Copilot

In **C programming**, a **unary operator** is an operator that operates on a **single operand** to produce a new value. These operators are essential for various tasks, such as **incrementing/decrementing** values, **negating** expressions, or **dereferencing** pointers. Let's explore some common unary operators:

1. **Unary Minus (-)**: The minus operator changes the sign of its argument. A positive number becomes negative, and a negative number becomes positive. For example:

   ```
   int a = 10;
   int b = -a;  // b = -10
   ```

2. **Increment (++)**: The increment operator increases the value of a variable by 1. It can be used in two ways:

   - **Prefix Increment**: The operator precedes the operand (e.g., `++a`). The value of the operand changes before it is used.
   - **Postfix Increment**: The operator follows the operand (e.g., `a++`). The value of the operand changes after it is used. Example:

   ```
   int a = 1;
   int b = ++a;  // b = 2
   ```

3. **Decrement (–)**: Similar to increment, the decrement operator decreases the value of a variable by 1. It also has prefix and postfix forms:

   - **Prefix Decrement**: The operator precedes the operand (e.g., `--a`). The value of the operand changes before it is used.
   - **Postfix Decrement**: The operator follows the operand (e.g., `a--`). The value of the operand changes after it is used. Example:

   ```
   int a = 1;
   int b = a--;   // b = 1
   int c = a;     // c = 0
   ```

4. **Logical NOT (!)**: The logical NOT operator reverses the logical state of its operand. If a condition is true, `!` makes it false, and vice versa. Example:

   ```
   int a = 10;
   int b = 5;
   if (!(a > b))
       printf("b is greater than a\n");
   else
       printf("a is greater than b");
   ```

5. **Addressof Operator (&)**: The addressof operator returns the memory address of a variable. It is used to create pointers. Example:

   ```
   int x = 42;
   int* ptr = &x;  // ptr now points to the memory location of x
   ```

Remember that these unary operators play crucial roles in C programming and are fundamental for various tasks. 🚀