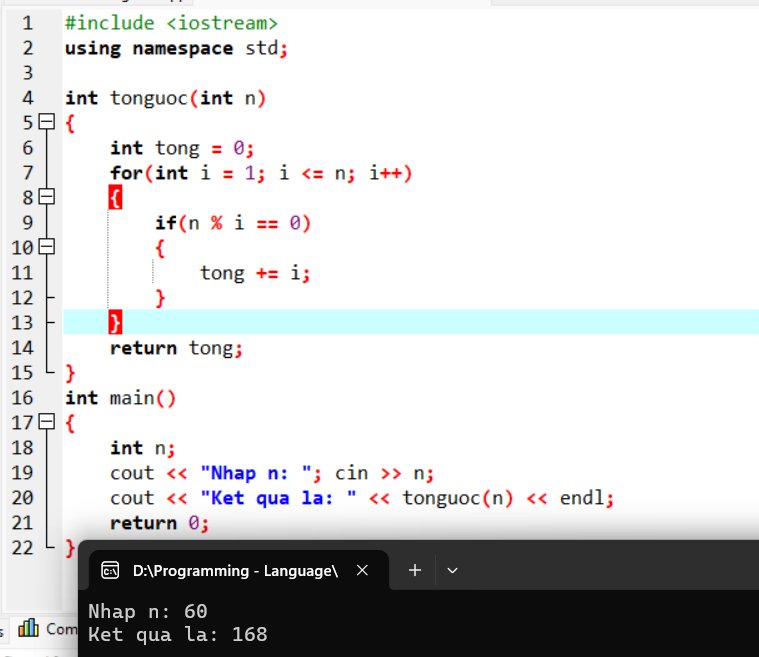
**1. Phương Pháp Cơ Bản**

Để tính tổng ước hay đếm ước của một số nguyên N bạn có thể làm cách đơn giản nhất là duyệt các số từ 1 đến N và kiểm tra tính chia hết của N với các số đó.

Đây là cách làm dễ hiểu nhưng lại không tối ưu về mặt thời gian thực thi.

Vì các ước của N thì chỉ nằm trong khoảng [1, N]

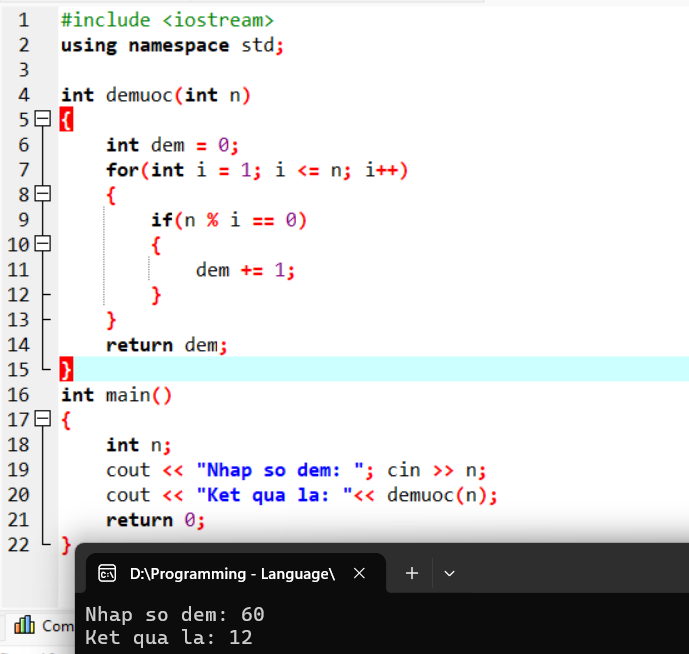
Code 1 : Tính tổng ước của N



- Dòng 11: **tong += i;**: Nếu i là ước của n, cộng i vào tổng tong.

- Dòng 20: cout << "Ket qua la: " << tonguoc(n) << endl;: Gọi hàm tonguoc(n) để tính tổng các ước và in kết quả ra màn hình.

