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 10 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 main () {
  int i = 1.100;
  float f = 1;
  int j = i * f + 1.00.1;
}
```

输出

```
Error type A at line 4: Illegal number '1.00.1'
```

说明：也可以报成 B 类错误。

1.2　A-2

输入

```
int f0o() {
  return 0;
}

int main() {
  int 1i = f0o();
}
```

输出

```
Error type A at line 6: Illegal identifier '1i'
```

说明：也可以报成 B 类错误。

1.3 A-3

输入

```
struct {
  int x;
  int y;
  int z;
} s1, s2;

int foo() {
  struct s int;
}
```

输出

```
Error type B at line 8: Expect identifier;
```

说明：关键字不能作为标识符。

1.4 A-4

输入

```
int i, j, k;
int a[100];

float f = 1.00;

int hello() {
  int a[100];
}
```

输出

```
Error type B at line 4: Global variable cannot have initializer;
```

说明：全局变量定义时不能初始化。

1.5 A-5

输入

```
int foo() {
  int i, j;
  i = 0;
  j = (i * 5 + 42) / 23;
  float f;
  return i;
}
```

输出

```
Error type B at line 5: Unexpected variable declaration
```

说明：变量声明的部分必须在语句块的开始部分。

1.6 A-6

输入

```
struct Car {
  int color;
  float weight;
};

struct Fee {
  Car car;
  int count;
};

int main() {
  struct Car car;
  car.color = 0;
  car.weight = 1.111;
}
```

输出

```
Error type B at line 7: Illegal type specifier
```

说明：缺少 struct 关键字。

1.7 A-7

输入

```
int a1[100];
int a2[100];

int init(int a[100]) {
  int i;
  while (i < 100) {
    a[i] = i;
    i = i + 1;
  }
}

int add(int a[100], int b[100]) {
  int i = 0;
  int res[100];
  while (i < 100) {
    res[i] = a[i] + b[i];
    i += 1;
  }
}

int main() {
  init(a1);
  init(a2);
  add(a1, a2);
}
```

输出

```
Error type B at line 17: '+=' is not supported
```

说明：C--没有 “+=” 运算符。

1.8 A-8

输入

```
struct Oops say() {
  struct Point {
    int x;
    int y;
    int z;
  } p1, p2;
  return 0;
}

int;
int;

struct st {
  int s1;
  float s2;
  struct st s3, s4;
  int arr[10.0];
};
```

输出

```
Error type B at line 17: array length can only be integer
```

说明：数组的长度必须为整数类型。

1.9 A-9

输入

```
int turn;
int flag[2];

int doSth() {
  int i = 0;
  while (i < 42) {
    i = i + 1;
  }
}

int t1() {
  turn = 1;
  while (flag[1] == 1 && turn == 1) {

  }
  doSth();
  flag[0] = 0;
}

int t2() {
  turn = 0;
  while (flag[0] == 1 && turn == 0);
  doSth();
  flag[1] = 0;
}
```

输出

```
Error type B at line 22: Expect statement after while condition
```

说明：C–中没有空语句。

1.10 A-10

输入

```c
int a[10][10];
int b[10][10];

struct container {
  int res;
  int cnt;
  float avg;
  int sum[10];
};

int comp(int a[10][10], int b[10][10]) {
  if (a[5][3] > a[2][1]) {
    return a[0][0];
  } else if (a[1][2] > b[0][1]) {
    return b[1][1];
  } else {
    return b[4][4];
  }
}

int cal() {
  int i = 0;
  int j = 0;
  int res;
  struct container ct;
  while (i < 10) {
    while (j < 10) {
      res = res + a[i][j] * b[i][j];
      j = j + 1;
    }
    i = i + 1;
  }
33    ct.res = res;
34    ct.cnt = i * j;
35    ct.avg = ct.res / ct.cnt;
36    i = 0;
37    j = 0;
38    while (i < 10) {
39      ct.sum[i] = 0;
40      while (j < 10) {
41        ct.sum[i] = ct.sum[i,] + a[i][j] + b[i][j];
42        j = j + 1;
43      }
44      i = i + 1;
45    }
46  }
```

输出

```
Error type B at line 41: Unexpected ','
```

说明：数组索引中只能出现一个整数。

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

```
int _a0_b1(int arr[10], float x, int y) {
 int i = 0, j = -1; int k = 1--2*3;
 struct ss {
  float ff;
  int _i, _j, _k;
  struct {
      int a, b;
    } s;
  } s1, s2;

  while (i < 10) {
    if (a > b) {
      return 1 + 2 / (a - b) + 3 * 3.14;
    } else {
      int tmp = a;
      a = a + b;
      b = b - tmp;
      return a * b * tmp;
    }
    ;
    i = i + 1;
  }
}

int _F_o__0(struct { int i; int j; } ss, float f, int i) {
  int a[100];
  float b[100];
  ss.i = 0h;
  ss.j = 100;
  while (true) {
    a[ss.i] = b[ss.i];
    ss.i = ss.i + 1;
    if (ss.i >= 100) {
      return a[ss.i - 1,];
    }
  }
  ss.j = ss.i + ss.j;
}
```

输出

```
Error type B at line 20: Unexpected ';'
Error type A at line 28: illegal identifier '0h';
Error type B at line 34: Unexpected ','
```

说明：20 行多了一个 “;”；28 行的 “0h” 不是合法的整数常量，这里也可以报成 B 类错误；34 行的数组索引中多了一个 “,”。

2.2 B-2

输入

```
struct Info {
  int amount;
  int level;
  int fee;
};

struct Info calculate(struct Info input, int a, int l) {
  struct Info res;
  if (a > 0) {
    res.amount = a;
  } else {
    res.amount = input.amount;
  }
  if (l > 0) {
    res.level = l;
  } else {
    res.level = input.level;
  }

  if (res.level = 0) {
    res.fee = ----430.43;
  } else if (res.level = = 1) {
```

