We can divide the rules for performing the usual arithmetic conversions into two cases:

- *The type of either operand is a floating type.* Use the following diagram to promote the operand whose type is narrower:

```
   long double
        ↑
     double
        ↑
      float
```

That is, if one operand has type `long double`, then convert the other operand to type `long double`. Otherwise, if one operand has type `double`, convert the other operand to type `double`. Otherwise, if one operand has type `float`, convert the other operand to type `float`. Note that these rules cover mixtures of integer and floating types: if one operand has type `long int`, for example, and the other has type `double`, the `long int` operand is converted to `double`.

- *Neither operand type is a floating type.* First perform integral promotion on both operands (guaranteeing that neither operand will be a character or short integer). Then use the following diagram to promote the operand whose type is narrower:

```
unsigned long int
        ↑
    long int
        ↑
  unsigned int
        ↑
       int
```

There's one special case, but it occurs only when `long int` and `unsigned int` have the same length (32 bits, say). Under these circumstances, if one operand has type `long int` and the other has type `unsigned int`, both are converted to `unsigned long int`.

When a signed operand is combined with an unsigned operand, the signed operand is converted to an unsigned value. The conversion involves adding or subtracting a multiple of $n + 1$, where n is the largest representable value of the unsigned type. This rule can cause obscure programming errors.

Suppose that the `int` variable `i` has the value −10 and the `unsigned int` variable u has the value 10. If we compare `i` and `u` using the < operator, we might expect to get the result 1 (true). Before the comparison, however, `i` is converted to `unsigned int`. Since a negative number can't be represented as an unsigned integer, the converted value won't be −10. Instead, the value 4,294,967,296 is added (assuming that 4,294,967,295 is the largest `unsigned int` value), giving