编译原理第一次实验测试用例：目录

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

```
int func_A1()
{
    int _3_test;
    float 6_wrong;
}
```

输出

```
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 3 行正确。

1.2 A-2

输入

```
int sub()
{
    int a, b,c ;
    c = 2;
    a = b / c + 3 * a + 12;
    b = !a && 4 >= 4 || a != 6;
    c = a % b;
    return b;
}
```

输出

```
Error type B at line 7: 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

```
int method(){
{
    int a, b, c;{
    c = a + b * a + 4;
    b = a && c;
    return b; }
}
```

输出

```
Error type B at line 8: missing '}'
```

说明：第 8 行缺少匹配的括号，可以报错在第 7 行，或者开始的括号所在的 1、2 行。

1.4 A-4

输入

```
int main(){
    int a;
    a = a + 1;
    a = a - 2;
    return a
}
```

输出

```
Error type B at line 6: syntax error, before '}'
```

说明：第 5 行缺少分号，可以报错在第 5 行。

1.5 A-5

输入

```
int array (int a, int b)
{
    int a1, b1[3], c[-3];
    b1[a1] = b;
    b1[1] = a + b1[a];
}
```

输出

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

说明：第 3 行数组定义格式错误，出现负数。

1.6 A-6

输入

```
int Array_Def()
{
    int i, a[10][2], b;
    i = 0;
    while (i < b) {
        a[i][3][3+1,2] = b;
        a[5][1] = i;
    }
}
```

输出

```
Error type B at line 6: syntax error, near ','
```

说明：第 6 行数组下标访问时格式不正确，中括号内出现逗号。

1.7 A-7

输入

```
struct Product{
    int name;
    int price;
    int date;
};
struct{
    float test;
};
struct;
int main()
{
    struct Product a;
    a.name = a.price = 2;
    a.data = 1;
}
```

输出

```
Error type B at line 9: syntax error, near '; '
```

说明：第 9 行定义结构体时缺少结构体名称。

1.8 A-8

输入

```
struct Product{
    int name;
    int price;
    int date;
};
int main()
{
    struct Inside_Function{
        int name;
```

```
10          int price;
11      };
12      struct Product a;
13      a.name = a.price = 2;
14      a.data = 1;
15  }
```

输出

```
1  Error type B at line 11: syntax error, near '}'
```

说明：第 11 行，在方法体内定义结构体时缺少声明的变量名。

1.9 A-9

输入

```
1  int function_1(int a) {
2      return a + 3;
3  }
4
5  float function_2(int b)
6  {
7      int c;
8      c = b * 3 ;
9      if(c == 12) return 3.5;
10     else return 1.2;
11 }
12
13 float functionMain(int b)
14 {
15     int a = b;
16     return function_2(function_1((2 * b) + 3));
17 }
```

输出

```
Error type B at line 16: syntax error, near ') '
```

说明：第 10 行函数调用时多了一个右括号。

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、多报错误或者打印语法树都会导致扣分。

2.1 B-1

输入

```
struct Test_Product{
    int name;
    float type,
}p[10];

float Test_Function(struct Test_Product test) {
    if(test.name >= 3 || test.type == 2.3) {
        return 3.2;
    } else{
        return test.type /2 * 7 + test,type;
    }
}

int main()
{
    int name[10];
    float type[10];
    int result;
    int i, N;
    while(i !  = N){
              p[i].name = name[i];
          p[i].type = (type[i * 3] - 2.2) / 2.3;
```

```
23                 i = i + 1;
24     }
25
26     while(i == 0) {
27                 N = N + p[i].name;
28         p[i].type = p[1 0].type * p[9].type;
29     }
30     return N;
31 }
```

输出

```
1 Error type B at line 4: syntax error, near '}'
2 Error type B at line 10: syntax error, near ','
3 Error type B at line 20: syntax error, near '!'
4 Error type B at line 28: syntax error, near '0'
```