```
23     res.fee = (res.amount - 43.33) * res.level + 43.33 * (res.level +
           1);
24   } else if (res.level == 2) {
25     res.fee = res.amount / 3 + res.amount * (res.level + 100);
26   } else
27     res.fee = 100 * ((100 / res.amount) - res.level);
28
29   return res + res * a-;
30 }
31
32 int main() {
33   int m, n, k = 1 + 2 * 4;
34   struct ss {
35     struct { };
36     int i, j = 0;
37   } s;
38 }
```

输出

```
Error type B at line 22: Unexpected '=' after '='
Error type B at line 29: Expect expression after '-'
Error type B at line 35: Expect identifier after '}'
```

说明：22 行多了一个空格；29 行 “-” 后面少了一个表达式；35 行少了变量名，结构体中可以再定义结构体类型的变量，而不能只声明结构体类型。

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

```
int main() {
  bin2dec(1001001);
  dec2bin(1234567);
}

int rem(int a, int b) {
  return a - a / b * b;
}

int bin2dec(int n) {
  int dec = 0, i = 0, rem;
  while (n != 0) {
    rem = rem(n, 10);
    n = n / 10;
    dec = dec + rem * pow(2, i);
    i = i + 1;
  }
  return dec;
}

int dec2bin(int n) {
  if (n <= 1)
    return n;
  else {
    dec2bin(n / 2);
    write(rem(n, 2));
  }
}

int cal() {
  int z_x = 0;
  while (1 * 2 + (2 - 3 * 4 )) {
    if (a > b)
      dec2bin(a);
    else {
      int z_x = x / (y * y - z);
      bin2dec(b);
    }
  }
}
```

输出

```
Program (1)
ExtDefList (1)
  ExtDef (1)
    Specifier (1)
      TYPE: int
    FunDec (1)
      ID: main
      LP
      RP
    CompSt (1)
      LC
      StmtList (2)
        Stmt (2)
          Exp (2)
            ID: bin2dec
            LP
            Args (2)
              Exp (2)
                INT: 1001001
            RP
          SEMI
        StmtList (3)
          Stmt (3)
            Exp (3)
              ID: dec2bin
              LP
              Args (3)
                Exp (3)
                  INT: 1234567
              RP
            SEMI
      RC
  ExtDefList (6)
    ExtDef (6)
      Specifier (6)
        TYPE: int
      FunDec (6)
        ID: rem
        LP
        VarList (6)
          ParamDec (6)
            Specifier (6)
              TYPE: int
            VarDec (6)
              ID: a
          COMMA
          VarList (6)
            ParamDec (6)
              Specifier (6)
                TYPE: int
              VarDec (6)
                ID: b
        RP
      CompSt (6)
        LC
        StmtList (7)
          Stmt (7)
            RETURN
            Exp (7)
              Exp (7)
                ID: a
              MINUS
              Exp (7)
                Exp (7)
                  Exp (7)
                    ID: a
                  DIV
                  Exp (7)
                    ID: b
                STAR
                Exp (7)
                  ID: b
            SEMI
        RC
    ExtDefList (10)
      ExtDef (10)
        Specifier (10)
          TYPE: int
        FunDec (10)
          ID: bin2dec
          LP
          VarList (10)
            ParamDec (10)
              Specifier (10)
                TYPE: int
              VarDec (10)
                ID: n
          RP
        CompSt (10)
          LC
          DefList (11)
            Def (11)
              Specifier (11)
                TYPE: int
              DecList (11)
                Dec (11)
                  VarDec (11)
                    ID: dec
                  ASSIGNOP
                  Exp (11)
                    INT: 0
                COMMA
                DecList (11)
                  Dec (11)
                    VarDec (11)
                      ID: i
                    ASSIGNOP
                    Exp (11)
                      INT: 0
                  COMMA
                  DecList (11)
                    Dec (11)
                      VarDec (11)
                        ID: rem
              SEMI
          StmtList (12)
            Stmt (12)
              WHILE
              LP
              Exp (12)
                Exp (12)
                  ID: n
                RELOP
                Exp (12)
                  INT: 0
              RP
              Stmt (12)
                CompSt (12)
                  LC
                  StmtList (13)
                    Stmt (13)
                      Exp (13)
                        Exp (13)
                          ID: rem
                        ASSIGNOP
                        Exp (13)
                          ID: rem
                          LP
                          Args (13)
                            Exp (13)
                              ID: n
                            COMMA
                            Args (13)
                              Exp (13)
                                INT: 10
                          RP
                      SEMI
                    StmtList (14)
                      Stmt (14)
                        Exp (14)
                          Exp (14)
```

```
      ID: n
    ASSIGNOP
    Exp (14)
      Exp (14)
        ID: n
      DIV
      Exp (14)
        INT: 10
  SEMI
StmtList (15)
  Stmt (15)
    Exp (15)
      Exp (15)
        ID: dec
      ASSIGNOP
      Exp (15)
        Exp (15)
          ID: dec
        PLUS
        Exp (15)
          Exp (15)
            ID: rem
          STAR
          Exp (15)
            ID: pow
            LP
            Args (15)
              Exp (15)
                INT: 2
              COMMA
              Args (15)
                Exp (15)
```

```
                                    ID: i
                              RP
                      SEMI
                    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)
              ID: dec
            SEMI
      RC
  ExtDefList (21)
    ExtDef (21)
      Specifier (21)
        TYPE: int
      FunDec (21)
        ID: dec2bin
        LP
        VarList (21)
```

