

编译原理第一次实验测试用例：目录

1	A 组测试用例	2
1.1	A-1	2
1.2	A-2	2
1.3	A-3	3
1.4	A-4	3
1.5	A-5	3
1.6	A-6	4
1.7	A-7	5
1.8	A-8	5
1.9	A-9	6
2	B 组测试用例	7
2.1	B-1	7
2.2	B-2	8
3	C 组测试用例	10
3.1	C-1	10
3.2	C-2	16
4	D 组测试用例	26
4.1	D-1	26
4.2	D-2	29
4.3	D-3	32
5	E 组测试用例	36
5.1	E1.1	36
5.2	E1.2	37
5.3	E1.3	38
6	结束语	40

1 A 组测试用例

本组测试用例共 9 个，每个仅包含单个的词法或者语法错误。除特殊说明外，不可多报。多报、漏报错误，或者打印语法树都会导致扣分。错误编号和行号之后的说明文字不要求与给出的输出完全一致，仅供助教理解使用，不作为评分依据。

1.1 A-1

输入

```
1 int 1Wrong()  
2 {  
3     int _1Correct;  
4     return 1;  
5 }
```

输出

```
1 Error type B at line 1: Illegal ID
```

说明：错误类型可以是 A，也可以是一个 A 一个 B，但只能在第 1 行。这里有一个非法标识符 1Wrong，注意标识符可以以下划线开始，所以第 3 行正确。

1.2 A-2

输入

```
1 int test2()  
2 {  
3     int a = 3;  
4     int b = 4 + 5 * a - 1;  
5     int t = a || b && 0;  
6     int c = a | b;  
7     return 3;  
8 }
```

输出

```
1 Error type A at Line 6: Mysterious character '||'
```

说明：必须有 type A 错误，这里有一个非法符号 |。可以多报一个 type B 错误。

1.3 A-3

输入

```
1 int test3()  
2 {  
3     int a, b, c; {  
4     c = [a + b) * (a + 4);  
5     b = a && c;  
6     return b; }  
7 }
```

输出

```
1 Error type B at line 4: syntax error
```

说明：第 4 行缺少匹配的括号。

1.4 A-4

输入

```
1 int test4() {  
2     int a = 1;  
3     float t = 3.1;  
4     int c[t];  
5     return a;  
6 }
```

输出

```
1 Error type B at Line 4: Missing ']'
```

说明：第 4 行数组定义错误，不允许使用变量作为维度。

1.5 A-5

输入

```

1 int test5(int a, int b)
2 {
3     int c = 3;
4     int d[12];
5     while(c < a) {
6         c = c + 1;
7         d[c] = d[] - 1;
8     }
9     return b;
10 }

```

输出

```

1 Error type B at Line 7: Error arguments between []

```

说明：第 7 行数组下表访问时格式不正确，中括号内不可为空。

1.6 A-6

输入

```

1 int test6()
2 {
3     int i = 0, b;
4     float t = 12.3;
5     int c[10][3];
6     b = 10;
7     while(b > i) {
8         b = b - 1;
9         i = i + 1;
10        c[b,i] = c[b][i] + 1;
11    }
12    return c[0][0];
13 }

```

输出

```
1 Error type B at Line 10: Error arguments between []
```

说明：第 10 行下标访问时格式不正确，错误的二维数组格式。

1.7 A-7

输入

```
1 struct Test7_a {  
2     int a;  
3     float b;  
4 };  
5 struct Test7_b {  
6     float c;  
7 }  
8  
9 int test7() {  
10     struct Test7_a aa[10];  
11     return aa[0].a;  
12 }
```

输出

```
1 Error type B at line 7: Missing ";"
```

说明：第 7 行结构体定义时缺少分号，也可以在第 8 或 9 行报错。

1.8 A-8

输入

```
1 struct Test8_a {  
2     int a;  
3 };  
4  
5 struct Test8_b {  
6     float b;  
7 } bb;
```

```

8
9 int test8() {
10     struct Test8_c {
11         int c;
12     } cc;
13     struct Test8_d {
14         int d;
15         struct test8_d_d{
16             int ddd;
17         };
18     } dd;
19     return 8;
20 }

```

输出

```

1 Error type B at Line 17: Syntax error

```

说明：第 17 行嵌套结构体定义缺少变量名。

1.9 A-9

输入

```

1 int test9_a(int aa) {
2     return aa + 3;
3 }
4
5 int test9_b(int bb, float fb) {
6     int tb = 3;
7     return bb + tb;
8 }
9
10 int test9() {
11     int a = 3;
12     test9_a(a);

```

```

13     test9_b(a, 0.2);
14     return test9(test9_b(test9(test9_a(a)), 1.2);
15 }

```

输出

```

1 Error type B at line 14: syntax error.