说明：第 3 行末尾分号错写成逗号，也可以报错在第 4 行；第 10 行结构体域访问. 错写成,；第 20 行不等号中多了一个空格；第 28 行数组访问下标 10 之间多了一个空格。

2.2 B-2

输入

```
1  struct Result{
2      int i;
3      int j;
4  }text1, text2; text3;
5
6  int gcd(int a, int b) {
7      if (b == 0)
8          return a;
9      else
10         return gcd(b, a % b);
11 }
12
```

```
13 int main() {
14     struct Result six;
15     int x1, x2;
16 
17     x1 = 1;
18     x2 = 7;
19     six.i = 7;
20     six.j = six.i + text1.i - 3;
21     if (six.j < text2.j) {
22         retun six.j + 3;
23     }
24 
25     x1 = 14;
26     x2 = 21;
27     while(x1 < x2) {
28         x1 = x1+ gcd(x1, x2);
29     }
30 
31     return x1 > 3 x2;
32 }
```

输出

```
1 Error type B at line 4: syntax error, near 'text3'
2 Error type A at line 10: Mysterious character '%'
3 Error type B at line 22: syntax error, near 'six'
4 Error type B at line 31: syntax error, near 'x2'
```

说明：第 4 行多个结构体变量声明中 text2 与 text3 中间逗号错写为分号；第 10 行使用未定义符号%, 也可以多报一个 B 类型错误；第 22 行 return 拼写错误；第 29 行多出现一个数字 3。

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

```
struct Test_type{
    int id;
    float value;
}test1;

int main() {
    struct Test_type_type{
        struct Test_type nested1;
        struct Test_type_nested{
            int idd;
        }nested2;
    }type;

    type.nested1 = test1;
    type.nested2.idd = type.nested1.id;
    return type.nested2.idd * type.nested1.id;
}
```

输出

```
Program(1)
  ExtDefList(1)
    ExtDef(1)
      Specifier(1)
        StructSpecifier(1)
          STRUCT
          OptTag(1)
            ID:Test_type
          LC
          DefList(2)
            Def(2)
```

```
            Specifier(2)
              TYPE:int
            DecList(2)
              Dec(2)
                VarDec(2)
                  ID:id
              SEMI
          DefList(3)
            Def(3)
              Specifier(3)
                TYPE:float
              DecList(3)
                Dec(3)
                  VarDec(3)
                    ID:value
              SEMI
        RC
    ExtDecList(4)
      VarDec(4)
        ID:test1
    SEMI
  ExtDefList(6)
    ExtDef(6)
      Specifier(6)
        TYPE:int
      FunDec(6)
        ID:main
        LP
        RP
      CompSt(6)
        LC
        DefList(7)
```

```
        Def(7)
          Specifier(7)
            StructSpecifier(7)
              STRUCT
              OptTag(7)
                ID:Test_type_type
              LC
              DefList(8)
                Def(8)
                  Specifier(8)
                    StructSpecifier(8)
                      STRUCT
                      Tag(8)
                        ID:Test_type
                  DecList(8)
                    Dec(8)
                      VarDec(8)
                        ID:nested1
                  SEMI
                DefList(9)
                  Def(9)
                    Specifier(9)
                      StructSpecifier(9)
                        STRUCT
                        OptTag(9)
                          ID:Test_type_nested
                        LC
                        DefList(10)
                          Def(10)
                            Specifier(10)
                              TYPE:int
                            DecList(10)
```

```
                                Dec(10)
                                  VarDec(10)
                                    ID:idd
                               SEMI
                           RC
                       DecList(11)
                         Dec(11)
                           VarDec(11)
                             ID:nested2
                       SEMI
                   RC
               DecList(12)
                 Dec(12)
                   VarDec(12)
                     ID:type
               SEMI
           StmtList(14)
             Stmt(14)
               Exp(14)
                 Exp(14)
                   Exp(14)
                     ID:type
                   DOT
                   ID:nested1
                 ASSIGNOP
                 Exp(14)
                   ID:test1
               SEMI
             StmtList(15)
               Stmt (15)
                 Exp (15)
                   Exp (15)
```

```
        Exp (15)
          Exp (15)
            ID: type
          DOT
          ID: nested2
        DOT
        ID: idd
      ASSIGNOP
      Exp (15)
        Exp (15)
          Exp (15)
            ID: type
          DOT
          ID: nested1
        DOT
        ID: id
    SEMI

  StmtList(16)
    Stmt (16)
      RETURN
      Exp (16)
        Exp (16)
          Exp (16)
            Exp (16)
              ID: type
            DOT
            ID: nested2
          DOT
          ID: idd
        STAR
        Exp (16)
```

```
                        Exp (16)
                          Exp (16)
                            ID: type
                          DOT
                          ID: nested1
                        DOT
                        ID: id
                    SEMI
            RC
```