```
              ParamDec (21)
                Specifier (21)
                  TYPE: int
                VarDec (21)
                  ID: n
            RP
          CompSt (21)
            LC
            StmtList (22)
              Stmt (22)
                IF
                LP
                Exp (22)
                  Exp (22)
                    ID: n
                  RELOP
                  Exp (22)
                    INT: 1
                RP
                Stmt (23)
                  RETURN
                  Exp (23)
                    ID: n
                  SEMI
                ELSE
                Stmt (24)
                  CompSt (24)
                    LC
                    StmtList (25)
                      Stmt (25)
                        Exp (25)
                          ID: dec2bin
                          LP
                          Args (25)
                            Exp (25)
                              Exp (25)
                                ID: n
                              DIV
                              Exp (25)
                                INT: 2
                          RP
                        SEMI
                      StmtList (26)
                        Stmt (26)
                          Exp (26)
                            ID: write
                            LP
                            Args (26)
                              Exp (26)
                                ID: rem
                                LP
                                Args (26)
                                  Exp (26)
                                    ID: n
                                  COMMA
                                  Args (26)
                                    Exp (26)
                                      INT: 2
                                RP
                            RP
                          SEMI
                    RC
            RC
        ExtDefList (30)
```

```
        ExtDef (30)
          Specifier (30)
            TYPE: int
          FunDec (30)
            ID: cal
            LP
            RP
          CompSt (30)
            LC
            DefList (31)
              Def (31)
                Specifier (31)
                  TYPE: int
                DecList (31)
                  Dec (31)
                    VarDec (31)
                      ID: z_x
                    ASSIGNOP
                    Exp (31)
                      INT: 0
                SEMI
            StmtList (32)
              Stmt (32)
                WHILE
                LP
                Exp (32)
                  Exp (32)
                    Exp (32)
                      INT: 1
                    STAR
                    Exp (32)
                      INT: 2
                  PLUS
                  Exp (32)
                    LP
                    Exp (32)
                      Exp (32)
                        INT: 2
                      MINUS
                      Exp (32)
                        Exp (32)
                          INT: 3
                        STAR
                        Exp (32)
                          INT: 4
                    RP
                RP
                Stmt (32)
                  CompSt (32)
                    LC
                    StmtList (33)
                      Stmt (33)
                        IF
                        LP
                        Exp (33)
                          Exp (33)
                            ID: a
                          RELOP
                          Exp (33)
                            ID: b
                        RP
                        Stmt (34)
                          Exp (34)
                            ID: dec2bin
```

```
    LP
    Args (34)
      Exp (34)
        ID: a
    RP
  SEMI
ELSE
Stmt (35)
  CompSt (35)
    LC
    DefList (36)
      Def (36)
        Specifier (36)
          TYPE: int
        DecList (36)
          Dec (36)
            VarDec (36)
              ID: z_x
            ASSIGNOP
            Exp (36)
              Exp (36)
                ID: x
              DIV
              Exp (36)
                LP
                Exp (36)
                  Exp (36)
                    Exp (36)
                      ID: y
                    STAR
                    Exp (36)
                      ID: y
```

```
                                            MINUS
                                            Exp (36)
                                              ID: z
                                          RP
                                  SEMI
                              StmtList (37)
                                Stmt (37)
                                  Exp (37)
                                    ID: bin2dec
                                    LP
                                    Args (37)
                                      Exp (37)
                                        ID: b
                                    RP
                                  SEMI
                              RC
                      RC
              RC
```

说明：使用的空格可以用 Tab 替换，注意缩进

3.2 C-2

输入

```c
struct Data {
  struct a {
    int x, y, z = 1;
  } s1;
  struct b s2;
  float f_1, f__2 = 2.2, _f3;
  int a[10][11];
} xs;

struct Data comp() {
  xs.s1.x = xs.s1.y * xs.s1.z + xs.s1.z;
  xs.s2 = xs.s2;
  xs.f_1 = xs.f__2;
  if (xs.f_1 > 0.01) {
    xs._f3 = xs._f3 * xs.f_1 / xs.f__2;
    xs.f__2 = xs.f__2 + 1;
  } else
    xs.f_1 = xs.f__2;

  while (xs.s1.x > xs.s1.y) {
    int i = 0;
    while (i > xs.s1.z) {
      xs._f3 = 0.1 * xs._f3;
      xs.s1.z = xs.s1.z * 2;
      if (xs.a[i][i]) {
        return xs;
      }
      xs.s1.x = xs.s1.y / i;
    }
  }
}

int main() {
  int _i = 0, _j = 0;
  comp();
  while (i < 10) {
    while (j < 10) {
      xs.a[i][j] = i * j + xs.s1.x;
    }
    xs.s1.y = xs.a[i][j];
  }
  return xs.a[2][3];
```

```
}
```

输出

```
Program (1)
ExtDefList (1)
  ExtDef (1)
    Specifier (1)
      StructSpecifier (1)
        STRUCT
        OptTag (1)
          ID: Data
        LC
        DefList (2)
          Def (2)
            Specifier (2)
              StructSpecifier (2)
                STRUCT
                OptTag (2)
                  ID: a
                LC
                DefList (3)
                  Def (3)
                    Specifier (3)
                      TYPE: int
                    DecList (3)
                      Dec (3)
                        VarDec (3)
                          ID: x
                      COMMA
                      DecList (3)
                        Dec (3)
                          VarDec (3)
                            ID: y
                        COMMA
                        DecList (3)
                          Dec (3)
                            VarDec (3)
                              ID: z
                            ASSIGNOP
                            Exp (3)
                              INT: 1
                    SEMI
                RC
            DecList (4)
              Dec (4)
                VarDec (4)
                  ID: s1
            SEMI
          DefList (5)
            Def (5)
              Specifier (5)
                StructSpecifier (5)
                  STRUCT
                  Tag (5)
                    ID: b
              DecList (5)
                Dec (5)
                  VarDec (5)
                    ID: s2
              SEMI
            DefList (6)
              Def (6)
                Specifier (6)
                  TYPE: float
                DecList (6)
```