```

说明：第 14 行缺少匹配的右括号。

2 B 组测试用例

本组测试用例共 2 个，每个用例包含多处不同的错误。除特殊说明外，漏报、多报错误或者出打印语法树都会导致扣分。

2.1 B-1

输入

```

1 int main() {
2     int a[10][10], dis[1.2][10];
3     int i = 0, j = 0, k = 0;
4     while(i < 10) {
5         while(j < 10) {
6             a[i][j] = i - j * 2;
7             if (a[i][j] < 0) {
8                 a[i][j] = -1;
9                 dis[i][j] = 10000;
10            } else {
11                dis[i][j] = a[i][j];
12            }
13            j++;
14        }
15        i = i + 1;
16    }
17    i = 0;

```

```

18     j = 0;
19     while(k < 10) {
20         while(i < 10) {
21             while(j < 10) {
22                 if(dis[i][k] + dis[k][j] < dis[i][j])
23                     {
24                         dis[i][j] = dis[i][k.] + dis[
25                             k][j];
26                     }
27                 j = j + 1;
28             }
29             i = i + 1;
30         }
31         k = k + 1;
32     }
33     return 1
34 }

```

```

1 Error type B at Line 2: syntax error, unexpected FLOAT, expecting INT
2 Error type B at Line 13: syntax error, unexpected PLUS
3 Error type B at Line 23: syntax error, unexpected RB, expecting ID
4 Error type B at Line 31: syntax error, unexpected RC

```

说明：第 2 行数组定义格式错误，不允许使用浮点数；第 13 行使用了未定义的 ++ 运算符；23 行数组下标访问格式错误，多了一个点；31 行缺少分号，也可以报在 32 行。

2.2 B-2

输入

```

1 struct Application {
2     int id;
3     float name;
4     int destination;
5 };

```



```

6
7 struct Application generateApplication(int gId, float gName, int
    gDestination) {
8     struct Application app;
9     app.id = gId;
10    app.Name = gName;
11    app.destination = gDestination + gId && (gId || 12);
12    return app;
13 }
14
15 int verify(struct Application app) {
16     if (app.id == 0) {
17         return 0;
18     } else if (app.name >< 0) {
19         return 0;
20     } else if (app.destination != app.id) {
21         return 0;
22     }
23     return 1;
24 }
25
26 int main() {
27     int a = 3;
28     int b = 4;
29     float k = 1.2;
30     return verify(generateApplication(a, k, b);
31 }

```

输出

```

1 Error type B at Line 8: syntax error, unexpected ID
2 Error type B at Line 9: syntax error, unexpected ID
3 Error type B at Line 18: syntax error, unexpected RELOP
4 Error type B at Line 30: syntax error, unexpected SEMI, expecting RP

```

说明：第 8 行结构体关键字 `typo`；第 9 行 `gId` 出现 `typo` 导致变量名不合法；第 18 行出现未知运算符 `><`；第 30 行缺少匹配的右括号。

3 C 组测试用例

本组测试用例共 2 个，不包含任何错误，需要输出正确的语法树。除特殊说明外，应与给出的语法树完全相同。语法树打印错误酌情扣分。

3.1 C-1

输入

```
1 struct Player{
2     int name;
3     int ready;
4 } player1;
5
6 struct Game{
7     struct Player players[10];
8     int id;
9 } game1;
10
11 int main() {
12     struct Set {
13         struct Game games[10];
14     } one;
15     one.games[0] = game1;
16     one.games[0].players[0] = player1;
17     return one.games[0].players[0].ready;
18 }
```

输出

```
1 Program (1)
2   ExtDefList (1)
```

```

3      ExtDef (1)
4          Specifier (1)
5              StructSpecifier (1)
6                  STRUCT
7                  OptTag (1)
8                      ID: Player
9                  LC
10             DefList (2)
11                 Def (2)
12                     Specifier (2)
13                         TYPE: int
14                     DecList (2)
15                         Dec (2)
16                             VarDec (2)
17                                 ID: name
18                     SEMI
19                 DefList (3)
20                     Def (3)
21                         Specifier (3)
22                             TYPE: int
23                         DecList (3)
24                             Dec (3)
25                                 VarDec (3)
26                                     ID: ready
27                             SEMI
28                     RC
29             ExtDecList (4)
30                 VarDec (4)
31                     ID: player1
32             SEMI
33      ExtDefList (6)
34          ExtDef (6)