Code 2 : Đếm ước của N



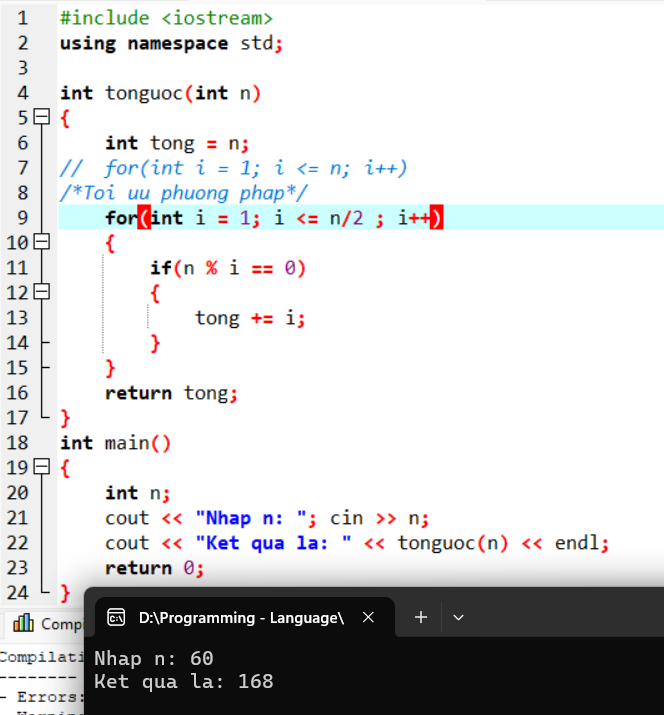
**2. Phương Pháp Tối Ưu 1**

Trong phương pháp cơ bản để tính tổng ước hay đếm ước của số N bạn cần N vòng lặp, bạn có thể cải tiến phương pháp bằng cách duyệt từ 1 tới N / 2.

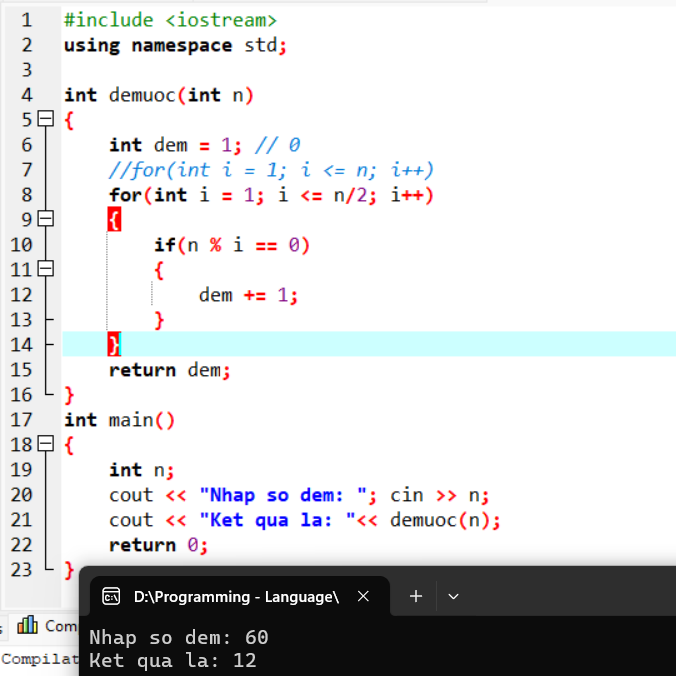
Giải thích vì sao chỉ cần duyệt tới N / 2 : Các ước của N ngoại trừ chính nó đều nhỏ hơn N / 2 vậy nên ta có thể mặc định là N có ước là chính nó và xét các ước còn lại trong đoạn [1, N / 2].

Ví dụ N = 60 có các ước 1, 2, 3, 4, 5, 6, 10, 12, 15, 20, 30, 60 nhưng ngoại trừ 60 thì các ước còn lại đều nằm trong đoạn [1, N / 2]

Code 1 : Tính tổng ước của N



Code 2 : Đếm ước của N



**3. Phương Pháp Tối Ưu 2**

Trong phương pháp tối ưu 1 để tìm ước của N bạn chỉ cần N / 2 vòng lặp. Phương pháp tối ưu thứ 2 thì bạn chỉ cần duyệt từ 1 tới √N là đủ.

Giải thích : Để tìm được tổng ước hay đếm ước của một số ta cần xét tất cả các ước của N, vậy nếu duyệt từ 1 tới √N thì sẽ không duyệt được các ước lớn hơn √N. Ví dụ với N = 60 thì √N = 7 (lấy số nguyên) sẽ không xét được các ước như 20, 30, 60 ?

Với số tự nhiên N, bạn luôn có thể viết N thành tích của 2 ước của nó. Ví dụ với N bằng 60 thì bạn có thể viết thành 1x60, 2x30, 3x20, 4x15, 5x12, 6x10.

Giả sử viết N = a \* b và a ≤ b trong đó a và b tương ứng với 2 ước của N thì chắc chắn a ≤ √N, vì nếu a > √N thì b > √N và khi đó tích của a và b sẽ vượt quá N.

Vậy nên khi xét tất cả các ước của N thì ta chỉ cần xét được ước nhỏ hơn (√N) và từ ước nhỏ hơn đó suy ra được ước còn lại.