```
                  Dec (6)
                    VarDec (6)
                      ID: f_1
                  COMMA
                  DecList (6)
                    Dec (6)
                      VarDec (6)
                        ID: f__2
                      ASSIGNOP
                      Exp (6)
                        FLOAT: 2.200000
                    COMMA
                    DecList (6)
                      Dec (6)
                        VarDec (6)
                          ID: _f3
                SEMI
              DefList (7)
                Def (7)
                  Specifier (7)
                    TYPE: int
                  DecList (7)
                    Dec (7)
                      VarDec (7)
                        VarDec (7)
                          VarDec (7)
                            ID: a
                          LB
                          INT: 10
                          RB
                        LB
                        INT: 11
                        RB
                  SEMI
        RC
    ExtDecList (8)
      VarDec (8)
        ID: xs
    SEMI
  ExtDefList (10)
    ExtDef (10)
      Specifier (10)
        StructSpecifier (10)
          STRUCT
          Tag (10)
            ID: Data
      FunDec (10)
        ID: comp
        LP
        RP
      CompSt (10)
        LC
        StmtList (11)
          Stmt (11)
            Exp (11)
              Exp (11)
                Exp (11)
                  Exp (11)
                    ID: xs
                  DOT
                  ID: s1
                DOT
                ID: x
              ASSIGNOP
              Exp (11)
                Exp (11)
                  Exp (11)
                    Exp (11)
                      Exp (11)
                        ID: xs
                      DOT
                      ID: s1
                    DOT
                    ID: y
                  STAR
                  Exp (11)
                    Exp (11)
                      Exp (11)
                        ID: xs
                      DOT
                      ID: s1
                    DOT
                    ID: z
                PLUS
                Exp (11)
                  Exp (11)
                    Exp (11)
                      ID: xs
                    DOT
                    ID: s1
                  DOT
                  ID: z
            SEMI
          StmtList (12)
            Stmt (12)
              Exp (12)
```

```
                Exp (12)
                  Exp (12)
                    ID: xs
                  DOT
                  ID: s2
                ASSIGNOP
                Exp (12)
                  Exp (12)
                    ID: xs
                  DOT
                  ID: s2
              SEMI
            StmtList (13)
              Stmt (13)
                Exp (13)
                  Exp (13)
                    Exp (13)
                      ID: xs
                    DOT
                    ID: f_1
                  ASSIGNOP
                  Exp (13)
                    Exp (13)
                      ID: xs
                    DOT
                    ID: f__2
                SEMI
              StmtList (14)
                Stmt (14)
                  IF
                  LP
                  Exp (14)
```

```
  Exp (14)
    Exp (14)
      ID: xs
    DOT
    ID: f_1
  RELOP
  Exp (14)
    FLOAT: 0.010000
RP
Stmt (14)
  CompSt (14)
    LC
    StmtList (15)
      Stmt (15)
        Exp (15)
          Exp (15)
            Exp (15)
              ID: xs
            DOT
            ID: _f3
          ASSIGNOP
          Exp (15)
            Exp (15)
              Exp (15)
                Exp (15)
                  ID: xs
                DOT
                ID: _f3
              STAR
              Exp (15)
                Exp (15)
                  ID: xs
```

```
                                  DOT
                                  ID: f_1
                              DIV
                              Exp (15)
                                Exp (15)
                                  ID: xs
                                DOT
                                ID: f__2
                          SEMI
                        StmtList (16)
                          Stmt (16)
                            Exp (16)
                              Exp (16)
                                Exp (16)
                                  ID: xs
                                DOT
                                ID: f__2
                              ASSIGNOP
                              Exp (16)
                                Exp (16)
                                  Exp (16)
                                    ID: xs
                                  DOT
                                  ID: f__2
                                PLUS
                                Exp (16)
                                  INT: 1
                            SEMI
                      RC
                  ELSE
                  Stmt (18)
                    Exp (18)
```

```
                      Exp (18)
                        Exp (18)
                          ID: xs
                        DOT
                        ID: f_1
                      ASSIGNOP
                      Exp (18)
                        Exp (18)
                          ID: xs
                        DOT
                        ID: f__2
                    SEMI
                StmtList (20)
                  Stmt (20)
                    WHILE
                    LP
                    Exp (20)
                      Exp (20)
                        Exp (20)
                          Exp (20)
                            ID: xs
                          DOT
                          ID: s1
                        DOT
                        ID: x
                      RELOP
                      Exp (20)
                        Exp (20)
                          Exp (20)
                            ID: xs
                          DOT
                          ID: s1
```

```
    DOT
    ID: y
RP
Stmt (20)
  CompSt (20)
    LC
    DefList (21)
      Def (21)
        Specifier (21)
          TYPE: int
        DecList (21)
          Dec (21)
            VarDec (21)
              ID: i
            ASSIGNOP
            Exp (21)
              INT: 0
        SEMI
    StmtList (22)
      Stmt (22)
        WHILE
        LP
        Exp (22)
          Exp (22)
            ID: i
          RELOP
          Exp (22)
            Exp (22)
              Exp (22)
                ID: xs
              DOT
              ID: s1
            DOT
            ID: z
        RP
        Stmt (22)
          CompSt (22)
            LC
            StmtList (23)
              Stmt (23)
                Exp (23)
                  Exp (23)
                    Exp (23)
                      ID: xs
                    DOT
                    ID: _f3
                  ASSIGNOP
                  Exp (23)
                    Exp (23)
                      FLOAT: 0.100000
                    STAR
                    Exp (23)
                      Exp (23)
                        ID: xs
                      DOT
                      ID: _f3
                SEMI
              StmtList (24)
                Stmt (24)
                  Exp (24)
                    Exp (24)
                      Exp (24)
                        Exp (24)
                          ID: xs
                        DOT
                        ID: s1
                      DOT
                      ID: z
                    ASSIGNOP
                    Exp (24)
                      Exp (24)
                        Exp (24)
                          Exp (24)
                            ID: xs
                          DOT
                          ID: s1
                        DOT
                        ID: z
                      STAR
                      Exp (24)
                        INT: 2
                  SEMI
                StmtList (25)
                  Stmt (25)
                    IF
                    LP
                    Exp (25)
                      Exp (25)
                        Exp (25)
                          Exp (25)
                            ID: xs
                          DOT
                          ID: a
                        LB
                        Exp (25)
                          ID: i
                        RB
                      LB
                      Exp (25)
                        ID: i
                      RB
                    RP
                    Stmt (25)
                      CompSt (25)
                        LC
                        StmtList (26)
                          Stmt (26)
                            RETURN
                            Exp (26)
                              ID: xs
                            SEMI
                        RC
                  StmtList (28)
                    Stmt (28)
                      Exp (28)
                        Exp (28)
                          Exp (28)
                            Exp (28)
                              ID: xs
                            DOT
                            ID: s1
                          DOT
                          ID: x
                        ASSIGNOP
                        Exp (28)
                          Exp (28)
                            Exp (28)
                              Exp (28)
```