```

```

35     Specifier (6)
36     StructSpecifier (6)
37         STRUCT
38         OptTag (6)
39             ID: Game
40         LC
41         DefList (7)
42             Def (7)
43                 Specifier (7)
44                     StructSpecifier (7)
45                         STRUCT
46                         Tag (7)
47                             ID: Player
48                     DecList (7)
49                         Dec (7)
50                             VarDec (7)
51                                 VarDec (7)
52                                     ID: players
53                                     LB
54                                     INT: 10
55                                     RB
56                     SEMI
57                 DefList (8)
58                     Def (8)
59                         Specifier (8)
60                             TYPE: int
61                         DecList (8)
62                             Dec (8)
63                                 VarDec (8)
64                                     ID: id
65                         SEMI
66         RC

```

```

67     ExtDecList (9)
68         VarDec (9)
69             ID: game1
70         SEMI
71     ExtDefList (11)
72         ExtDef (11)
73             Specifier (11)
74                 TYPE: int
75             FunDec (11)
76                 ID: main
77                 LP
78                 RP
79             Compst (11)
80                 LC
81                 DefList (12)
82                     Def (12)
83                         Specifier (12)
84                             StructSpecifier (12)
85                                 STRUCT
86                                 OptTag (12)
87                                     ID: Set
88                                     LC
89                                     DefList (13)
90                                         Def (13)
91                                             Specifier (13)
92                                                 StructSpecifier (13)
93                                                     STRUCT
94                                                     Tag (13)
95                                                         ID: Game
96                                                         DecList (13)
97                                                             Dec (13)
98                                                                 VarDec (13)

```

```

99             VarDec (13)
100             ID: games
101             LB
102             INT: 10
103             RB
104             SEMI
105             RC
106             DecList (14)
107             Dec (14)
108             VarDec (14)
109             ID: one
110             SEMI
111             StmtList (15)
112             Stmt (15)
113             Exp (15)
114             Exp (15)
115             Exp (15)
116             Exp (15)
117             ID: one
118             DOT
119             ID: games
120             LB
121             Exp (15)
122             INT: 0
123             RB
124             ASSIGNOP
125             Exp (15)
126             ID: game1
127             SEMI
128             StmtList (16)
129             Stmt (16)
130             Exp (16)

```

131	Exp (16)
132	Exp (16)
133	Exp (16)
134	Exp (16)
135	Exp (16)
136	ID: one
137	DOT
138	ID: games
139	LB
140	Exp (16)
141	INT: 0
142	RB
143	DOT
144	ID: players
145	LB
146	Exp (16)
147	INT: 0
148	RB
149	ASSIGNOP
150	Exp (16)
151	ID: player1
152	SEMI
153	StmtList (17)
154	Stmt (17)
155	RETURN
156	Exp (17)
157	Exp (17)
158	Exp (17)
159	Exp (17)
160	Exp (17)
161	Exp (17)
162	ID: one

163		DOT
164		ID: games
165		LB
166		Exp (17)
167		INT: 0
168		RB
169		DOT
170		ID: players
171		LB
172		Exp (17)
173		INT: 0
174		RB
175		DOT
176		ID: ready
177		SEMI
178	RC	

3.2 C-2

输入

```

1  int verify(int ready) {
2      int tem = 0;
3      int i = 2;
4      int last = ready / 4;
5      while(i < last) {
6          tem = ready / i;
7          if (tem * i == ready) {
8              return 0;
9          }
10         i = i + 1;
11     }
12     return 1;
13 }
```



```

14
15 int main() {
16     int numbers[100];
17     int count = 0;
18     int j = 142;
19     int max = 242;
20     while(j < max) {
21         if(verify(j) == 1) {
22             numbers[count] = j;
23             count = count + 1;
24         }
25         j = j + 1;
26     }
27     return count;
28 }

```

输出

```

1 Program (1)
2   ExtDefList (1)
3     ExtDef (1)
4       Specifier (1)
5         TYPE: int
6       FunDec (1)
7         ID: verify
8         LP
9         VarList (1)
10          ParamDec (1)
11            Specifier (1)
12              TYPE: int
13            VarDec (1)
14              ID: ready
15          RP
16        Compst (1)

