

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	4
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	9
3.1	C-1	10
3.2	C-2	14
4	D 组测试用例	22
4.1	D-1	22
4.2	D-2	25
4.3	D-3	29
5	E 组测试用例	33
5.1	E1.1	33
5.2	E1.2	34
5.3	E1.3	35
6	结束语	37

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 float _func1(int X1)
2 {
3     float a_1_2;
4     int 6_Wrong;
5     return 0;
6 }
```

输出

```
1 Error type B at line 4: syntax error, near ';;'
```

说明：错误类型也可以是 A 类，或者一个 A 一个 B，但是只能在第 4 行。这里有一个非法标识符 6_Wrong，注意标识符可以以下划线开始，所以第 1 行正确。

1.2 A-2

输入

```
1 int sub(int a, int b)
2 {
3     int c = a + b - a * b / 3;
4     int d = !a && b || c >= 4;
5     int e = a & b;
6     return e;
7 }
```

输出

```
1 Error type A at line 5: mysteriously character '&'
```

说明：必须有 type A 错误；可以多报一个 type B 错误。这里有一个非法的符号 &。

1.3 A-3

输入

```
1 int add()  
2 {  
3     int a, b, c;  
4     c = (a + b) - (c * d / (-a));  
5     b = a || c;  
6     return b;  
7 }
```

输出

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

说明：第 4 行缺少匹配的左括号。

1.4 A-4

输入

```
1 int;  
2 int if()  
3 {  
4     int a;  
5     if (a > 0)  
6         return 1;  
7     else  
8         return 0;  
9 }
```

输出

```
1 Error type B at line 2: syntax error
```

说明：第 2 行保留字 if 不能作为标识符使用。注意第 1 行语句没有意义但是合法。

1.5 A-5

输入

```
1 int array(int i)
2 {
3     int a[3][3*3];
4     a[i][i] = 1;
5     while (i < 9)
6     {
7         a[0][i]= 1;
8         i = i + 1;
9     }
10 }
```

输出

```
1 Error type B at line 3: syntax error, near ' * '
```

说明：第3行数组定义格式错误，出现表达式。

1.6 A-6

输入

```
1 int a = 1;
2 int add(int b)
3 {
4     return a + b;
5 }
6 int main()
7 {
8     int sum;
9     sum = add(2);
10 }
```

输出

```
1 Error type B at line 1: syntax error, the global variable cannot be
   initialized in the definition.
```

说明：第 1 行定义全局变量时进行了初始化。

1.7 A-7

输入

```
1 struct Date
2 {
3     int year, month;
4     int day;
5 };
6 int main()
7 {
8     Date d[10];
9     d[0].year = 2020;
10 }
```

输出

```
1 Error type B at line 8: syntax error, missing 'struct'
```

说明：第 8 行定义结构体时缺少关键字 `struct`。

1.8 A-8

输入

```
1 int main()
2 {
3     struct A
4     {
5         int a;
6         float b;
7     };
}
```

```

8      struct A a[100];
9      int i = 0;
10     while (i < 100)
11     {
12         a[i].a = a[i].b = 0;
13         i = i + 1;
14     }
15 }

```

输出

```

1 Error type B at line 7: syntax error, near ';'

```

说明：第 7 行，在方法体内定义结构体时缺少声明的变量名，报在第 3 行也可以。

1.9 A-9

输入

```

1 int max(int x, int y)
2 {
3     int t;
4     if (x > y)
5         t = x;
6     else
7         t = y
8     return t;
9 }
10 int main()
11 {
12     int a = 6, b = 4, c = 5;
13     int maxs = max(a, max(b, c));
14 }

```

输出

```

1 Error type B at line 7: syntax error, missing ';'

```

说明：第 7 行漏写了句尾的分号。也可在第 8 行报错。

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、多报错误或者打印语法树都会导致扣分。

2.1 B-1

输入

```
1 struct Student
2 {
3     int ID,
4     float score;
5 }class[2];
6
7 struct Student test(struct Student a)
8 {
9     return a;
10 }
11
12 int main()
13 {
14     int i = 0, n = 2;
15     float sum = 0;
16     struct Student classes[n][10];
17     class[0].ID = 1;
18     class[0].score = 90;
19     class[1].ID = 2;
20     class[1].score = .85 * 100;
21     test(class[0]);
22     while (i < 2)
23     {
24         sum += class[i].score;
```