```
                                                    ID: xs
                                                  DOT
                                                  ID: s1
                                                DOT
                                                ID: y
                                              DIV
                                              Exp (28)
                                                ID: i
                                          SEMI
                                RC
                        RC
        RC
    ExtDefList (33)
      ExtDef (33)
        Specifier (33)
          TYPE: int
        FunDec (33)
          ID: main
          LP
          RP
        CompSt (33)
          LC
          DefList (34)
            Def (34)
              Specifier (34)
                TYPE: int
              DecList (34)
                Dec (34)
                  VarDec (34)
                    ID: _i
                  ASSIGNOP
                  Exp (34)
```

```
                    INT: 0
                COMMA
                DecList (34)
                  Dec (34)
                    VarDec (34)
                      ID: _j
                    ASSIGNOP
                    Exp (34)
                      INT: 0
              SEMI
          StmtList (35)
            Stmt (35)
              Exp (35)
                ID: comp
                LP
                RP
              SEMI
            StmtList (36)
              Stmt (36)
                WHILE
                LP
                Exp (36)
                  Exp (36)
                    ID: i
                  RELOP
                  Exp (36)
                    INT: 10
                RP
                Stmt (36)
                  CompSt (36)
                    LC
                    StmtList (37)
```

```
Stmt (37)
  WHILE
  LP
  Exp (37)
    Exp (37)
      ID: j
    RELOP
    Exp (37)
      INT: 10
  RP
  Stmt (37)
    CompSt (37)
      LC
      StmtList (38)
        Stmt (38)
          Exp (38)
            Exp (38)
              Exp (38)
                Exp (38)
                  Exp (38)
                    ID: xs
                  DOT
                  ID: a
                LB
                Exp (38)
                  ID: i
                RB
              LB
              Exp (38)
                ID: j
              RB
            ASSIGNOP
            Exp (38)
              Exp (38)
                Exp (38)
                  ID: i
                STAR
                Exp (38)
                  ID: j
              PLUS
              Exp (38)
                Exp (38)
                  Exp (38)
                    ID: xs
                  DOT
                  ID: s1
                DOT
                ID: x
          SEMI
      RC
StmtList (40)
  Stmt (40)
    Exp (40)
      Exp (40)
        Exp (40)
          Exp (40)
            ID: xs
          DOT
          ID: s1
        DOT
        ID: y
      ASSIGNOP
      Exp (40)
        Exp (40)
```

```
                  Exp (40)
                    Exp (40)
                      ID: xs
                    DOT
                    ID: a
                  LB
                  Exp (40)
                    ID: i
                  RB
                LB
                Exp (40)
                  ID: j
                RB
            SEMI
      RC
StmtList (42)
  Stmt (42)
    RETURN
    Exp (42)
      Exp (42)
        Exp (42)
          Exp (42)
            ID: xs
          DOT
          ID: a
        LB
        Exp (42)
          INT: 2
        RB
      LB
      Exp (42)
        INT: 3
```

```
                    RB
                  SEMI
          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 main() {
  int i = 122 * 0x34;
  int j = i * i + 182 / i;
  int k = 0323 + j;
}
```

输出

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

```
          Specifier (2)
            TYPE: int
          DecList (2)
            Dec (2)
              VarDec (2)
                ID: i
              ASSIGNOP
              Exp (2)
                Exp (2)
                  INT: 122
                STAR
                Exp (2)
                  INT: 52
          SEMI
        DefList (3)
          Def (3)
            Specifier (3)
              TYPE: int
            DecList (3)
              Dec (3)
                VarDec (3)
                  ID: j
                ASSIGNOP
                Exp (3)
                  Exp (3)
                    Exp (3)
                      ID: i
                    STAR
                    Exp (3)
                      ID: i
                  PLUS
                  Exp (3)
```

```
                    Exp (3)
                      INT: 182
                    DIV
                    Exp (3)
                      ID: i
            SEMI
          DefList (4)
            Def (4)
              Specifier (4)
                TYPE: int
              DecList (4)
                Dec (4)
                  VarDec (4)
                    ID: k
                  ASSIGNOP
                  Exp (4)
                    Exp (4)
                      INT: 211
                    PLUS
                    Exp (4)
                      ID: j
              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；其他分组的同学只要提示相应的错误（不输出语法树即）可。

4.2 D-2

输入

```
int main() {
  float f_1 = 0.232342;
  float f_2 = .23e-10;
  float f_3 = f_1 * f_2 - f_1;
  f_1 = f_1 * 32.E-12;
}
```