说明：使用的空格可以用 Tab 替换，注意缩进

3.2 C-2

输入

```
int map[10][10];
int dis[10][10];
int floyd(int num){
    int i,j,k;
    i = j = k = 0;
    while(i < num) {
        while(j < num) {
            while(k < num) {
                    if (dis[j][i] + dis[i][k] < dis[j][k]) {
                        i = j + k;
                            }
                        k = k + 1;
            }
            j = j + 1;
        }
        i = i + 1;
    }
    return 1;
}
```

```

int main(){
    int number, i;
    number = 3;
    dis = map;
    floyd(number);
    return number;
}
```

输出

```
Program (1)
  ExtDefList (1)
    ExtDef (1)
      Specifier (1)
        TYPE: int
      ExtDecList (1)
        VarDec (1)
          VarDec (1)
            VarDec (1)
              ID: map
            LB
            INT: 10
            RB
          LB
          INT: 10
          RB
      SEMI
    ExtDefList (2)
      ExtDef (2)
        Specifier (2)
          TYPE: int
        ExtDecList (2)
          VarDec (2)
```

```
          VarDec (2)
            VarDec (2)
              ID: dis
            LB
            INT: 10
            RB
          LB
          INT: 10
          RB
      SEMI
    ExtDefList (3)
      ExtDef (3)
        Specifier (3)
          TYPE: int
        FunDec (3)
          ID: floyd
          LP
          VarList (3)
            ParamDec (3)
              Specifier (3)
                TYPE: int
              VarDec (3)
                ID: num
          RP
        CompSt (3)
          LC
          DefList (4)
            Def (4)
              Specifier (4)
                TYPE: int
              DecList (4)
                Dec (4)
```

```
                VarDec (4)
                  ID: i
              COMMA
              DecList (4)
                Dec (4)
                  VarDec (4)
                    ID: j
                COMMA
                DecList (4)
                  Dec (4)
                    VarDec (4)
                      ID: k
          SEMI
        StmtList (5)
          Stmt (5)
            Exp (5)
              Exp (5)
                ID: i
              ASSIGNOP
              Exp (5)
                Exp (5)
                  ID: j
                ASSIGNOP
                Exp (5)
                  Exp (5)
                    ID: k
                  ASSIGNOP
                  Exp (5)
                    INT: 0
            SEMI
          StmtList (6)
            Stmt (6)
```

```
                    WHILE
                    LP
                    Exp (6)
                      Exp (6)
                        ID: i
                      RELOP
                      Exp (6)
                        ID: num
                    RP
                    Stmt (6)
                      CompSt (6)
                        LC
                        StmtList (7)
                          Stmt (7)
                            WHILE
                            LP
                            Exp (7)
                              Exp (7)
                                ID: j
                              RELOP
                              Exp (7)
                                ID: num
                            RP
                            Stmt (7)
                              CompSt (7)
                                LC
                                StmtList (8)
                                  Stmt (8)
                                    WHILE
                                    LP
                                    Exp (8)
                                      Exp (8)
```

```
      ID: k
    RELOP
    Exp (8)
      ID: num
  RP
  Stmt (8)
    CompSt (8)
      LC
      StmtList (9)
        Stmt (9)
          IF
          LP
          Exp (9)
            Exp (9)
              Exp (9)
                Exp (9)
                  Exp (9)
                    ID: dis
                  LB
                  Exp (9)
                    ID: j
                  RB
                LB
                Exp (9)
                  ID: i
                RB
              PLUS
              Exp (9)
                Exp (9)
                  Exp (9)
                    ID: dis
                  LB
```

```
          Exp (9)
            ID: i
          RB
        LB
        Exp (9)
          ID: k
        RB
      RELOP
      Exp (9)
        Exp (9)
          Exp (9)
            ID: dis
          LB
          Exp (9)
            ID: j
          RB
        LB