```

25         i = i + 1;
26     }
27     return 0;
28 }

```

输出

```

1 Error type B at line 3: syntax error, near ',',
2 Error type B at line 16: syntax error, near '['
3 Error type A at line 20: illegal float number '.85'
4 Error type B at line 24: syntax error, near '+'

```

说明：第 3 行末尾分号错写成逗号，也可以报错在第 4 行；第 16 行用变量定义数组；第 20 行小数点前必须有数字，也可识别成 B 类型错误，或多报一个 B 类错误；第 24 行使用了未定义的操作符 +=。

2.2 B-2

输入

```

1 int shsort(int s[4], int n)
2 {
3     int i, j, d;
4     d = n / 2;
5     while(d >= 1)
6     {
7         i = d + 1;
8         while (i <= n)
9         {
10            s[0] = s[i];
11            j = i - d;
12            while((j > 0) && (s[0] < s[j]))
13            {
14                s[j + d] = s[j];
15                j = j - d;
16            }

```



```

17         s[j + d] = s[0];
18         i ++;
19     }
20     d = d / 2;
21 }
22 return 0;
23 }
24 int main()
25 {
26     int a[4];
27     a[0] = 2;
28     a[1] = 4;
29     a[2] = 3;
30     a[3]] = 1;
31     int n = 4;
32     shsort(a, n);
33     return 0;
34 }

```

输出

```

1 Error type B at line 5: syntax error, near '>'
2 Error type B at line 18: syntax error, near '+'
3 Error type B at line 30: syntax error, near ']'
4 Error type B at line 31: syntax error, the local variable cannot be
   defined after statements

```

说明：第 5 行 `>=` 号中间多了一个空格；第 18 行使用了未定义符号 `++`；第 30 行多打了一个右括号`]`；第 31 行语句之后不能再出现变量定义。

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 Student
2 {
3     int SID;
4     int age;
5     struct Teacher
6     {
7         int TID;
8     } t;
9 };
10
11 int main()
12 {
13     struct Student s1;
14     s1.SID = 1;
15     s1.age = 18;
16     s1.t.TID = 3;
17     return s1;
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: Student
9           LC
10          DefList (3)
```

```

11      Def (3)
12          Specifier (3)
13              TYPE: int
14          DecList (3)
15              Dec (3)
16              VarDec (3)
17              ID: SID
18          SEMI
19      DefList (4)
20          Def (4)
21              Specifier (4)
22                  TYPE: int
23              DecList (4)
24                  Dec (4)
25                  VarDec (4)
26                  ID: age
27          SEMI
28      DefList (5)
29          Def (5)
30              Specifier (5)
31                  StructSpecifier (5)
32                      STRUCT
33                      OptTag (5)
34                          ID: Teacher
35                      LC
36                      DefList (7)
37                          Def (7)
38                              Specifier (7)
39                                  TYPE: int
40                                  DecList (7)
41                                      Dec (7)
42                                          VarDec (7)

```

```

43                                     ID: TID
44                                     SEMI
45                                     RC
46                                     DecList (8)
47                                     Dec (8)
48                                     VarDec (8)
49                                     ID: t
50                                     SEMI
51                                     RC
52                                     SEMI
53 ExtDefList (11)
54     ExtDef (11)
55         Specifier (11)
56             TYPE: int
57         FunDec (11)
58             ID: main
59             LP
60             RP
61         CompSt (12)
62             LC
63             DefList (13)
64                 Def (13)
65                     Specifier (13)
66                         StructSpecifier (13)
67                             STRUCT
68                             Tag (13)
69                                 ID: Student
70                         DecList (13)
71                             Dec (13)
72                             VarDec (13)
73                                 ID: s1
74                                     SEMI

```

75	StmtList (14)
76	Stmt (14)
77	Exp (14)
78	Exp (14)
79	Exp (14)
80	ID: s1
81	DOT
82	ID: SID
83	ASSIGNOP
84	Exp (14)
85	INT: 1
86	SEMI
87	StmtList (15)
88	Stmt (15)
89	Exp (15)
90	Exp (15)
91	Exp (15)
92	ID: s1
93	DOT
94	ID: age
95	ASSIGNOP
96	Exp (15)
97	INT: 18
98	SEMI
99	StmtList (16)
100	Stmt (16)
101	Exp (16)
102	Exp (16)
103	Exp (16)
104	Exp (16)
105	ID: s1
106	DOT

```

107         ID: t
108         DOT
109         ID: TID
110         ASSIGNOP
111         Exp (16)
112         INT: 3
113         SEMI
114         StmtList (17)
115         Stmt (17)
116         RETURN
117         Exp (17)
118         ID: s1
119         SEMI
120     RC