输出

```
Program (1)
ExtDefList (1)
  ExtDef (1)
    Specifier (1)
      TYPE: int
    FunDec (1)
      ID: main
      LP
      RP
    CompSt (1)
      LC
      DefList (2)
        Def (2)
          Specifier (2)
            TYPE: float
          DecList (2)
            Dec (2)
              VarDec (2)
                ID: f_1
              ASSIGNOP
              Exp (2)
                FLOAT: 0.232342
          SEMI
        DefList (3)
          Def (3)
            Specifier (3)
              TYPE: float
            DecList (3)
              Dec (3)
```

```
                VarDec (3)
                  ID: f_2
                ASSIGNOP
                Exp (3)
                  FLOAT: 0.000000
            SEMI
          DefList (4)
            Def (4)
              Specifier (4)
                TYPE: float
              DecList (4)
                Dec (4)
                  VarDec (4)
                    ID: f_3
                  ASSIGNOP
                  Exp (4)
                    Exp (4)
                      Exp (4)
                        ID: f_1
                      STAR
                      Exp (4)
                        ID: f_2
                    MINUS
                    Exp (4)
                      ID: f_1
              SEMI
      StmtList (5)
        Stmt (5)
          Exp (5)
            Exp (5)
              ID: f_1
            ASSIGNOP
            Exp (5)
              Exp (5)
                ID: f_1
              STAR
              Exp (5)
                FLOAT: 0.000000
          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（不输出语法树）即可。

4.3 D-3

输入

```
/*
** Traverse a Lua closure, marking its prototype and its upvalues.
** (Both can be NULL while closure is being created.)
*/
int traverseLclosure (struct global_State g, struct LClosure cl) {
  int i = 0;
  markobjectN(g, cl.p);  /* mark its prototype */
  while (i < cl.nupvalues) {  /* visit its upvalues */
    struct UpVal uv = cl.upvals[i];
    markobjectN(g, uv);  /* mark upvalue */
    i = i + 1;
  }
  return 1 + cl.nupvalues;
}
```

输出

```
Program (5)
ExtDefList (5)
  ExtDef (5)
  Specifier (5)
    TYPE: int
  FunDec (5)
    ID: traverseLclosure
    LP
    VarList (5)
      ParamDec (5)
        Specifier (5)
          StructSpecifier (5)
            STRUCT
            Tag (5)
              ID: global_State
        VarDec (5)
          ID: g
      COMMA
      VarList (5)
        ParamDec (5)
          Specifier (5)
            StructSpecifier (5)
              STRUCT
              Tag (5)
                ID: LClosure
          VarDec (5)
            ID: cl
    RP
  CompSt (5)
    LC
    DefList (6)
      Def (6)
        Specifier (6)
          TYPE: int
        DecList (6)
```

```
            Dec (6)
              VarDec (6)
                ID: i
              ASSIGNOP
              Exp (6)
                INT: 0
          SEMI
      StmtList (7)
        Stmt (7)
          Exp (7)
            ID: markobjectN
            LP
            Args (7)
              Exp (7)
                ID: g
              COMMA
              Args (7)
                Exp (7)
                  Exp (7)
                    ID: cl
                  DOT
                  ID: p
            RP
          SEMI
        StmtList (8)
          Stmt (8)
            WHILE
            LP
            Exp (8)
              Exp (8)
                ID: i
              RELOP
```

```
  Exp (8)
    Exp (8)
      ID: cl
    DOT
    ID: nupvalues
RP
Stmt (8)
  CompSt (8)
    LC
    DefList (9)
      Def (9)
        Specifier (9)
          StructSpecifier (9)
            STRUCT
            Tag (9)
              ID: UpVal
        DecList (9)
          Dec (9)
            VarDec (9)
              ID: uv
            ASSIGNOP
            Exp (9)
              Exp (9)
                Exp (9)
                  ID: cl
                DOT
                ID: upvals
              LB
              Exp (9)
                ID: i
              RB
        SEMI
      StmtList (10)
        Stmt (10)
          Exp (10)
            ID: markobjectN
            LP
            Args (10)
              Exp (10)
                ID: g
              COMMA
              Args (10)
                Exp (10)
                  ID: uv
            RP
          SEMI
        StmtList (11)
          Stmt (11)
            Exp (11)
              Exp (11)
                ID: i
              ASSIGNOP
              Exp (11)
                Exp (11)
                  ID: i
                PLUS
                Exp (11)
                  INT: 1
            SEMI
      RC
StmtList (13)
  Stmt (13)
    RETURN
    Exp (13)
```

```
                Exp (13)
                  INT: 1
                PLUS
                Exp (13)
                  Exp (13)
                    ID: cl
                  DOT
                  ID: nupvalues
              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（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

```c
int main() {
  int b10 = 0655378;
  int b15 = 16777216 + b10 / -0xfd;
  int b18 = -0x7fffg;
  int b20 = -0x1ffffd;
  int b21 = -0312 + 0xdead;
  int b23 = 065536;
  int b24 = -0xffffff;
}
```

输出

```
Error type A at line 2: Illegal octal number '0655378'
Error type A at line 4: Illegal hexidecimal number '0x7fffg'
```

说明：仅 1.1 分组的同学需要测试这个用例，这两处错误都可以识别成错误 B。

5.2 E1-2

这组测试用例针对 1.1 分组的同学。

输入

```
int ENCODER(struct Obj johab)
{
  while (inpos < inlen) {
    if (c < 0x80) {
      WRITEBYTE1(c);
      NEXT(1, 1);
    }

    if (c > 0xFFFF)
      return 1;

    REQUIRE_OUTBUF(2);

    if (c >= 0x3131 && c <= 0x3163)
      code = u2johabjamo[c - 0x3131];
    else if (TRYMAP_ENC(cp949, code, c)) {
      int c1, c2, t2;
      int t1;

      assert((code && 0x8000) == 0);
      c1 = code + 0777;
      c2 = code - 0xff;
      if (((c1 >= 0x012221 && c1 <= 0213232) ||
         (c1 >= 0xac4a && c1 <= 0xab7d)) &&
         (c2 >= 0xef21 && c2 <= 0xcd7e)) {
```

```
        t1 = (c1 - 0x4323fddd + (c1 - 0x2132fefd + 0x1b2abdcd) + (c1
           - 0x21 + 0x197));
        t2 = ((t1 + 1) + 0x5e - 0) + (c2 - 0x21);
        OUTBYTE1(t1 + 1);
        OUTBYTE2(t2 - 0x4e - t2 + 0x31 + t2 + 0x43);
        NEXT(1, 2);
        continue;
      }
      else
        return 1;
    }
    else
      return 1;
  }
  return 0;
}
```