```

```

17      LC
18      DefList (2)
19          Def (2)
20              Specifier (2)
21                  TYPE: int
22              DecList (2)
23                  Dec (2)
24                      VarDec (2)
25                          ID: tem
26                          ASSIGNOP
27                          Exp (2)
28                              INT: 0
29                      SEMI
30      DefList (3)
31          Def (3)
32              Specifier (3)
33                  TYPE: int
34              DecList (3)
35                  Dec (3)
36                      VarDec (3)
37                          ID: i
38                          ASSIGNOP
39                          Exp (3)
40                              INT: 2
41                      SEMI
42      DefList (4)
43          Def (4)
44              Specifier (4)
45                  TYPE: int
46              DecList (4)
47                  Dec (4)
48                      VarDec (4)

```

49	ID: last
50	ASSIGNOP
51	Exp (4)
52	Exp (4)
53	ID: ready
54	DIV
55	Exp (4)
56	INT: 4
57	SEMI
58	StmtList (5)
59	Stmt (5)
60	WHILE
61	LP
62	Exp (5)
63	Exp (5)
64	ID: i
65	RELOP
66	Exp (5)
67	ID: last
68	RP
69	Stmt (5)
70	Compst (5)
71	LC
72	StmtList (6)
73	Stmt (6)
74	Exp (6)
75	Exp (6)
76	ID: tem
77	ASSIGNOP
78	Exp (6)
79	Exp (6)
80	ID: ready

81	DIV
82	Exp (6)
83	ID: i
84	SEMI
85	StmtList (7)
86	Stmt (7)
87	IF
88	LP
89	Exp (7)
90	Exp (7)
91	Exp (7)
92	ID: tem
93	STAR
94	Exp (7)
95	ID: i
96	RELOP
97	Exp (7)
98	ID: ready
99	RP
100	Stmt (7)
101	Compst (7)
102	LC
103	StmtList (8)
104	Stmt (8)
105	RETURN
106	Exp (8)
107	INT: 0
108	SEMI
109	RC
110	StmtList (10)
111	Stmt (10)
112	Exp (10)

```

113             Exp (10)
114             ID: i
115             ASSIGNOP
116             Exp (10)
117             Exp (10)
118             ID: i
119             PLUS
120             Exp (10)
121             INT: 1
122             SEMI
123             RC
124             StmtList (12)
125             Stmt (12)
126             RETURN
127             Exp (12)
128             INT: 1
129             SEMI
130             RC
131             ExtDefList (15)
132             ExtDef (15)
133             Specifier (15)
134             TYPE: int
135             FunDec (15)
136             ID: main
137             LP
138             RP
139             Compst (15)
140             LC
141             DefList (16)
142             Def (16)
143             Specifier (16)
144             TYPE: int

```

```

145         DecList (16)
146         Dec (16)
147         VarDec (16)
148         VarDec (16)
149         ID: numbers
150         LB
151         INT: 100
152         RB
153     SEMI
154 DefList (17)
155     Def (17)
156         Specifier (17)
157         TYPE: int
158         DecList (17)
159         Dec (17)
160         VarDec (17)
161         ID: count
162         ASSIGNOP
163         Exp (17)
164         INT: 0
165     SEMI
166 DefList (18)
167     Def (18)
168         Specifier (18)
169         TYPE: int
170         DecList (18)
171         Dec (18)
172         VarDec (18)
173         ID: j
174         ASSIGNOP
175         Exp (18)
176         INT: 142

```

```

177         SEMI
178     DefList (19)
179     Def (19)
180         Specifier (19)
181         TYPE: int
182     DecList (19)
183     Dec (19)
184         VarDec (19)
185         ID: max
186         ASSIGNOP
187         Exp (19)
188         INT: 242
189     SEMI
190 StmtList (20)
191     Stmt (20)
192     WHILE
193     LP
194     Exp (20)
195     Exp (20)
196     ID: j
197     RELOP
198     Exp (20)
199     ID: max
200     RP
201     Stmt (20)
202     Compst (20)
203     LC
204     StmtList (21)
205     Stmt (21)
206     IF
207     LP
208     Exp (21)