```

说明：使用的空格可以用 Tab 替换，注意缩进

3.2 C-2

输入

```

1  int MatrixMax(int a[3][4])
2  {
3      int i=0,j=0;
4      int max,max_i=0,max_j=0;
5      max = a[0][0];
6      while(i < 3)
7      {
8          while(j < 4)
9          {
10             if(a[i][j] > max)
11             {
12                 max=a[i][j];
13                 max_i=i;
14                 max_j=j;

```

```

15         }
16         j = j + 1;
17     }
18     i = i + 1;
19 }
20 return 0;
21 }

```

输出

```

1 Program (1)
2   ExtDefList (1)
3     ExtDef (1)
4       Specifier (1)
5         TYPE: int
6       FunDec (1)
7         ID: MatrixMax
8         LP
9         VarList (1)
10          ParamDec (1)
11            Specifier (1)
12              TYPE: int
13          VarDec (1)
14            VarDec (1)
15              VarDec (1)
16                ID: a
17                LB
18                INT: 3
19                RB
20            LB
21            INT: 4
22            RB
23          RP
24    CompSt (2)

```

```

25      LC
26      DefList (3)
27          Def (3)
28              Specifier (3)
29                  TYPE: int
30              DecList (3)
31                  Dec (3)
32                      VarDec (3)
33                          ID: i
34                          ASSIGNOP
35                          Exp (3)
36                              INT: 0
37                      COMMA
38                      DecList (3)
39                          Dec (3)
40                              VarDec (3)
41                                  ID: j
42                                  ASSIGNOP
43                                  Exp (3)
44                                      INT: 0
45          SEMI
46      DefList (4)
47          Def (4)
48              Specifier (4)
49                  TYPE: int
50              DecList (4)
51                  Dec (4)
52                      VarDec (4)
53                          ID: max
54                      COMMA
55                      DecList (4)
56                          Dec (4)

```



```

57         VarDec (4)
58             ID: max_i
59         ASSIGNOP
60         Exp (4)
61             INT: 0
62     COMMA
63     DecList (4)
64         Dec (4)
65             VarDec (4)
66                 ID: max_j
67             ASSIGNOP
68             Exp (4)
69                 INT: 0
70     SEMI
71 StmtList (5)
72     Stmt (5)
73         Exp (5)
74             Exp (5)
75                 ID: max
76             ASSIGNOP
77             Exp (5)
78                 Exp (5)
79                     Exp (5)
80                         ID: a
81                     LB
82                     Exp (5)
83                         INT: 0
84                     RB
85                 LB
86                 Exp (5)
87                     INT: 0
88                 RB

```

```

89         SEMI
90     StmtList (6)
91     Stmt (6)
92         WHILE
93         LP
94         Exp (6)
95             Exp (6)
96             ID: i
97         RELOP
98         Exp (6)
99             INT: 3
100        RP
101    Stmt (7)
102        CompSt (7)
103        LC
104        StmtList (8)
105        Stmt (8)
106            WHILE
107            LP
108            Exp (8)
109                Exp (8)
110                ID: j
111            RELOP
112            Exp (8)
113                INT: 4
114            RP
115        Stmt (9)
116            CompSt (9)
117            LC
118            StmtList (10)
119                Stmt (10)
120                    IF

```

121	LP
122	Exp (10)
123	Exp (10)
124	Exp (10)
125	Exp (10)
126	ID: a
127	LB
128	Exp (10)
129	ID: i
130	RB
131	LB
132	Exp (10)
133	ID: j
134	RB
135	RELOP
136	Exp (10)
137	ID: max
138	RP
139	Stmt (11)
140	CompSt (11)
141	LC
142	StmtList (12)
143	Stmt (12)
144	Exp (12)
145	Exp (12)
146	ID: max
147	ASSIGNOP
148	Exp (12)
149	Exp (12)
150	Exp (12)
151	ID: a
152	LB

153	Exp (12)
154	ID: i
155	RB
156	LB
157	Exp (12)
158	ID: j
159	RB
160	SEMI
161	StmtList (13)
162	Stmt (13)
163	Exp (13)
164	Exp (13)
165	ID: max_i
166	ASSIGNOP
167	Exp (13)
168	ID: i
169	SEMI
170	StmtList (14)
171	Stmt (14)
172	Exp (14)
173	Exp (14)
174	ID: max_j
175	ASSIGNOP
176	Exp (14)
177	ID: j
178	SEMI
179	RC
180	StmtList (16)
181	Stmt (16)
182	Exp (16)
183	Exp (16)
184	ID: j

