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 4

1.8 A-8 5

1.9 A-9 6

2 B 组测试用例 6

2.1 B-1 6

2.2 B-2 7

3 C 组测试用例 9

3.1 C-1 9

3.2 C-2 12

4 D 组测试用例 18

4.1 D-1 18

4.2 D-2 20

4.3 D-3 23

5 E 组测试用例 25

5.1 E1.1 25

5.2 E1.2 26

5.3 E1.3 27

6 结束语 31

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()
{
    int _a_aaaaaaaaaaaaaaaaaaa, 4b;
}
```

输出

```
Error type B at line 3 : syntax error, near '4b'
```

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 3 行。这里有一个非法的标识符 4b。

1.2 A-2

输入

```
int sub()
{
    int a, b;
    a = a + 1;
    b = a & 2;
    return a - b;
}
```

输出

```
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 + 1));
5 }
```

输出

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

说明：第 4 行有未匹配的括号。

1.4 A-4

输入

```
1 int func()
2 {
3     int _a, b
4     b = 0;
5     _a = b + 1;
6 }
```

输出

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

说明：第 3 行结尾少了分号，因此错误位置报在第 4 行也算对。

1.5 A-5

输入

```
int arrayDef()
{
    int i, a[2][100];
    i = 0;
5      while (i > 100) {
6          a[0][i]= 1;
7          a[1,i] = i;
8          i = i + 1;
9      }
10 }
```

输出

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

说明：第 7 行错误的数组下标格式

1.6 A-6

输入

```
1  int func()
2  {
3      int i = 0, a[2][5.8][10];
4      while (i < 10) {
5          a[0][0][i] = i * i;
6          a[1][0][i] = i / i;
7      }
8  }
```

输出

```
Error type B at line 3 : syntax error, near '5.8'
```

说明：第 3 行数组声明的时候使用了浮点数 5.8

1.7 A-7

输入

```
1  struct Vector {
2      float x, y;
3      int a, b;
4       float c, d;
5   }
6
7   int sdef()
8   {
9       struct Vector v;
10      v.x = v.y = 1.0;
11      v.a = v.b = 1;
12      v.c = v.d = 3.5;
13  }
```

输出

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

说明：结构体定义时少了分号，因此错误位置在 5、6 行也算正确。

1.8 A-8

输入

```
1   struct Vector {
2       float x, y;
3   };
4
5   int sdef()
6   {
7       Vector v;
8       v.x = v.y = 1.0;
9   }
```

输出

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

说明：第 7 行定义结构体变量时少了 struct 关键字。

1.9 A-9

输入

```
1  int m1(int arg)
2  {
3      int a;
4      a = arg + 5;
5      return a;
6  }
7
8  int m2()
9  {
10     int b = 2, c;
11     c = m1((b);
12 }
```

输出

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

说明：第 11 行在函数调用时多了一个左括号。

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

```
struct Vector {
    float x y;
};

float squaring(struct Vector v0, struct Vector v1:)
{
```

```
7       struct Vector c0 = v0;
8       struct Vector c1 = v1;
9       return (c0.x - c1.x) * (c0.y - c1.y);
10  }
11
12  float main()
13  {
14      strut Vector v0, v1;
15      v0.x = 1.2;
16      v0.y = 3.6;
17      v1.x = 4.5;
18      v1.y = 6.7;
19      return squaring(v0_0, v0_1));
20  }
```

输出

```
Error type B at line 2 : syntax error, near 'y'
Error type A at line 5 : mysterious character ':'
Error type B at line 14 : syntax error, near 'strut'
Error type B at line 19 : syntax error, near ';'
```

说明：第 2 行缺少一个逗号；第 5 行多了一个分号；第 14 行 struct 关键字错写为 strut；第 19 行多了一个右括号。

2.2 B-2

输入

```
int func()
{
    int _arg1[5], _arg2[5];
    int i = 0, j = 0, tmp;
    while (i+1) < 6) {
        _arg1[i] = i + 2;
        i = ++;
8       }
9   
10      j = 5;
11      while (j > 0) {
12          _arg2[j] = _arg1[j] - j;
13              if (_arg1[j] > _arg2[j] +i) {
14                  tmp = _arg1[j];
15                  _arg1[j] = _arg2[j];
16                  _arg2[j] = tmp;
17              }
18          j--;
19      }
20  
21      i = 0;
22      while (i < 5) {
23          printInt(_arg1[i]);
24          printInt(_arg2[i]);
25          i = i + 1;
26      }
27      return _arg1[0] + _arg2[0]
28  }
```

输出

