正誤表（第２刷用）

本文中の間違い　これまでに発見された本文中の間違いは第２刷では全て修正済みである．

その他の注意事項

bison について　bison は，yacc の上位互換に相当し，しかも様々な拡張機能を有する構文解析器生成系である．本書は yacc のみを考慮して記述しており，bison については触れていない．上位互換性のため，本書のほとんどの内容は bison を用いる場合にも有効であるが，いくつかの点で注意が必要である．注意点は bison のヴァージョンにも依存すると思われるが，現時点で判っている点を以下に述べる．

生成ファイル名　yacc でコンパイルすると，C プログラム・ファイル `y.tab.c` が生成される．これに対して，bison で，たとえば `hoge.y` というファイルをコンパイルとすると，C プログラム・ファイル `hoge.tab.c` が生成される．生成ファイルを yacc 流にする場合には，コンパイル・オプションに `-y` または `--yacc` を追加すればよい．このコマンド・オプションは，bison の動作を伝統的な yacc に適合させるためのオプションである．

トークンの宣言　本書の宣言方法

```
%token ID,NUM,REAL;
```

を用いて bison でコンパイルすると，bison がエラーまたは警告が発する場合がある．その場合には，宣言を以下のように修正する．

```
%token ID NUM REAL
```

トークン種別名の並びの区切りには空白を用いる．なお，このエラーまたは警告はコマンド・オプション `-y` または `--yacc` を用いても解消しない．

`yyval`　bison では変数 `yyval` は構文解析関数 `yyparse()` の局所変数として宣言されている．そのため，図 9.17 のように `yyval` を `yyparse()` の外で使用するようなプログラムをコンパイルすると，C コンパイラが変数の未宣言エラーを発する．これを解消するには，たとえば yacc 記述の宣言部（図 9.15）の`%{ %}`内に

```
YYSTYPE myval;
```

と大域変数 `myval` を宣言しておき，yacc 記述の構文規則部の開始記号の意味値を求める箇所では，`$$ = ...` を `myval = ...` に置換し[1]，`yyparse()` の外では `yyval` の代わりに `myval` を用いればよい．なお，このエラーはコマンド・オプション `-y` または `--yacc` を用いても解消しない．

[1] この場合，その開始記号は各構文規則の右辺には現れないと仮定する．