        Exp (9)
          ID: k
        RB
    RP
    Stmt (9)
      CompSt (9)
        LC
        StmtList (10)
          Stmt (10)
            Exp (10)
              Exp (10)
                ID: i
              ASSIGNOP
              Exp (10)
                Exp (10)
```

```
                                        ID: j
                                      PLUS
                                      Exp (10)
                                        ID: k
                                   SEMI
                                RC
                            StmtList (12)
                              Stmt (12)
                                Exp (12)
                                  Exp (12)
                                    ID: k
                                  ASSIGNOP
                                  Exp (12)
                                    Exp (12)
                                      ID: k
                                    PLUS
                                    Exp (12)
                                      INT: 1
                                SEMI
                          RC
                      StmtList (14)
                        Stmt (14)
                          Exp (14)
                            Exp (14)
                              ID: j
                            ASSIGNOP
                            Exp (14)
                              Exp (14)
                                ID: j
                              PLUS
                              Exp (14)
                                INT: 1
```

```
                                    SEMI
                              RC
                        StmtList (16)
                          Stmt (16)
                            Exp (16)
                              Exp (16)
                                ID: i
                              ASSIGNOP
                              Exp (16)
                                Exp (16)
                                  ID: i
                                PLUS
                                Exp (16)
                                  INT: 1
                            SEMI
                    RC
               StmtList (18)
                 Stmt (18)
                   RETURN
                   Exp (18)
                     INT: 1
                   SEMI
        RC
  ExtDefList (21)
    ExtDef (21)
      Specifier (21)
        TYPE: int
      FunDec (21)
        ID: main
        LP
        RP
      CompSt (21)
```

```
248                 LC
249                 DefList (22)
250                   Def (22)
251                     Specifier (22)
252                       TYPE: int
253                     DecList (22)
254                       Dec (22)
255                         VarDec (22)
256                           ID: number
257                       COMMA
258                       DecList (22)
259                         Dec (22)
260                           VarDec (22)
261                             ID: i
262                     SEMI
263                 StmtList (23)
264                   Stmt (23)
265                     Exp (23)
266                       Exp (23)
267                         ID: number
268                       ASSIGNOP
269                       Exp (23)
270                         INT: 3
271                     SEMI
272                   StmtList (24)
273                     Stmt (24)
274                       Exp (24)
275                         Exp (24)
276                           ID: dis
277                         ASSIGNOP
278                         Exp (24)
279                           ID: map
```

```
                    SEMI
                  StmtList (25)
                    Stmt (25)
                      Exp (25)
                        ID: floyd
                        LP
                        Args (25)
                          Exp (25)
                            ID: number
                        RP
                      SEMI
                    StmtList (26)
                      Stmt (26)
                        RETURN
                        Exp (26)
                          ID: number
                        SEMI
              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

```
int func_test()
{
    int _dec_ = 312;
    int _oct_ = 0733;
```

```
    int _dhex_ = 0xC612D + _oct_;
    int _result_ = - _dhex_ + _oct_ * ( _dec_ - 0X3aB );
}
```

输出