```

209	Exp (21)
210	ID: verify
211	LP
212	Args (21)
213	Exp (21)
214	ID: j
215	RP
216	RELOP
217	Exp (21)
218	INT: 1
219	RP
220	Stmt (21)
221	Compst (21)
222	LC
223	StmtList (22)
224	Stmt (22)
225	Exp (22)
226	Exp (22)
227	Exp (22)
228	ID: numbers
229	LB
230	Exp (22)
231	ID: count
232	RB
233	ASSIGNOP
234	Exp (22)
235	ID: j
236	SEMI
237	StmtList (23)
238	Stmt (23)
239	Exp (23)
240	Exp (23)

241	ID: count
242	ASSIGNOP
243	Exp (23)
244	Exp (23)
245	ID: count
246	PLUS
247	Exp (23)
248	INT: 1
249	SEMI
250	RC
251	StmtList (25)
252	Stmt (25)
253	Exp (25)
254	Exp (25)
255	ID: j
256	ASSIGNOP
257	Exp (25)
258	Exp (25)
259	ID: j
260	PLUS
261	Exp (25)
262	INT: 1
263	SEMI
264	RC
265	StmtList (27)
266	Stmt (27)
267	RETURN
268	Exp (27)
269	ID: count
270	SEMI
271	RC

4 D 组测试用例

本组测试用例共 3 个，针对不同分组进行测试。对应分组的同学需要输出语法树，提示错误则不得分；其他分组的同学只要提示错误即可，如果打印了语法树，则将视为违规，将会 * 倒扣分*。

4.1 D-1

输入

```
1 int func_test()  
2 {  
3     int _dec_ = 452;  
4     int _oct_ = 06233;  
5     int _dhex_ = 0xAbcDe + _oct_;  
6     int _result_ = - _dhex_ + _oct_ * ( _dec_ - 0X365F );  
7     return 0;  
8 }
```

输出

```
1 Program (1)  
2   ExtDefList (1)  
3     ExtDef (1)  
4       Specifier (1)  
5         TYPE: int  
6       FunDec (1)  
7         ID: func_test  
8         LP  
9         RP  
10      Compst (2)  
11        LC  
12        DefList (3)  
13          Def (3)  
14            Specifier (3)  
15              TYPE: int
```

```

16      DecList (3)
17      Dec (3)
18      VarDec (3)
19      ID: _dec_
20      ASSIGNOP
21      Exp (3)
22      INT: 452
23      SEMI
24  DefList (4)
25  Def (4)
26  Specifier (4)
27  TYPE: int
28  DecList (4)
29  Dec (4)
30  VarDec (4)
31  ID: _oct_
32  ASSIGNOP
33  Exp (4)
34  INT: 3227
35  SEMI
36  DefList (5)
37  Def (5)
38  Specifier (5)
39  TYPE: int
40  DecList (5)
41  Dec (5)
42  VarDec (5)
43  ID: _dhex_
44  ASSIGNOP
45  Exp (5)
46  Exp (5)
47  INT: 703710

```

```

48         PLUS
49         Exp (5)
50         ID: _oct_
51     SEMI
52 DefList (6)
53     Def (6)
54         Specifier (6)
55         TYPE: int
56         DecList (6)
57         Dec (6)
58         VarDec (6)
59         ID: _result_
60     ASSIGNOP
61     Exp (6)
62         Exp (6)
63         MINUS
64         Exp (6)
65         ID: _dhex_
66     PLUS
67     Exp (6)
68         Exp (6)
69         ID: _oct_
70     STAR
71     Exp (6)
72     LP
73     Exp (6)
74         Exp (6)
75         ID: _dec_
76     MINUS
77     Exp (6)
78         INT: 13919
79     RP

```

```

80          SEMI
81      StmtList (7)
82          Stmt (7)
83              RETURN
84              Exp (7)
85                  INT: 0
86              SEMI
87      RC

```

说明：1.1 分组的同学需要输出该语法树，8 进制和 16 进制数必须正确转换；其他分组的同学只要提示有错误，而且不输出语法树即可。

4.2 D-2

输入

```

1  int float_test()
2  {
3      float X_1 = 1.0E3;
4      float X_2 = 1.0e3;
5      float X_3 = 1.E3;
6      float X_4 = .14e2;
7      float X_5 = .12E-2;
8      float X_6 = 23.e+3;
9      return 0;
10 }

```

输出

```

1 Program (1)
2   ExtDefList (1)
3     ExtDef (1)
4       Specifier (1)
5         TYPE: int
6       FunDec (1)
7         ID: float_test