Ví dụ với N = 60 :

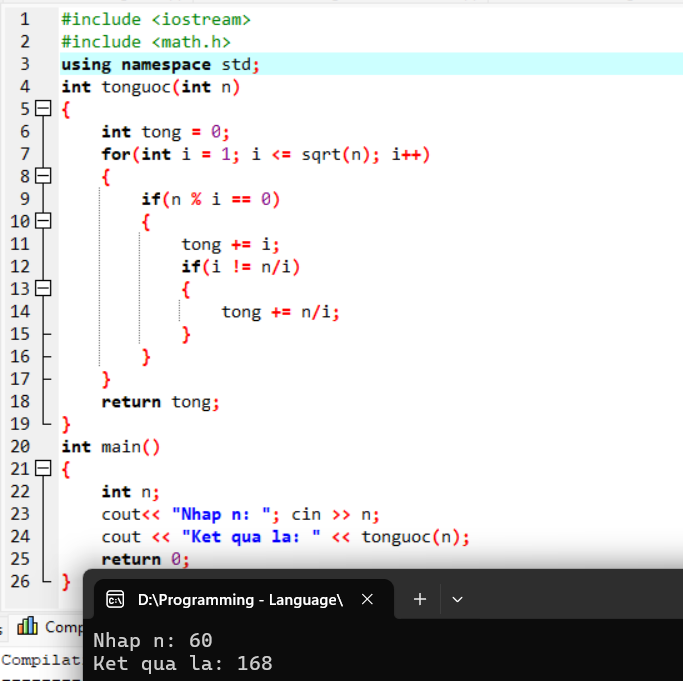
|  |  |
| --- | --- |
| a | b |
| 1 | 60 |
| 2 | 30 |
| 3 | 20 |
| 4 | 15 |
| 5 | 12 |
| 6 | 10 |

**Chú ý** : Với N là số chính phương thì sẽ xảy ra trường hợp 2 ước a và b bằng nhau, khi đó bạn chỉ được xét 1 lần.

Ví dụ với N = 16

|  |  |
| --- | --- |
| a | b |
| 1 | 16 |
| 2 | 8 |
| 4 | 4 |

Code 1 : Tính tổng ước của N



- Dòng 7: Lặp từ 1 đến căn bậc 2 của n

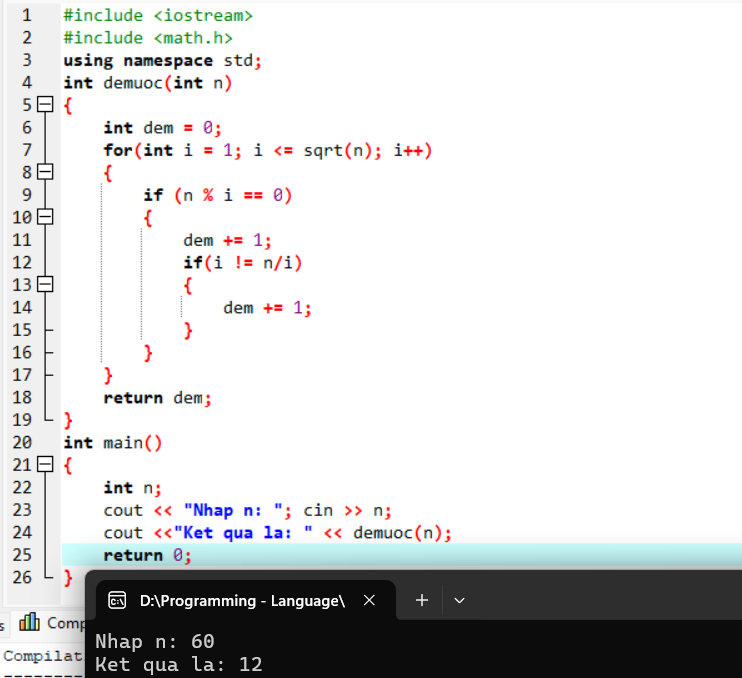
- Dòng 9: Kiểm tra nếu ‘i’ là ước của ‘n’ (tức là n chua hết cho i)

- Dòng 11: Cộng ước số ‘i’ vào tổng ‘tong’

- Dòng 12: Kiểm tra nếu i không phải là căn bậc 2 của n

- Dòng 14: Cộng ước số n/i vào tổng ‘tong’

Code 2 : Đếm ước của N



**KẾT LUẬN** : Trong 3 phương pháp trên thì phương pháp thứ 3 tối ưu nhất tuy nhiên cũng sẽ khó hiểu nhất, các bạn nên sử dụng code thứ 3 để tìm ước và tính tổng ước sau khi kết thúc bài học này.