```
Program (1)
  ExtDefList (1)
    ExtDef (1)
      Specifier (1)
        TYPE: int
      FunDec (1)
        ID: func_test
        LP
        RP
      CompSt (2)
        LC
        DefList (3)
          Def (3)
            Specifier (3)
              TYPE: int
            DecList (3)
              Dec (3)
                VarDec (3)
                  ID: _dec_
                ASSIGNOP
                Exp (3)
                  INT: 312
            SEMI
          DefList (4)
            Def (4)
              Specifier (4)
                TYPE: int
              DecList (4)
```

```
    Dec (4)
      VarDec (4)
        ID: _oct_
      ASSIGNOP
      Exp (4)
        INT: 475
  SEMI
DefList (5)
  Def (5)
    Specifier (5)
      TYPE: int
    DecList (5)
      Dec (5)
        VarDec (5)
          ID: _dhex_
        ASSIGNOP
        Exp (5)
          Exp (5)
            INT: 811309
          PLUS
          Exp (5)
            ID: _oct_
    SEMI
  DefList (6)
    Def (6)
      Specifier (6)
        TYPE: int
      DecList (6)
        Dec (6)
          VarDec (6)
            ID: _result_
          ASSIGNOP
```

```
                        Exp (6)
                          Exp (6)
                            MINUS
                            Exp (6)
                              ID: _dhex_
                          PLUS
                          Exp (6)
                            Exp (6)
                              ID: _oct_
                            STAR
                            Exp (6)
                              LP
                              Exp (6)
                                Exp (6)
                                  ID: _dec_
                                MINUS
                                Exp (6)
                                  INT: 939
                              RP
                    SEMI
        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（475、811309 和 939）；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

```
int float_test()
{
    float X_1 = 1.E3;
    float X_2 = 2.0e-4;
    float X_3 = 12.3E-1;
    float X_4 = 123.4e2;
```

```
    float X_5 = .123E2;
    float X_6 = 1.e+1;
}
```

输出

```
Program (1)
  ExtDefList (1)
    ExtDef (1)
      Specifier (1)
        TYPE: int
      FunDec (1)
        ID: float_test
        LP
        RP
      CompSt (2)
        LC
        DefList (3)
          Def (3)
            Specifier (3)
              TYPE: float
            DecList (3)
              Dec (3)
                VarDec (3)
                  ID: X_1
                ASSIGNOP
                Exp (3)
                  FLOAT: 1000.000000
            SEMI
          DefList (4)
            Def (4)
              Specifier (4)
                TYPE: float
              DecList (4)
```

```
    Dec (4)
      VarDec (4)
        ID: X_2
      ASSIGNOP
      Exp (4)
        FLOAT: 0.000200
  SEMI
DefList (5)
  Def (5)
    Specifier (5)
      TYPE: float
    DecList (5)
      Dec (5)
        VarDec (5)
          ID: X_3
        ASSIGNOP
        Exp (5)
          FLOAT: 1.230000
    SEMI
  DefList (6)
    Def (6)
      Specifier (6)
        TYPE: float
      DecList (6)
        Dec (6)
          VarDec (6)
            ID: X_4
          ASSIGNOP
          Exp (6)
            FLOAT: 12340.000000
      SEMI
    DefList (7)
```

```
                    Def (7)
                      Specifier (7)
                        TYPE: float
                      DecList (7)
                        Dec (7)
                          VarDec (7)
                            ID: X_5
                          ASSIGNOP
                          Exp (7)
                            FLOAT: 12.300000
                      SEMI
                    DefList (8)
                      Def (8)
                        Specifier (8)
                          TYPE: float
                        DecList (8)
                          Dec (8)
                            VarDec (8)
                              ID: X_6
                            ASSIGNOP
                            Exp (8)
                              FLOAT: 10.000000
                        SEMI
        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