```
Error type B at line 5 : syntax error, near '('
Error type B at line 7 : syntax error, near '+'
Error type B at line 18 : syntax error, near '-'
Error type B at line 28 : syntax error, near '}'
```

说明：第5行少了一个左括号符号；第7行使用了错误的++操作符；第18行使用了错误的–操作符；第28行上一句return语句少了分号。

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

```c
struct Vector {
    int a;
    int b;
};

int test()
{
    struct Vector v1;
    v1.a = -1;
    v1.b = v1.a + 2;
    return v1.a - v1.b;
}
```

输出

```
Program (1)
  ExtDefList (1)
    ExtDef (1)
      Specifier (1)
        StructSpecifier (1)
          STRUCT
          OptTag (1)
            ID: Vector
          LC
          DefList (2)
            Def (2)
              Specifier (2)
              TYPE: int
            DecList (2)
              Dec (2)
                VarDec (2)
                  ID: a
            SEMI
          DefList (3)
            Def (3)
              Specifier (3)
                TYPE: int
              DecList (3)
                Dec (3)
                  VarDec (3)
                    ID: b
              SEMI
        RC
    SEMI
  ExtDefList (6)
    ExtDef (6)
      Specifier (6)
        TYPE: int
      FunDec (6)
        ID: test
        LP
        RP
      CompSt (7)
        LC
        DefList (8)
          Def (8)
            Specifier (8)
              StructSpecifier (8)
                STRUCT
```

```
                  Tag (8)
                    ID: Vector
              DecList (8)
                Dec (8)
                  VarDec (8)
                    ID: v1
              SEMI
          StmtList (9)
            Stmt (9)
              Exp (9)
                Exp (9)
                  Exp (9)
                    ID: v1
                  DOT
                  ID: a
                ASSIGNOP
                Exp (9)
                  MINUS
                  Exp (9)
                    INT: 1
              SEMI
            StmtList (10)
              Stmt (10)
                Exp (10)
                  Exp (10)
                    Exp (10)
                      ID: v1
                    DOT
                    ID: b
                  ASSIGNOP
                  Exp (10)
                    Exp (10)
```

```
                      Exp (10)
                        ID: v1
                      DOT
                      ID: a
                    PLUS
                    Exp (10)
                      INT: 2
                SEMI
              StmtList (11)
                Stmt (11)
                  RETURN
                  Exp (11)
                    Exp (11)
                      Exp (11)
                        ID: v1
                      DOT
                      ID: a
                    MINUS
                    Exp (11)
                      Exp (11)
                        ID: v1
                      DOT
                      ID: b
                  SEMI
          RC
```

说明：使用的空格可以换位 Tab。

3.2 C-2

输入

```
int test()
{
    int a[4][5];
    int i = 0;
    a[i][i] = 4;
    while (i < 4) {
        a[i][0] = i;
        a[1][i] = i + 1;
        i = i + 1;
    }
    return a[0][0];
}
```

输出

```
Program (1)
  ExtDefList (1)
    ExtDef (1)
      Specifier (1)
        TYPE: int
      FunDec (1)
        ID: test
        LP
        RP
      CompSt (2)
        LC
        DefList (3)
          Def (3)
            Specifier (3)
              TYPE: int
            DecList (3)
              Dec (3)
                VarDec (3)
                  VarDec (3)
                    VarDec (3)
                      ID: a
                    LB
                    INT: 4
                    RB
                  LB
                  INT: 5
                  RB
            SEMI
          DefList (4)
            Def (4)
              Specifier (4)
                TYPE: int
              DecList (4)
                Dec (4)
                  VarDec (4)
                    ID: i
                  ASSIGNOP
                  Exp (4)
                    INT: 0
              SEMI
        StmtList (5)
          Stmt (5)
            Exp (5)
              Exp (5)
                Exp (5)
                  Exp (5)
                    ID: a
                  LB
                  Exp (5)
                    ID: i
                  RB
                LB
                Exp (5)
                  ID: i
                RB
              ASSIGNOP
              Exp (5)
                INT: 4
            SEMI
          StmtList (6)
            Stmt (6)
              WHILE
              LP
              Exp (6)
                Exp (6)
                  ID: i
                RELOP
                Exp (6)
                  INT: 4
              RP
              Stmt (6)
                CompSt (6)
                  LC
                  StmtList (7)
                    Stmt (7)
                      Exp (7)
                        Exp (7)
                          Exp (7)
                            Exp (7)
                              ID: a
                            LB
                            Exp (7)
                              ID: i
                            RB
                          LB
                          Exp (7)
```