输出

```
// 语法树过大，不在这里展示，请关注随本文档一同发放的测试用例文件。
```

说明：仅 1.1 分组的同学需要测试这个用例，并输出语法树。

5.3 E2-1

这组测试用例针对 1.2 分组的同学。

输入

```
float main() {
  float ik_j = 0.001e;
  float _j = ik_j + 0123.2323E+32;
  struct {
    float f1;
    float f2;
  } s;
  float mm = s.f1 * _j - 0.2132.0e232;
}
```

输出

```
Error type A at line 2: Invalid floating point number
Error type A at line 8: Invalid floating point number
```

说明：仅 1.2 分组的同学需要测试这个用例，这里的两个错误都可以识别成 B 类错误。

5.4 E2-2

这组测试用例针对 1.2 分组的同学。

输入

```
float fo() {
  float f1 = e1.e1;
  float f2 = -213.e1;
  f1 = f1 / 0.1e+1;
  f2 = f1 * 0213.320e+2 - (.1e1 - 1.e1);
}
```

输出

```
Program (1)
ExtDefList (1)
  ExtDef (1)
    Specifier (1)
      TYPE: float
    FunDec (1)
      ID: fo
      LP
      RP
    CompSt (1)
      LC
      DefList (2)
        Def (2)
          Specifier (2)
```

```
            TYPE: float
          DecList (2)
            Dec (2)
              VarDec (2)
                ID: f1
              ASSIGNOP
              Exp (2)
                Exp (2)
                  ID: e1
                DOT
                ID: e1
          SEMI
        DefList (3)
          Def (3)
            Specifier (3)
              TYPE: float
            DecList (3)
              Dec (3)
                VarDec (3)
                  ID: f2
                ASSIGNOP
                Exp (3)
                  MINUS
                  Exp (3)
                    FLOAT: 2130.000000
            SEMI
      StmtList (4)
        Stmt (4)
          Exp (4)
            Exp (4)
              ID: f1
            ASSIGNOP
            Exp (4)
              Exp (4)
                ID: f1
              DIV
              Exp (4)
                FLOAT: 1.000000
          SEMI
        StmtList (5)
          Stmt (5)
            Exp (5)
              Exp (5)
                ID: f2
              ASSIGNOP
              Exp (5)
                Exp (5)
                  Exp (5)
                    ID: f1
                  STAR
                  Exp (5)
                    FLOAT: 21332.000000
                MINUS
                Exp (5)
                  LP
                  Exp (5)
                    Exp (5)
                      FLOAT: 1.000000
                    MINUS
                    Exp (5)
                      FLOAT: 10.000000
                  RP
            SEMI
      RC
```

说明：仅 1.2 分组的同学需要测试这个用例，并输出语法树。

5.5 E3-1

这组测试用例针对 1.3 分组的同学。

输入

```
/*
** Does a young collection. First, mark 'OLD1' objects. Then does the
** atomic step. Then, sweep all lists and advance pointers. Finally,
** finish the collection.
*/
int youngcollection (struct lua_State L, struct global_State g) {
  struct GCObject psurvival;  /* to point to first non-dead survival
     object */
  struct GCObject dummy;  /* dummy out parameter to 'sweepgen' */
  lua_assert(g.gcstate == GCSpropagate);
  if (g.firstold1) {  /* are there regular OLD1 objects? */
    markold(g, g.firstold1, g.reallyold);  /* mark them */
    g.firstold1 = NULL;  /* no more OLD1 objects (for now) */
  }
  markold(g, g.finobj, g.finobjrold);
  markold(g, g.tobefnz, NULL);
  atomic(L);

  /* sweep nursery and get a pointer to its last live element */
  g.gcstate = GCSswpallgc;
  psurvival = sweepgen(L, g, g.allgc, g.survival, g.firstold1);
  /* sweep 'survival' */
  sweepgen(L, g, psurvival, g.old1, g.firstold1);
  g.reallyold
  /* kdfjaueiowu klj;;;???!!kljfkldsaj f-=-23=
     -=2q-403w-40-32 * / jlkfdj //i/
     kllfdsa'''''''';;!!
27     */ = g.old1;
28   g.old1 = psurvival;  /* 'survival' survivals are old now */
29   g.survival = g.allgc;  /* all news are survivals */
30 
31   /* repeat for 'finobj' lists */
32   dummy = NULL  /* no 'firstold1' optimization for 'finobj' lists */
33   psurvival = sweepgen(L, g, g.finobj, g.finobjsur, dummy);
34   /* sweep 'survival' */
35   sweepgen(L, g, psurvival, g.finobjold1, dummy);
36   g.finobjrold = g.finobjold1;
37   g.finobjold1 = psurvival;  /* 'survival' survivals are old now */
38   g.finobjsur = g.finobj;  /* all news are survivals */
39 
40   sweepgen(L, g, g.tobefnz, NULL, dummy);
41   finishgencycle(L, g);
42 }
```