185	ASSIGNOP
186	Exp (16)
187	Exp (16)
188	ID: j
189	PLUS
190	Exp (16)
191	INT: 1
192	SEMI
193	RC
194	StmtList (18)
195	Stmt (18)
196	Exp (18)
197	Exp (18)
198	ID: i
199	ASSIGNOP
200	Exp (18)
201	Exp (18)
202	ID: i
203	PLUS
204	Exp (18)
205	INT: 1
206	SEMI
207	RC
208	StmtList (20)
209	Stmt (20)
210	RETURN
211	Exp (20)
212	INT: 0
213	SEMI
214	RC

说明：考察对数组的翻译。

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；其他分组的同学只需要在对应位置提示错误即可，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_ = 947;
4     int _oct_ = 0705;
5     int _dhex_ = 0xFFaBc - _oct_;
6     int _result_ = - _dhex_ + _oct_ * ( _dec_ - 0X23fD );
7 }
```

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: 947
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: 453
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: 1047228
48                     MINUS

```

```

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: 9213
79     RP
80 SEMI

```


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（453、1047228 和 9213）；其他分组的同学只要提示相应的错误，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 = 3.80e-7;  
4     float X_2 = 345.2e+4;  
5     float X_3 = 2.76E+3;  
6     float X_4 = .994E-2;  
7     float X_5 = 11.e1;  
8     float X_6 = -0.4E-03;  
9     float result = (15.E-1 + X_3) + X_6;  
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: 0.000000
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: 3452000.000000
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: 2760.000000

```

```

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: 0.009940
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: 110.000000
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     MINUS
83     Exp (8)
84     FLOAT: 0.000400
85 SEMI
86 DefList (9)
87     Def (9)
88     Specifier (9)
89     TYPE: float
90     DecList (9)
91     Dec (9)
92     VarDec (9)
93     ID: result
94     ASSIGNOP
95     Exp (9)
96     Exp (9)
97     LP
98     Exp (9)
99     Exp (9)
100     FLOAT: 1.500000
101     PLUS
102     Exp (9)
103     ID: X_3
104     RP
105     PLUS
106     Exp (9)
107     ID: X_6
108 SEMI
109 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。其它分组同学

只需要提示相应错误，而且不输出语法树即可。

4.3 D-3

输入

```

1  /*
2  **This is a test for comments
3  ///////////////////////////////////
4  /
5  */
6  int /**/F(int n) // {%^@#~~~ //
7  {
8      /****8
9      int i = 0;
10     ****\*/
11     if(n == 0)
12         //
13         return 1; /*}   \\
14     \\
15     */
16     return n * F(n - 1);
17 }
18 int main() //b > a comments*//\///*
19 {
20     int n, r;
21     n = 10;
22     r = F(n);//\/*//\/*//\//\/*//\
23     return /*ends\//\//\//\/*/0;
24 }

```

输出

```

1 Program (6)
2   ExtDefList (6)
3     ExtDef (6)

```

```

4      Specifier (6)
5          TYPE: int
6      FunDec (6)
7          ID: F
8          LP
9          VarList (6)
10             ParamDec (6)
11                 Specifier (6)
12                     TYPE: int
13             VarDec (6)
14                 ID: n
15             RP
16      CompSt (7)
17          LC
18          StmtList (11)
19              Stmt (11)
20                  IF
21                  LP
22                  Exp (11)
23                      Exp (11)
24                          ID: n
25                      RELOP
26                      Exp (11)
27                          INT: 0
28                  RP
29                  Stmt (13)
30                      RETURN
31                      Exp (13)
32                          INT: 1
33                      SEMI
34                  StmtList (16)
35                      Stmt (16)

```

```

36         RETURN
37     Exp (16)
38         Exp (16)
39         ID: n
40     STAR
41     Exp (16)
42     ID: F
43     LP
44     Args (16)
45         Exp (16)
46         Exp (16)
47         ID: n
48     MINUS
49     Exp (16)
50         INT: 1
51     RP
52     SEMI
53     RC
54 ExtDefList (18)
55     ExtDef (18)
56         Specifier (18)
57             TYPE: int
58     FunDec (18)
59         ID: main
60         LP
61         RP
62     CompSt (19)
63         LC
64         DefList (20)
65             Def (20)
66                 Specifier (20)
67                     TYPE: int

