5.4 Context-Dependent Precedence

Often the precedence of an operator depends on the context. This sounds outlandish at first, but it is really very common. For example, a minus sign typically has a very high precedence as a unary operator, and a somewhat lower precedence (lower than multiplication) as a binary operator.

The Bison precedence declarations can only be used once for a given token; so a token has only one precedence declared in this way. For context-dependent precedence, you need to use an additional mechanism: the `%prec` modifier for rules.

The `%prec` modifier declares the precedence of a particular rule by specifying a terminal symbol whose precedence should be used for that rule. It's not necessary for that symbol to appear otherwise in the rule. The modifier's syntax is:

```
%prec terminal-symbol
```

and it is written after the components of the rule. Its effect is to assign the rule the precedence of *terminal-symbol*, overriding the precedence that would be deduced for it in the ordinary way. The altered rule precedence then affects how conflicts involving that rule are resolved (see Operator Precedence).

Here is how `%prec` solves the problem of unary minus. First, declare a precedence for a fictitious terminal symbol named UMINUS. There are no tokens of this type, but the symbol serves to stand for its precedence:

```
…
%left '+' '-'
%left '*'
%left UMINUS
```

Now the precedence of UMINUS can be used in specific rules:

```
exp:
  …
| exp '-' exp
  …
| '-' exp %prec UMINUS
```