```
/* a comment */
// also a comment
// aaaa //
```

```
/*****
aasdfasdf
******
///\\\
*******/
int a, b;
/***
* main
* //ss
**/
int main() {
        // int a,b
        // /*\/
        int c;
        c = a;
        b = a + c + //aasdfdsb;
        /*asdfasff  */ b;
        if(c <= b) { //b > a comments*//\\//*
        //\\*//\\*//\\//*//\\*//\\//
                return c;
        } else

                return b;
}
```

输出

```
Program (9)
  ExtDefList (9)
    ExtDef (9)
      Specifier (9)
        TYPE: int
      ExtDecList (9)
        VarDec (9)
```

```
        ID: a
      COMMA
      ExtDecList (9)
        VarDec (9)
          ID: b
    SEMI
  ExtDefList (14)
    ExtDef (14)
      Specifier (14)
        TYPE: int
      FunDec (14)
        ID: main
        LP
        RP
      CompSt (14)
        LC
        DefList (17)
          Def (17)
            Specifier (17)
              TYPE: int
            DecList (17)
              Dec (17)
                VarDec (17)
                  ID: c
            SEMI
        StmtList (18)
          Stmt (18)
            Exp (18)
              Exp (18)
                ID: c
              ASSIGNOP
              Exp (18)
```

```
            ID: a
        SEMI
      StmtList (19)
        Stmt (19)
          Exp (19)
            Exp (19)
              ID: b
            ASSIGNOP
            Exp (19)
              Exp (19)
                Exp (19)
                  ID: a
                PLUS
                Exp (19)
                  ID: c
              PLUS
              Exp (20)
                ID: b
          SEMI
        StmtList (21)
          Stmt (21)
            IF
            LP
            Exp (21)
              Exp (21)
                ID: c
              RELOP
              Exp (21)
                ID: b
            RP
            Stmt (21)
              CompSt (21)
```

```
                        LC
                        StmtList (23)
                          Stmt (23)
                            RETURN
                            Exp (23)
                              ID: c
                            SEMI
                        RC
                    ELSE
                    Stmt (26)
                      RETURN
                      Exp (26)
                        ID: b
                      SEMI
            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）

```c
int test_for_wrong_oct_number()
{
    int _correct_oct_number_ = 003575;
    int _decimal_number = 1836;
    int _wrong_oct_number_ = 01836;
    int _correct_oct_number2_ = 00001266;
}
```

输出

```
Error type A at Line 5: Illegal octal number "01836"
```

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 01836，识别成错误类型 B 也可以。

输入（E1-2）

```
int test_for_wrong_dhex_number()
{
    int xF245 = 0;
    int _correct_dhex_number = 0xF369;
    int _wrong_dhex_number_ = 0xFG369;
    int _wrong_dhex_number2_ = 0xx245;
    int _correct_not_dhex_number3 = xF245;
}
```

输出

```
Error type A at Line 5: Illegal hexadecimal number "0xFG369"
Error type A at Line 6: Illegal hexadecimal number "0xx245"
```

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 0xFG369 与 0xx245，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）

```
float function()
{
    float x4 = 1.e1;
    float x1 = 4e;
    float x2 = 1.e1.3;

    return x1 + x2;
}
```

输出

```
Error type A at Line 4: Illegal float number "4e"
Error type A at Line 5: Illegal float number "1.e1.3"
```

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 4e 和 1.e1.3，识别成错误类型 B 也可以。

输入（E2-2）

```
float function()
{
    float a = 12.34e.-4;
    float b = .e;
}
```

输出

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

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.-4 和.e，识别成错误类型 B 和 A 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）

```
/*****
  //@Start
~^^^^@#$%@#%^
**/
/*
// @param int a
/*
   int i
*/
int test(int a){
```

```
  return a * /**?jj\\\\/ /////// */ 3;
}
/***@Override//*/
int method() {
  int i = 3;//\/*/\/\/\/\/\/\//// i = i +1;
  int sum // = 12;
  /*^_^*/ = 3;
  while(i <= /*test run!*/3) {
    sum /*blast */= sum + i;
    i = i * 2;
  /*   TODO

      */
  }

  test/*(*/(sum);  // should be executed

  return /*-1;*/sum/**i -1 +2;*/;
}
/* \***//*\*end of function/*\*/
```