```

```

68         DecList (20)
69         Dec (20)
70         VarDec (20)
71         ID: n
72         COMMA
73         DecList (20)
74         Dec (20)
75         VarDec (20)
76         ID: r
77     SEMI
78 StmtList (21)
79     Stmt (21)
80     Exp (21)
81     Exp (21)
82     ID: n
83     ASSIGNOP
84     Exp (21)
85     INT: 10
86     SEMI
87 StmtList (22)
88     Stmt (22)
89     Exp (22)
90     Exp (22)
91     ID: r
92     ASSIGNOP
93     Exp (22)
94     ID: F
95     LP
96     Args (22)
97     Exp (22)
98     ID: n
99     RP

```



```

100          SEMI
101      StmtList (23)
102          Stmt (23)
103              RETURN
104              Exp (23)
105                  INT: 0
106              SEMI
107  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；其他分组同学只需要提示相应错误，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_ = 003456;
4      int _decimal_number = 748;
5      int _wrong_oct_number_ = 0748;
6      return 0;
7  }

```

输出

```

1  Error type A at Line 5: Illegal octal number "0748"

```

说明：仅 1.1 分组的同学需要测试这个用例，针对错误的 8 进制数 0748，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 _correct_dhex_number_ = 0xf29C0;
4     int _wrong_dhex_number_ = 0xCd0G;
5     int _correct_dhex_number2_ = 0x000abCD;
6     int _wrong_dhex_number2_ = 0xx346f;
7 }

```

输出

```

1 Error type A at Line 4: Illegal hexadecimal number "0xCd0G"
2 Error type A at Line 6: Illegal hexadecimal number "0xx346f"

```

说明：仅 1.1 分组的同学需要测试这个用例，针对错误的 16 进制数 0xCd0G 与 0xx346f，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 x1 = 002.45E10;
4     float x2 = 45.e-1.1;
5     float x3 = .45e-3;
6     float x4 = 34.34E+1.2;
7     return x1 + x3;
8 }

```

输出

```

1 Error type A at Line 4: Illegal float number "45.e-1.1"
2 Error type A at Line 6: Illegal float number "34.34E+1.2"

```

说明：仅 1.2 分组的同学需要测试这个用例，针对错误浮点数 45.e-1.1 和 34.34E+1.2，识别成错误类型 B 也可以。

输入 (E2-2)

```
1 float function()  
2 {  
3     float a = 12.34e+;  
4     float b = .e;  
5     float c = .e-2;  
6 }
```

输出

```
1 | Error type A at Line 3: Illegal float number "12.34e+"
2 | Error type B at line 4: syntax error, unexpected DOT
3 | Error type A at Line 5: Illegal float number ".e-2"
```

说明：仅 1.2 分组的同学需要测试这个用例，针对错误浮点数 12.34e+、.e 和.e-2，识别成错误类型 B 和 A 都可以。

5.3 E1.3

这组测试用例针对 1.3 分组的同学。

输入 (E3-1)

```

1  /****
2  * @param int a
3  */
4  int main()
5  {
6      /*      TODO
7              definition 12345
8          */
9      int day,x1,x2;
10     day = 9 /* this is okay*/;
11     x2 = 1;
12     // \\\\//\\\\\\\\~\\//\\\\\\\\\\*/
13     while(day> /*>=<*/ 0)
14     {

```

```

15     x1=(x2+1)*2;
16     /*
17         //////////>
18         /* \\\\' this is not okay!! */
19     */
20     x2=x1;
21     day = day - 1;
22     }// assert(day == ?);
23     return /*-1;*/0;
24 }

```

输出

```

1 Error type B at Line 19: syntax error, near '*/'

```

说明：仅 1.3 分组的同学需要测试这个用例，针对嵌套的多行注释。19 行出现了多余的注释符号。识别成类型 A 也可以，需要合理的错误提示。

输入 (E3-2)

```

1  /**comment
2  still
3  */
4  /* ///\\asfdlkajhsldf\\<>@#&
5  */
6  int main() //a function
7  {
8      float h, s; int i;
9      h = s = 100 //;
10     h = h / 2; // first
11     i = 2;
12     /* a new comment*/
13     while (i <= 10) //\\//\\/*
14     {
15         s = s + 2 * h;
16         h = h / 2 /*\\//\\//\\*\\//\\*****\\*/;

```

```
17         i = i + 1;
18     }
19     return 0;
20 }
21 /**end ~??
```

输出

```
1 Error type B at Line 9: missing ';'
2 Error type B at Line 21: syntax error, no match comment "/*"
```

说明：仅 1.3 分组的同学需要测试这个用例。第 9 行语句漏写了分号，也可以报在第 10 行；第 21 行第缺少结束的注释，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 21 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，可以写邮件与李聪助教或陈紫琦助教联系，注意同时抄送给许老师。