|  |  |  |
| --- | --- | --- |
| **Toán tử** | **Mô tả** | **Ví dụ** |
| + | Thực hiện **cộng**hai toán hạng | **a + b** kết quả bằng 19 |
| - | Thực hiện **trừ**hai toán hạng | **a - b** kết quả bằng 1 |
| \* | Thực hiện **nhân**hai toán hạng | **a \* b** kết quả bằng 90 |
| / | Thực hiện **chia lấy phần nguyên** hai toán hạng nếu 2 toán hạng là số nguyên. Ngược lại thì thực hiện chia bình thường | **a / b** kết quả bằng 1 |
| % | Thực hiện **chia lấy dư** | **a%b** kết quả bằng 1 |
| ++ | **Tăng** giá trị lên 1 đơn vị | **a++** kết quả bằng 11 |
| -- | **Giảm** giá trị xuống 1 đơn vị | **a--**kết quả bằng 9 |

**Lưu ý:**đối với toán tử **++**và --cần phần biệt a++ và ++a (hoặc a-- và --a):

* **a++**: là sử dụng giá trị của biến a để thực hiện biểu thức trước rồi mới thực hiện tăng lên 1 đơn vị. Tương tự cho a--.
* **++a**: là tăng giá trị biến a lên 1 đơn vị rồi mới sử dụng biến a để thực hiện biểu thức. Tương tự cho --a.

|  |  |  |
| --- | --- | --- |
| **Toán tử** | **Mô tả** | **Ví dụ** |
| == | So sánh 2 toán hạng có bằng nhau hay không. Nếu **bằng**thì trả về true nếu **không bằng** thì trả về false | a == b sẽ trả về false |
| != | So sánh 2 toán hạng có bằng nhau hay không. Nếu **không bằng** thì trả về true nếu **bằng** thì trả về false | a != b sẽ trả về true |
| > | So sánh 2 toán hạng bên trái có **lớn hơn** toán hạng bên phải hay không. Nếu **lớn hơn** thì trả về true nếu **không lớn hơn** thì trả về false | a > b sẽ trả về true |
| < | So sánh 2 toán hạng bên trái có **nhỏ hơn** toán hạng bên phải hay không. Nếu **nhỏ hơn** thì trả về true nếu **không nhỏ hơn** thì trả về false | a < b sẽ trả về false |
| >= | So sánh 2 toán hạng bên trái **có lớn hơn** **hoặc bằng** toán hạng bên phải hay không. Nếu **lớn hơn hoặc bằng** thì trả về true nếu **nhỏ hơn** thì trả về false | a >= b sẽ trả về true |
| <= | So sánh 2 toán hạng có **nhỏ hơn hoặc bằng** hay không. Nếu **nhỏ hơn hoặc bằng** thì trả về true nếu **lớn hơn** thì trả về false | a <= b sẽ trả về false |

**Lưu ý:**

1. Các toán tử quan hệ này chỉ áp dụng cho số hoặc ký tự.
2. Hai toán hạng hai bên phải cùng loại (cùng là số hoặc cùng là ký tự).
3. Bản chất của việc so sánh 2 ký tự với nhau là so sánh mã ASCII của các ký tự đó.

* **Ví dụ:** so sánh ‘**A**’ và ‘**B**’ bản chất là so sánh số **65**với **66**.

1. Không nên sử dụng các toán tử trên để so sánh các chuỗi với nhau vì bản chất việc so sánh chuỗi là so sánh từng ký tự tương ứng với nhau mà so sánh ký tự là so sánh mã ASCII của ký tự đó như vậy ký tự ‘**K**’ sẽ khác ký tự ‘**k**’. Để so sánh hai chuỗi người ta thường dùng hàm so sánh chuỗi đã được hỗ trợ sẵn (sẽ tìm hiểu ở những bài tiếp theo).

Toán tử logic

Giả sử mệnh đề A là đúng và mệnh đề B là sai:

|  |  |  |
| --- | --- | --- |
| **Toán tử** | **Mô tả** | **Ví dụ** |
| && | Hay còn gọi là toán tử logic AND (và). Trả về true nếu **tất cả** toán hạng đều mang giá trị true. Và trả về false nếu có **ít nhất 1**toán hạng mang giá trị false. | A && B kết quả là false |
| || | Hay còn gọi là toán tử logic OR (hoặc). Trả về true nếu có **ít nhất 1** toán hạng mang giá trị true. Và trả về false nếu **tất cả** toán hạng đều mang giá trị false. | A || B kết quả là true. |
| ! | Hay còn gọi là toán tử logic NOT (phủ định). Có chức năng **đảo ngược** trạng thái logic của toán hạng. Nếu toán hạng đang mang giá trị true thì kết quả sẽ là false và ngược lại. | !A kết quả là false |

**Lưu ý:**

* Các toán tử **&&**và **||** có thể áp dụng đồng thời nhiều toán hạng, ví dụ như: **A && B && C || D || K**(Thứ tự thực hiện sẽ được trình bày ở phần sau).
* Các toán hạng trong biểu thức chứa toán tử logic phải trả về true hoặc false.

