記号処理・第 7 回

2018 年 5 月 22 日

今回の内容

7.1 Minimum C の構文解析部

次の parse0.c は、前回紹介した字句解析部 (scan.h および scan.c) と組み合わせて、単に Minimum C プログラムの文法チェックだけを行うプログラムです。第 5 回に定めた Minimum C の文法に基づいて構文図を作り、第 4 回に解説した構文図からのパーザの生成法に従って書かれたものですが、その際、プログラムの自明な効率化が施してあります。このパーザは字句解析部が設定した大域変数 token の値だけを使用します。単に文法チェックだけを行うので ident や number の値は使用しません。

メモ

parse0.c

```
 1 /*
 2  *      parse0.c
 3  */
 4
 5 #include <stdio.h>
 6 #include <stdlib.h>
 7 #include "scan.h"
 8
 9 static void Exp(void);
10 static void Block(void);
11
12 static void FunArgs(void)
13 {
14     GetToken();
15     if (token != mc_rparen) {
16         Exp();
17         while (token == mc_comma) {
18             GetToken();
19             Exp();
20         }
21         if (token != mc_rparen)
22             Error("関数呼び出しの引数リストに ) がありません");
23     }
24     GetToken();
25 }
26
27 static void Factor(void)
28 {
29     switch (token) {
30     case mc_ident:
```

```
31          /* 変数: <Ident>、または、関数呼び出し: <Ident> <FunArgs> */
32          GetToken();
33          if (token == mc_lparen)
34              FunArgs();
35          break;
36      case mc_number:
37          /* 定数: <Number> */
38          GetToken();
39          break;
40      case mc_lparen:
41          /* 括弧で囲まれた式: ( <Exp> ) */
42          GetToken();
43          Exp();
44          if (token == mc_rparen)
45              GetToken();
46          else
47              Error("閉じ括弧があるべき位置にありません");
48          break;
49      default:
50          Error("式に文法誤りがあります");
51          break;
52      }
53  }
54
55  static void Term(void)
56  {
57      Factor();
58      while (token == mc_mult || token == mc_div || token == mc_mod) {
59          GetToken();
60          Factor();
61      }
62  }
63
64  static void Exp(void)
65  {
66      if (token == mc_plus || token == mc_minus) {
67          GetToken();
68          Term();
69      }
70      else
71          Term();
72      while (token == mc_plus || token == mc_minus) {
73          GetToken();
74          Term();
75      }
76  }
77
78  static void Cond(void)
79  {
80      Exp();
81      if (mc_eq <= token && token <= mc_ge) {
82          GetToken();
83          Exp();
84      }
85      else
86          Error("条件式に文法誤りがあります");
87  }
88
89  static void Statement(void)
90  {
91      switch (token) {
92      case mc_ident:
93          /* <Ident> = <Exp>; | <Ident> <FunArgs>; */
94          GetToken();
95          switch (token) {
```

```
 96          case mc_assign:
 97              GetToken();
 98              Exp();
 99              if (token != mc_semicolon)
100                  Error("代入文に ; がありません");
101              GetToken();
102              break;
103          case mc_lparen:
104              FunArgs();
105              if (token != mc_semicolon)
106                  Error("関数呼び出し文に ; がありません");
107              GetToken();
108              break;
109          default:
110              Error("代入文の =、または、関数呼び出し文の ( がありません");
111              break;
112          }
113          break;
114      case mc_return:
115          /* return <Exp>; */
116          GetToken();
117          Exp();
118          if (token != mc_semicolon)
119              Error("return 文に ; がありません");
120          GetToken();
121          break;
122      case mc_input:
123          /* input <Ident>; */
124          GetToken();
125          if (token != mc_ident)
126              Error("input 文に変数名がありません");
127          GetToken();
128          if (token != mc_semicolon)
129              Error("input 文に ; がありません");
130          GetToken();
131          break;
132      case mc_print:
133          /* print <Exp>; */
134          GetToken();
135          Exp();
136          if (token != mc_semicolon)
137              Error("print 文に ; がありません");
138          GetToken();
139          break;
140      case mc_if:
141          /* if ( <Cond> ) <Statement> [ else <Statement> ] */
142          GetToken();
143          if (token != mc_lparen)
144              Error("if の後に ( がありません");
145          GetToken();
146          Cond();
147          if (token != mc_rparen)
148              Error("if の条件式の後に ) がありません");
149          GetToken();
150          Statement();
151          if (token == mc_else) {
152              GetToken();
153              Statement();
154          }
155          break;
156      case mc_while:
157          /* while ( <Cond> ) <Statement> */
158          GetToken();
159          if (token != mc_lparen)
```

```
160             Error("while の後に ( がありません");
161         GetToken();
162         Cond();
163         if (token != mc_rparen)
164             Error("while の条件式の後に ) がありません");
165         GetToken();
166         Statement();
167         break;
168     case mc_lcb:
169         /* "{" [ <VarDef> ] { <Statement> } "}" */
170         Block();
171         break;
172     case mc_semicolon:
173         Error("空文は許されていません");
174         break;
175     default:
176         Error("文を始めるべき記号に誤りがあります");
177         break;
178     }
179 }
180
181 static void VarDef(void)
182 {
183     GetToken();
184     if (token != mc_ident)
185         Error("宣言される変数名がありません");
186     GetToken();
187     while (token == mc_comma) {
188         GetToken();
189         if (token != mc_ident)
190             Error("宣言される変数名がありません");
191         GetToken();
192     }
193     if (token != mc_semicolon)
194         Error("変数宣言に , か ; がありません");
195     GetToken();
196 }
197
198 static void FunDef(void)
199 {
200     GetToken();
201     if (token != mc_lparen)
202         Error("関数定義の引数リストに ( がありません");
203     GetToken();
204     if (token != mc_rparen) {
205         if (token != mc_ident)
206             Error("関数定義の引数リストに文法誤りがあります");
207         GetToken();
208         while (token == mc_comma) {
209             GetToken();
210             if (token != mc_ident)
211                 Error("関数定義の引数リストに文法誤りがあります");
212             GetToken();
213         }
214         if (token != mc_rparen)
215             Error("関数定義の引数リストに ) がありません");
216     }
217     GetToken();
218     Block();
219 }
220
221 static void Block(void)
222 {
223     if (token != mc_lcb)
224         Error("ブロックが { で始まっていません");
225     GetToken();
```

```
226     if (token == mc_int)
227         VarDef();
228     while (token != mc_rcb)
229         Statement();
230     GetToken();
231 }
232
233 static void Prog(void)
234 {
235     while (token == mc_int || token == mc_ident) {
236         if (token == mc_int)
237             VarDef();
238         else
239             FunDef();
240     }
241 }
242
243 void Parse(void)
244 {
245     GetToken();
246     Prog();
247     if (token != mc_eof)
248         Error("変数宣言や関数定義として認識できません");
249 }
250
251 int main (int argc, char *argv[])
252 {
253     if (argc < 2) {
254         fprintf(stderr, "ソースファイルが指定されていません\n");
255         exit(1);
256     }
257     if (OpenFile(argv[1]) != 0) {
258         fprintf(stderr, "ソースファイルを読み込むことができません\n");
259         exit(1);
260     }
261     Parse();
262     CloseFile();
263     return 0;
264 }
265
```