```
        INT: 0
      RB
    ASSIGNOP
    Exp (7)
      ID: i
  SEMI
StmtList (8)
  Stmt (8)
    Exp (8)
      Exp (8)
        Exp (8)
          Exp (8)
            ID: a
          LB
          Exp (8)
            INT: 1
          RB
        LB
        Exp (8)
          ID: i
        RB
      ASSIGNOP
      Exp (8)
        Exp (8)
          ID: i
        PLUS
        Exp (8)
          INT: 1
    SEMI
  StmtList (9)
    Stmt (9)
      Exp (9)
```

```
                            Exp (9)
                              ID: i
                            ASSIGNOP
                            Exp (9)
                              Exp (9)
                                ID: i
                              PLUS
                              Exp (9)
                                INT: 1
                          SEMI
                  RC
            StmtList (11)
              Stmt (11)
                RETURN
                Exp (11)
                  Exp (11)
                    Exp (11)
                      ID: a
                    LB
                    Exp (11)
                      INT: 0
                    RB
                  LB
                  Exp (11)
                    INT: 0
                  RB
                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

```
int func()
{
    int oct = 0125;
    int dec = 239;
    int hex1 = 0x0B0c;
    int dhex2 = 0XD01;
}
```

输出

```
Program (1)
  ExtDefList (1)
    ExtDef (1)
      Specifier (1)
        TYPE: int
      FunDec (1)
        ID: func
        LP
        RP
      CompSt (2)
        LC
        DefList (3)
          Def (3)
            Specifier (3)
              TYPE: int
            DecList (3)
```

```
              Dec (3)
                VarDec (3)
                  ID: oct
                ASSIGNOP
                Exp (3)
                  INT: 85
            SEMI
          DefList (4)
            Def (4)
              Specifier (4)
                TYPE: int
              DecList (4)
                Dec (4)
                  VarDec (4)
                    ID: dec
                  ASSIGNOP
                  Exp (4)
                    INT: 239
              SEMI
            DefList (5)
              Def (5)
                Specifier (5)
                  TYPE: int
                DecList (5)
                  Dec (5)
                    VarDec (5)
                      ID: hex1
                    ASSIGNOP
                    Exp (5)
                      INT: 2828
                SEMI
              DefList (6)
```

```
                Def (6)
                  Specifier (6)
                    TYPE: int
                  DecList (6)
                    Dec (6)
                      VarDec (6)
                        ID: dhex2
                      ASSIGNOP
                      Exp (6)
                        INT: 3329
                  SEMI
        RC
```

说明：1.1 分组（任务 ID 1,4,7,10,13,16）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（85、239、2828 和 3329）；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

```
int main()
{
    float X_1 = 2.42e-3;
    float X_2 = 1.12E2;
    float X_3 = 66.2e+2;
    float X_4 = 1.62E+2;
    float X_5 = .221E-2;
    float X_6 = 3.24e1;
}
```

输出

```
Program (1)
  ExtDefList (1)
    ExtDef (1)
      Specifier (1)
      TYPE: int
    FunDec (1)
      ID: main
      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: 0.002420
          SEMI
        DefList (4)
          Def (4)
            Specifier (4)
              TYPE: float
            DecList (4)
              Dec (4)
                VarDec (4)
                  ID: X_2
                ASSIGNOP
                Exp (4)
                  FLOAT: 112.000000
            SEMI
          DefList (5)
```

```
              Def (5)
                Specifier (5)
                  TYPE: float
                DecList (5)
                  Dec (5)
                    VarDec (5)
                      ID: X_3
                    ASSIGNOP
                    Exp (5)
                      FLOAT: 6620.000000
                SEMI
              DefList (6)
                Def (6)
                  Specifier (6)
                    TYPE: float
                  DecList (6)
                    Dec (6)
                      VarDec (6)
                        ID: X_4
                      ASSIGNOP
                      Exp (6)
                        FLOAT: 162.000000
                  SEMI
                DefList (7)
                  Def (7)
                    Specifier (7)
                      TYPE: float
                    DecList (7)
                      Dec (7)
                        VarDec (7)
                          ID: X_5
                        ASSIGNOP
                        Exp (7)
                          FLOAT: 0.002210
                    SEMI
                  DefList (8)
                    Def (8)
                      Specifier (8)
                        TYPE: float
                      DecList (8)
                        Dec (8)
                          VarDec (8)
                            ID: X_6
                          ASSIGNOP
                          Exp (8)
                            FLOAT: 32.400000
                      SEMI
        RC
```

说明：1.2 分组（任务 ID 2,5,8,11,14,17）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

```
int test()
{ // a line of comments
    float a = 4.5;
    /*
    float a = 0.1;
    /* */
//} // float b = 4;
    float b = 3;
    /* float b = 0.4;/*****
    int b = 2
    */
    return a + b;
}
```