Toán tử khởi tạo và gán

Toán tử khởi tạo và gán thường được sử dụng nhằm mục đích lưu lại giá trị cho một biến nào đó. Một số toán tử khởi tạo và gán hay được sử dụng:

|  |  |  |
| --- | --- | --- |
| **Toán tử** | **Mô tả** | **Ví dụ** |
| = | Gán giá trị của toán hạng **bên phải** cho toán hạng **bên trái**. | **K = 10**sẽ gán 10 cho biến**K** |
| += | Lấy toán hạng **bên trái cộng** toán hạng **bên phải** sau đó gán kết quả lại cho toán hạng **bên trái**. | **K += 1** tương đương với**K = K + 1** |
| -= | Lấy toán hạng **bên trái trừ** toán hạng**bên phải** sau đó gán kết quả lại cho toán hạng **bên trái**. | **K -= 1**tương đương với **K = K – 1** |
| \*= | Lấy toán hạng **bên trái nhân** toán hạng **bên phải** sau đó gán kết quả lại cho toán hạng **bên trái**. | **K \*= 1** tương đương với **K = K \* 1** |
| /= | Lấy toán hạng **bên trái chia lấy phần nguyên** với toán hạng **bên phải** sau đó gán kết quả lại cho toán hạng **bên trái**. | **K /= 1** tương đương với **K = K / 1** |
| %= | Lấy toán hạng **bên trái chia lấy dư** với toán hạng **bên phải** sau đó gán kết quả lại cho toán hạng **bên trái**. |  |

### Toán tử so sánh trên bit

Các toán tử so sánh trên bit cũng ít gặp nên mình chỉ giới thiệu qua cho các bạn tham khảo thôi chứ chúng ta không giới thiệu rõ phần này.

Giả sử a có giá trị bằng 10 và b có giá trị bằng 9. Giá trị biến a đổi ra nhị phân là 1010 và giá trị biến b đổi ra nhị phân là 1001

|  |  |  |
| --- | --- | --- |
| **Toán tử** | **Mô tả** | **Ví dụ** |
| & | Sao chép bit 1 tới kết quả nếu nó **tồn tại trong cả hai** toán hạng tại vị trí tương ứng, ngược lại thì bit kết quả bằng 0 | **a&b** sẽ cho kết quả là 1000 tương đương với số 8 trong hệ thập phân |
| | | Sao chép bit 1 tới kết quả nếu nó **tồn tại ở một trong hai** toán hạng tại vị trí tương ứng, ngược lại thì bit kết quả bằng 0 | **a|b**sẽ cho kết quả 1011 tương đương với số 11 trong hệ thập phân |
| ^ | Sao chép bit 1 tới kết quả nếu**nó chỉ tồn tại ở một** toán hạng tại vị trí tương ứng, ngược lại thì bit kết quả bằng 0 | **a^b** sẽ cho kết quả 0011 tương đương với số 3 trong hệ thập phân |
| ~ | Dùng để **đảo bit** 0 thành 1 và ngược lại 1 thành 0 | **~a** sẽ cho kết quả 0101 |
| << | Dịch trái n bit. Giá trị toán hạng bên trái sẽ được **dịch trái** n bit với n được xác định bởi toán hạng bên phải | **a<<2** sẽ cho kết quả 101000 |
| >> | Dịch phải n bit. Giá trị toán hạng bên trái sẽ được **dịch phải** n bit với n được xác định bởi toán hạng bên phải | **a>>2** sẽ cho kết quả 0010 |

### Toán tử khác

Ngoài những toán tử đã giới thiệu ở trên chúng ta vẫn còn nhiều toán tử khác cũng hay sử dụng:

|  |  |  |
| --- | --- | --- |
| **Toán tử** | **Mô tả** | **Ví dụ** |
| **sizeof()** | Trả về kích cỡ của một kiểu dữ liệu | sizeof(int) sẽ trả về 4 |
| **typeof()** | Trả về kiểu của một lớp (khái niệm về lớp sẽ được trình bày trong bài [CLASS TRONG C#](https://www.howkteam.vn/redirect?Id=CDPvobsj8EgqSWOjQulkj7KLp%2fXKC0Hu8VwCLICnWtR7jIx329UJHtR2ye6amrOPNG9XCkpuL3sH%2bBTXHvuZsilhzAD9vlxbFZmJ6rRipmz8lpawuM6yDPefVkfU0NB2)) | typeof(string) sẽ trả về **System.String** |
| **new** | Cấp phát vùng nhớ mới, áp dụng cho kiểu dữ liệu tham chiếu | DateTime dt = new DateTime() (sẽ được trình bày chi tiết trong [STRUCT TRONG C#](https://www.howkteam.vn/redirect?Id=CDPvobsj8EgqSWOjQulkj7KLp%2fXKC0Hu8VwCLICnWtQ7ETwTu7CirW%2bhsSye3P82H1l%2bWTXM1wfXsN8xH%2fpA8ZtxXVHZAuKSuwusPnxCws9%2b2Z1%2bogKJ7z4ELgvikmEhjnbRYRILrh7Uv3YD7IISXQ%3d%3d)) |
| **is** | Xác định đối tượng có phải là một kiểu cụ thể nào đó hay không. Nếu đúng sẽ trả về true ngược lại trả về false | Sẽ được trình bày trong những bài sau |
| **as** | Ép kiểu mà không gây ra lỗi. Nếu ép kiểu không thành công sẽ trả về null | Sẽ được trình bày chi tiết trong [ÉP KIỂU TRONG C#](https://www.howkteam.vn/redirect?Id=CDPvobsj8EgqSWOjQulkj4GqZRiiE5hH8eawJjqljbEIHKZRkZ%2bMh5f6GAmTalx%2fYg8HPxJSYtIps%2bS7VoZstQ%3d%3d) |
| **? :** | Được gọi là **toán tử 3 ngôi**. Tương đương với cấu trúc điều kiện (sẽ được trình bày ở bài [CẤU TRÚC RẼ NHÁNH TRONG C#](https://www.howkteam.vn/redirect?Id=CDPvobsj8EgqSWOjQulkj7KLp%2fXKC0Hu8VwCLICnWtQ7ETwTu7CirW%2bhsSye3P82KzwIOBaJxSdnvy2tMhVofL0%2fq3qdwWXpamKOqZUqofIiJYzrA6lGjRBGS0mMYtYR0vG2Yb8ZiHZzFxPdF8rP8A%3d%3d))  **Cú pháp:**  **(toán hạng 1) ? (toán hạng 2) : (toán hạng 3)**  **Ý nghĩa:**trả về toán hạng 2 nếu toán hạng 1 là true và ngược lại trả về toán hạng 3 | **(1 < 2) ? 1 : 0**    kết quả là 1 vì **toán hạng 1** là (1 < 2) là đúng nên trả về**toán hạng 2** là 1 |
| **,** | Sử dụng toán tử “**,**” để kết nối nhiều biểu thức lại với nhau.  **Cú pháp:**  **(biểu thức 1, biểu thức 2)**  **Ý nghĩa:**Duyệt qua**biểu thức 1**sau đó duyệt qua **biểu thức 2**và trả về giá trị của biểu thức 2 | **(t = 5, 2)** sẽ duyệt qua biểu thức 1 là**t = 5**, thực hiện gán 5 cho t sau đó duyệt qua biểu thức 2 là 2, cuối cùng trả về giá trị là 2 |

## **Độ ưu tiên của toán tử**

Độ ưu tiên của các toán tử biểu thị cho việc toán tử nào được ưu tiên thực hiện trước trong câu lệnh. Độ ưu tiên được tóm tắt ở bảng sau:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Mức** | **Toán tử** | | | | | | | | | | **Thứ tự** |
| Cao nhất | () [] . | | | | | | | | | | Trái sang phải |
|  | +  - | ++  -- | | | !  ~ | | new | | sizeof()  typeof() | | Phải sang trái |
| \* | | | / | | | | % | | | Trái sang phải |
| << | | | | | >> | | | | | Trái sang phải |
| < | | <= | | | > | | | | >= | Trái sang phải |
| == | | | | | != | | | | | Trái sang phải |
| & | | | ^ | | | | | | | | Trái sang phải |
| && | | | | | || | | | | | Trái sang phải |
| ? : | | | | | | | | | |  |
| = | | +=  -= | | | \*=  /= | | | | %= | Phải sang trái |
| Thấp nhất | , | | | | | | | | | |  |

Bảng thống kê trên chỉ thể hiện những toán tử chúng ta đã học, ngoài ra vẫn còn nhiều toán tử khác những ít khi sử dụng nên không đề cập đến.

Từ bảng trên ta có thể rút ra được 1 kinh nghiệm đó là nếu muốn biểu thức nào được thực hiện trước ta chỉ cần nhóm chúng vào cặp ngoặc tròn **( )**là được!