7.2 演習問題

1. Minimum C の `if` 文に関する曖昧性が、このパーザではどのように扱われることになったか考察しなさい。たとえば、次の $\langle Statement \rangle$ 中の `else` は、2 つの `if` のうちどちらと対応していると解釈されるか考えなさい。

```
if (x > 0) if (y < 0) z = 0; else z = 1;
```

2. 次の Minumum C プログラムを、それぞれこのパーザが読み込んだとき、`parse0.c` の何行目で関数 `Error` が呼び出されるか考えなさい。

(1)

```
foo (x)
{
    if (0 < x < 100)
        x = -x;
}
```

(2)

```
foo (x)
{
    foo (x) = x;
}
```

(3)

```
foo (x)
{
    return x * (x + y;
}
```

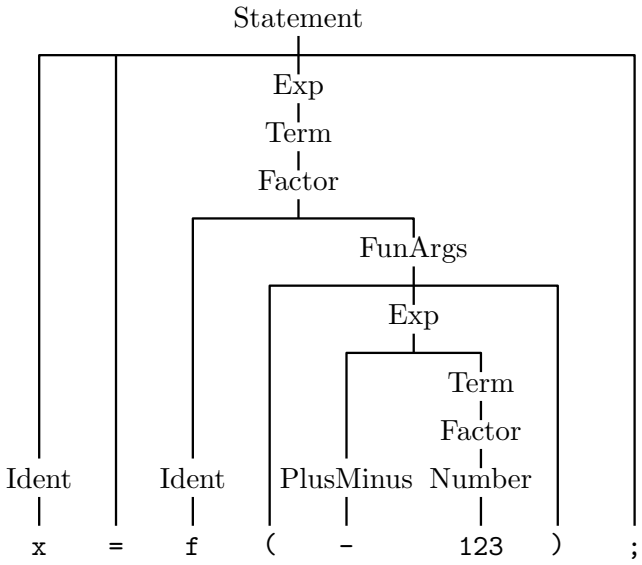
(4)

```
foo (x) {
    while (1) {
        return 0;
    }
}
```

記号処理・第7回・終わり

付録：第 5 回演習問題の解答例

1.



2. 例えば、`if (x>y) if (x>z) a=x; else a=y;` という終端記号列は $\langle Statement \rangle$ に対して次のような 2 つの構文木を持つ。