the part of the simple expression that precedes the operator, and the term that immediately follows the operator. The two operands of a relational operator are the simple expressions that immediately precede and follow the operator. The operand of a sign in a simple expression is the term that immediately follows the sign. The operand of `not` in a factor is the factor following `not`

The order of evaluation of the operands of an operator is implementation–dependent. A standard program must not make any assumption about this order. The left operand might be evaluated before or after the right operand, or they might be evaluated in parallel. In fact, sometimes one operand might not be evaluated at all for some values of the other operand. For example, evaluating the expression `(j * (i div j))` when `j` is zero might yield zero on one implementation, where on another implementation it might be an error due to the division by zero.

The type of a factor is derived from the type of its constituent (e.g., variable or function). If the constituent's type is a subrange, then the type of the factor is the host type of the subrange; if the constituent's type is a set type with a subrange as its base type, then the type of the factor is a set type with the host type of that subrange type as its base type; otherwise, the type of the factor is the same as the type of the constituent.

The symbol `nil` possesses every pointer type and represents the nil value.

A set constructor denotes a set value. If there are no element descriptions in the set constructor, then it denotes the empty set that is a value of every set type. Otherwise, the elements of the set value are described by the element descriptions in the set constructor. All expressions in the element descriptions of a set constructor must have the same type, which is the base type of the type of the set constructor. The type of a set constructor is both packed and unpacked, and is compatible with any other set type that has a compatible base type.

An element description consisting of a single expression describes the element that has the value denoted by the expression. An element description of the form `a..b` describes an element for each value `x` that satisfies `a <= x <= b`. If `a > b`, then `a..b` denotes no elements. The order of evaluation of the expressions in an element description and