输出

```
Program (1)
  ExtDefList (1)
    ExtDef (1)
      Specifier (1)
        TYPE: int
      FunDec (1)
        ID: test
        LP
        RP
      CompSt (2)
        LC
        DefList (3)
          Def (3)
            Specifier (3)
              TYPE: float
            DecList (3)
              Dec (3)
                VarDec (3)
                  ID: a
                ASSIGNOP
                Exp (3)
                  FLOAT: 4.500000
            SEMI
          DefList (5)
            Def (5)
              Specifier (5)
                TYPE: float
              DecList (5)
                Dec (5)
```

```
                  VarDec (5)
                    ID: b
                  ASSIGNOP
                  Exp (5)
                    INT: 3
              SEMI
        StmtList (7)
          Stmt (7)
            RETURN
            Exp (7)
              Exp (7)
                ID: a
              PLUS
              Exp (7)
                ID: b
            SEMI
        RC
```

说明：1.3 分组（任务 ID 3,6,9,12,15,18）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）

```
int func()
{
    int a = 0X1ha;
}
```

输出

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

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 0X1ha。

输入（E1-2）

```
int test()
{
    int _C_ = 0109;
}
```

输出

```
Error type B at line 3 : syntax error, near '09'
```

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 0109。

5.2 E1.2

输入（E2-1）

```
float main()
{
    float C_C = 4.23e.-1;
    float A_A = 1.69e-1;
    return C_C + A_A;
}
```

输出

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

说明：仅 1.2 分组同学需要测试这组用例，错误在于指数为 e.-1

输入（E2-2）

```
float func()
{
    float a = .E-1;
}
```

输出

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

说明：仅 1.2 分组同学需要测试这组用例，错误在于底数只有小数点

5.3 E1.3

输入（E3-1）

```
// \\ // \\ // \\ // \\ \\
// \\ // \\ // \\ // \\ //
/*
* I am a line of comments, but plz do not ignore me ~_~
*/

float /* ^_^ * >_< * _/\_ */test()
{
    /*
    /~/~/~~~~~~~~~~~~~ */
    float a, b, c;
    // /*/
    a = 1.5;
    // a = b + 1;
    /* *******\/\/
    // a line of comments
    float a = 5;
    */
    b = 1.5;
    c = (a + b) + (a + b);
} //a line of comments
/*
I am a line of comments
*/
```

输出

```
Program (1)
  ExtDefList (1)
    ExtDef (1)
      Specifier (1)
        TYPE: float
      FunDec (1)
        ID: test
        LP
        RP
      CompSt (2)
        LC
        DefList (3)
          Def (3)
            Specifier (3)
              TYPE: float
            DecList (3)
              Dec (3)
                VarDec (3)
                  ID: a
              COMMA
              DecList (3)
                Dec (3)
                  VarDec (3)
                    ID: b
                COMMA
                DecList (3)
                  Dec (3)
                    VarDec (3)
                      ID: c
            SEMI
        StmtList (5)
          Stmt (5)
```

```
            Exp (5)
              Exp (5)
                ID: a
              ASSIGNOP
              Exp (5)
                FLOAT: 1.500000
            SEMI
          StmtList (7)
            Stmt (7)
              Exp (7)
                Exp (7)
                  ID: b
                ASSIGNOP
                Exp (7)
                  FLOAT: 1.500000
              SEMI
            StmtList (8)
              Stmt (8)
                Exp (8)
                  Exp (8)
                    ID: c
                  ASSIGNOP
                  Exp (8)
                    Exp (8)
                      LP
                      Exp (8)
                        Exp (8)
                          ID: a
                        PLUS
                        Exp (8)
                          ID: b
                      RP
                    PLUS
                    Exp (8)
                      LP
                      Exp (8)
                        Exp (8)
                          ID: a
                        PLUS
                        Exp (8)
                          ID: b
                      RP
                SEMI
        RC
```

说明：必须输出正确的语法树，否则该用例不得分

输入（E3-2）

```
/*
* @a line of comments
*/

// a line of comments //

float /*~~~*/ test()//test(int a)
{
    /*
    float a, b, c, d; /* /float a, b;
    c =  2 * a  2 * a - 3 * b; /3 * b;
    /\/ deleting ... 1, 2, 3 \/\
    d = a + b  a - b;
} */
    float a, b, c, d; /* /float a, b; */
    c = /* 2 * a */ 2 * a - 3 * b; //3 * b;
    // deleting ... 1, 2, 3 \\
    d = /* a + b */ a - b; /* ; // a / b /
```

```
19 }//a line of comments
```

输出

```
提示语法错误即可
```

说明：针对未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 19 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，可以写邮件与杨文华助教联系，注意同时抄送给许老师。