```

```

8      LP
9      RP
10     CompSt (2)
11     LC
12     DefList (3)
13     Def (3)
14     Specifier (3)
15     TYPE: float
16     Declist (3)
17     Dec (3)
18     VarDec (3)
19     ID: X_1
20     ASSIGNOP
21     Exp (3)
22     FLOAT: 1000.000000
23     SEMI
24     DefList (4)
25     Def (4)
26     Specifier (4)
27     TYPE: float
28     Declist (4)
29     Dec (4)
30     VarDec (4)
31     ID: X_2
32     ASSIGNOP
33     Exp (4)
34     FLOAT: 1000.000000
35     SEMI
36     DefList (5)
37     Def (5)
38     Specifier (5)
39     TYPE: float

```

```

40         DecList (5)
41         Dec (5)
42         VarDec (5)
43         ID: X_3
44         ASSIGNOP
45         Exp (5)
46         FLOAT: 1000.000000
47     SEMI
48 DefList (6)
49     Def (6)
50         Specifier (6)
51         TYPE: float
52         DecList (6)
53         Dec (6)
54         VarDec (6)
55         ID: X_4
56         ASSIGNOP
57         Exp (6)
58         FLOAT: 14.000000
59     SEMI
60 DefList (7)
61     Def (7)
62         Specifier (7)
63         TYPE: float
64         DecList (7)
65         Dec (7)
66         VarDec (7)
67         ID: X_5
68         ASSIGNOP
69         Exp (7)
70         FLOAT: 0.001200
71     SEMI

```

```

72         DefList (8)
73         Def (8)
74         Specifier (8)
75         TYPE: float
76         Declist (8)
77         Dec (8)
78         VarDec (8)
79         ID: X_6
80         ASSIGNOP
81         Exp (8)
82         FLOAT: 23000.000000
83         SEMI
84     StmtList (9)
85     Stmt (9)
86     RETURN
87     Exp (9)
88     INT: 0
89     SEMI
90 RC

```

说明：1.2 分组的同学需要输出语法树注意科学计数法浮点数的正确转换。其他分组同学只要提示出错，而且不输出语法树即可。

4.3 D-3

输入

```

1 //there is a comment
2 /* this is also a comment
3 still in the comment
4 */
5 // /*this is a single comment*/
6 int main () {
7     int a,b /* this is okay*/;
8     a = 3;

```


21	DecList (7)
22	Dec (7)
23	VarDec (7)
24	ID: b
25	SEMI
26	StmtList (8)
27	Stmt (8)
28	Exp (8)
29	Exp (8)
30	ID: a
31	ASSIGNOP
32	Exp (8)
33	INT: 3
34	SEMI
35	StmtList (13)
36	Stmt (13)
37	Exp (13)
38	Exp (13)
39	ID: b
40	ASSIGNOP
41	Exp (13)
42	Exp (13)
43	ID: a
44	PLUS
45	Exp (13)
46	INT: 4
47	SEMI
48	StmtList (14)
49	Stmt (14)
50	WHILE
51	LP
52	Exp (14)

53	Exp (14)
54	ID: a
55	RELOP
56	Exp (14)
57	ID: b
58	RP
59	Stmt (14)
60	CompSt (14)
61	LC
62	StmtList (15)
63	Stmt (15)
64	Exp (15)
65	Exp (15)
66	ID: a
67	ASSIGNOP
68	Exp (15)
69	Exp (15)
70	ID: a
71	PLUS
72	Exp (15)
73	ID: a
74	SEMI
75	StmtList (16)
76	Stmt (16)
77	Exp (16)
78	Exp (16)
79	ID: b
80	ASSIGNOP
81	Exp (16)
82	Exp (16)
83	ID: b
84	PLUS

```

85                                     Exp (16)
86                                     INT: 4
87                                     SEMI
88                                     RC
89                               StmtList (18)
90                               Stmt (18)
91                               RETURN
92                               Exp (18)
93                               ID: a
94                               SEMI
95       RC

```

说明：1.3 分组的同学需要输出语法树，不能提示有语法错误；其他分组同学只需提示有错误，且不输出语法树即可。

5 E 组测试用例

本组测试用例共 6 个，针对不同分组进行测试。

5.1 E1.1

这组测试用例针对 1.1 分组的同学输入（E1-1）

```

1  int test_for_wrong_oct_number()
2  {
3      int _correct_oct_number_ = 006772;
4      int _decimal_number = 8092;
5      int _wrong_oct_number_ = 08092;
6      int _correct_oct_number2_ = 000000003345;
7      return 0;
8  }