输出

```
Program (10)
  ExtDefList (10)
    ExtDef (10)
      Specifier (10)
        TYPE: int
      FunDec (10)
        ID: test
        LP
        VarList (10)
          ParamDec (10)
            Specifier (10)
```

```
12            TYPE: int
13          VarDec (10)
14            ID: a
15      RP
16    CompSt (10)
17      LC
18      StmtList (11)
19        Stmt (11)
20          RETURN
21          Exp (11)
22            Exp (11)
23              ID: a
24            MUL
25            Exp (11)
26              INT: 3
27          SEMI
28      RC
29  ExtDefList (14)
30    ExtDef (14)
31      Specifier (14)
32        TYPE: int
33      FunDec (14)
34        ID: method
35        LP
36        RP
37      CompSt (14)
38        LC
39        DefList (15)
40          Def (15)
41            Specifier (15)
42              TYPE: int
43            DecList (15)
```

```
      Dec (15)
        VarDec (15)
          ID: i
        ASSIGNOP
        Exp (15)
          INT: 3
    SEMI
  DefList (16)
    Def (16)
      Specifier (16)
        TYPE: int
      DecList (16)
        Dec (16)
          VarDec (16)
            ID: sum
          ASSIGNOP
          Exp (17)
            INT: 3
      SEMI
StmtList (18)
  Stmt (18)
    WHILE
    LP
    Exp (18)
      Exp (18)
        ID: i
      RELOP
      Exp (18)
        INT: 3
    RP
    Stmt (18)
      CompSt (18)
```

```
        LC
        StmtList (19)
          Stmt (19)
            Exp (19)
              Exp (19)
                ID: sum
              ASSIGNOP
              Exp (19)
                Exp (19)
                  ID: sum
                ADD
                Exp (19)
                  ID: i
            SEMI
          StmtList (20)
            Stmt (20)
              Exp (20)
                Exp (20)
                  ID: i
                ASSIGNOP
                Exp (20)
                  Exp (20)
                    ID: i
                  MUL
                  Exp (20)
                    INT: 2
              SEMI
      RC
  StmtList (26)
    Stmt (26)
      Exp (26)
        ID: test
```

```
                    LP
                    Args (26)
                      Exp (26)
                        ID: sum
                    RP
                  SEMI
                StmtList (28)
                  Stmt (28)
                    RETURN
                    Exp (28)
                      ID: sum
                    SEMI
          RC
```

说明：仅 1.3 分组的同学需要测试这个用例，必须输出正确的语法树。

输入（E3-2）

```
/***comment
still
*/
int test() {
        int x;
        float p;
        if (x == 0){ //sadfasfd
*/
                return x;
        }
        else return x * 3;
//}
int main() {
        int x,y //;
        x = 3;
        while (x > /*4*/3) {
                y = y + /*\/\/\/\/\*\/\/\\****\/*/ 1;
```

```
18          }
19          while ( /*(*/x >1 )) {
20                  y = y - 2 * 3 / 4 + x;
21      x = x + 1;
22          }
23          return y;
24  }
25  /* end? ~^
```

输出

```
1  Error type B at 8: syntax error, near '*/'
2  Error type B at 13: syntax error, near 'int'
3  Error type B at 15: syntax error, near 'x'
4  Error type B at 19: syntax error, near '{'
5  Error type B at 26: syntax error, unexpected $end
```

说明：仅 1.3 分组的同学需要测试这个用例。第 8 行出现一个未匹配的 */；第 12 行的右大括号被注释掉，报在第 13 行也可以；第 14 行分号被注释掉，报在第 15 行也可以。第 19 行 while 语句中第二个左括号被注释掉。最后一个错误针对终止的注释进行测试，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 25 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。