```

输出

```

1  Error type A at Line 5: Illegal octal number '08092'

```

说明：仅 1.1 分组同学需要测试这个用例，针对错误的 8 进制数 08092，识别成错误类型 B 也可以。

输入（E1-2）

```
1 int test_for_wrong_dhex_number()  
2 {  
3     int xF33 = 0xe33;  
4     int _correct_dhex_number = 0xF598;  
5     int _wrong_dhex_number_ = 0xG598;  
6     int _wrong_dhex_number2_ = 0xFF598;  
7     int _correct_not_dhex_number3 = xF33;  
8     return xF33;  
9 }
```

输出

```
1 Error type A at Line 5: Illegal hexadecima number '0xG598'  
2 Error type A at Line 6: Illegal hexadecima number '0xFF598'
```

说明：仅 1.1 分组同学需要测试这个用例，针对错误的 16 进制数 0xG598 和 0xFF598，识别成错误类型 B 也可以。

5.2 E1.2

这组测试用例针对 1.2 分组的同学输入（E2-1）

```
1 float function()  
2 {  
3     float x4 = 34.e.3;  
4     float x1 = 34.e3.;  
5     float x2 = 34.e-1.2;  
6  
7     return x1 + x2;  
8 }
```

输出

```

1 Error type A at Line 3: There must be digits at both sides of the
   decimal point '34.'
2 Error type B at Line 4: syntax error
3 Error type A at Line 5: There must be digits at both sides of the
   decimal point '.2'

```

说明：仅 1.2 分组同学需要测试这个用例，针对错误的浮点数 34.e.3，34.e3.，以及 34.e-1.2，识别成错误类型 B 也可以。

输入（E2-2）

```

1 float function()
2 {
3     float a = 23.43E+3;
4     float b = 23.43E+.3;
5     float c = .e34;
6     return a;
7 }

```

输出

```

1 error type A at Line 4: Illegal floating point number '23.43E.3'
2 Error type B at Line 5: syntax error

```

说明：仅 1.2 分组同学需要测试这个用例，针对错误的浮点数 23.43E+.3 和.e34，识别成错误类型 B 也可以。

5.3 E1.3

这组测试用例针对 1.3 分组的同学输入（E3-1）

```

1 /* a starting comment !!!!!!!*/
2 // /* /* this is  okay*/ */
3 /*
4     /////\\\\\\\\\\\\\\/>
5     /* \\\\\\\\\\\'''''' this is nor okay!! */
6 */
7 int main () {

```

```

8      int a,b /* this is okay*/;
9      a = 3;
10     // \\\\\\/////\\\\\\\\/////\\\\\\\\*/
11     /* /* ///// a lot /////a\\\\\\\\\\/////
12         /////* /* /*
13
14     */
15     b = a + 4;
16     while(a < /* >_< ///// ///// */ b) {
17         a = a + a;
18         b = b + 4;
19     }
20     return a;
21 }
22 /* \***/*\*end of function/*\*/

```

输出

```

1 Error type B at Line 5: Multiple layers of comments.

```

说明：仅 1.3 分组同学需要测试这个用例，针对嵌套的多行注释。在第 6 行报 type A 错误也可以。

输入 (E3-2)

```

1  /**
2   * this comment ends. /* ///\\asfdlkajhsldf\\<>@#$
3  */
4  int main/*a comment is okay here 222*/() {
5      int a = //12;
6      /* a new comment*/
7      3;
8      int b = 0;
9      if (a > /* should still go*/ b) {
10         while(a > b) {
11             b = b + 1 //; this comma is lost!
12         }

```

```
13     }  
14     return b;  
15 }  
16 /* end ?~!
```

输出

```
1 Error type B at Line 11: Missing ';' '  
2 Error type A at Line 16: no match comment "/*".
```

说明：仅 1.3 分组同学需要测试这个用例。第 11 行注释掉了分号，也可以报在第 12 行；第 16 行缺少结束的注释，如果打印了语法树，或者程序异常终止、死循环无法退出等，则该用例不得分。不限定错误类型以及提示方式，但是出错位置必须限定在 16 行或者以后的位置；直接提示“未终止的注释”也可以。。

6 结束语

如果对本测试用例有任何疑议，可以写邮件与王珏助教联系，注意同时抄送给许老师。