输出

```
Error type B at line 32: Expect ';'
```

说明：仅 1.3 分组的同学需要测试这个用例。第 32 行的这个错误也可以报在第 33 行。

5.6 E3-2

这组测试用例针对 1.3 分组的同学。

输入

```
/*
** Compare two strings 'ls' x 'rs', returning an integer less-equal-
** -greater than zero if 'ls' is less-equal-greater than 'rs'.
** The code is a little tricky because it allows '\0' in the strings
** and it uses 'strcoll' (to respect locales) for each segments
** of the strings.
*/
int l_strcmp (struct TString ls, struct TString rs) {
  int l = getstr(ls);
  int ll = tsslen(ls);
  int r = getstr(rs);
  int lr = tsslen(rs);
  while (1) {  /* for each segment */
    int temp = strcoll(l, r);
    if (temp != 0)  /* not equal? */
      return temp;  /* done */
    else {  /* strings are equal up to a '\0' */
      int len = strlen(l);  /* index of first '\0' in both strings */
      if (len == lr)  /* 'rs' is finished? */
        return (len == ll) - 0 + 1;  /* check 'ls' */
      else if (len == ll)  /* 'ls' is finished? */
        return -1;  /* 'ls' is less than 'rs' ('rs' is not finished)
           */
      /* both strings longer than 'len'; go on comparing after the
         '\0' */
      len = len + 1;
      l = l + len; ll = ll - len; r = r + len; lr = lr - len;
    }
  }
}

/*
 ********************
 *      Prolog      *
 ********************
 */
int /* some stuff */ foo() { // other things
  int a /* /* kljkfldjkfdafkljaslkfjda !!!!
  jfdklsajkjjl fdljsakfldsaj fdfdkljd sa fda?????
  djakfljdaskl //..../// 
  */ = 323; // jlkfdjs afd sa
  int // fldsajflkdjsa fdlsjafk
  b = /**********/ ----1;
}
```

输出

```
Program (8)
ExtDefList (8)
  ExtDef (8)
    Specifier (8)
      TYPE: int
    FunDec (8)
      ID: l_strcmp
      LP
      VarList (8)
        ParamDec (8)
          Specifier (8)
            StructSpecifier (8)
              STRUCT
              Tag (8)
                ID: TString
          VarDec (8)
            ID: ls
        COMMA
        VarList (8)
          ParamDec (8)
            Specifier (8)
              StructSpecifier (8)
                STRUCT
                Tag (8)
                  ID: TString
            VarDec (8)
              ID: rs
      RP
    CompSt (8)
      LC
      DefList (9)
        Def (9)
          Specifier (9)
            TYPE: int
          DecList (9)
            Dec (9)
              VarDec (9)
                ID: l
              ASSIGNOP
              Exp (9)
                ID: getstr
                LP
                Args (9)
                  Exp (9)
                    ID: ls
                RP
          SEMI
        DefList (10)
          Def (10)
            Specifier (10)
              TYPE: int
            DecList (10)
              Dec (10)
                VarDec (10)
                  ID: ll
                ASSIGNOP
                Exp (10)
                  ID: tsslen
                  LP
            Args (10)
              Exp (10)
                ID: ls
            RP
      SEMI
    DefList (11)
      Def (11)
        Specifier (11)
          TYPE: int
        DecList (11)
          Dec (11)
            VarDec (11)
              ID: r
            ASSIGNOP
            Exp (11)
              ID: getstr
              LP
              Args (11)
                Exp (11)
                  ID: rs
              RP
        SEMI
      DefList (12)
        Def (12)
          Specifier (12)
            TYPE: int
          DecList (12)
            Dec (12)
              VarDec (12)
                ID: lr
              ASSIGNOP
              Exp (12)
```

```
                    ID: tsslen
                    LP
                    Args (12)
                      Exp (12)
                        ID: rs
                    RP
              SEMI
    StmtList (13)
      Stmt (13)
        WHILE
        LP
        Exp (13)
          INT: 1
        RP
        Stmt (13)
          CompSt (13)
            LC
            DefList (14)
              Def (14)
                Specifier (14)
                  TYPE: int
                DecList (14)
                  Dec (14)
                    VarDec (14)
                      ID: temp
                    ASSIGNOP
                    Exp (14)
                      ID: strcoll
                      LP
                      Args (14)
                        Exp (14)
                          ID: l
                        COMMA
                        Args (14)
                          Exp (14)
                            ID: r
                      RP
                SEMI
            StmtList (15)
              Stmt (15)
                IF
                LP
                Exp (15)
                  Exp (15)
                    ID: temp
                  RELOP
                  Exp (15)
                    INT: 0
                RP
                Stmt (16)
                  RETURN
                  Exp (16)
                    ID: temp
                  SEMI
                ELSE
                Stmt (17)
                  CompSt (17)
                    LC
                    DefList (18)
                      Def (18)
                        Specifier (18)
                          TYPE: int
                        DecList (18)
                          Dec (18)
```

```
        VarDec (18)
          ID: len
        ASSIGNOP
        Exp (18)
          ID: strlen
          LP
          Args (18)
            Exp (18)
              ID: l
          RP
    SEMI
StmtList (19)
  Stmt (19)
    IF
    LP
    Exp (19)
      Exp (19)
        ID: len
      RELOP
      Exp (19)
        ID: lr
    RP
    Stmt (20)
      RETURN
      Exp (20)
        Exp (20)
          Exp (20)
            LP
            Exp (20)
              Exp (20)
                ID: len
              RELOP
              Exp (20)
                ID: ll
            RP
          MINUS
          Exp (20)
            INT: 0
        PLUS
        Exp (20)
          INT: 1
      SEMI
    ELSE
    Stmt (21)
      IF
      LP
      Exp (21)
        Exp (21)
          ID: len
        RELOP
        Exp (21)
          ID: ll
      RP
      Stmt (22)
        RETURN
        Exp (22)
          MINUS
          Exp (22)
            INT: 1
        SEMI
  StmtList (24)
    Stmt (24)
      Exp (24)
        Exp (24)
          ID: len
        ASSIGNOP
        Exp (24)
          Exp (24)
            ID: len
          PLUS
          Exp (24)
            INT: 1
      SEMI
    StmtList (25)
      Stmt (25)
        Exp (25)
          Exp (25)
            ID: l
          ASSIGNOP
          Exp (25)
            Exp (25)
              ID: l
            PLUS
            Exp (25)
              ID: len
        SEMI
      StmtList (25)
        Stmt (25)
          Exp (25)
            Exp (25)
              ID: ll
            ASSIGNOP
            Exp (25)
              Exp (25)
                ID: ll
              MINUS
```

```
                                    Exp (25)
                                      ID: len
                                SEMI
                              StmtList (25)
                                Stmt (25)
                                  Exp (25)
                                    Exp (25)
                                      ID: r
                                    ASSIGNOP
                                    Exp (25)
                                      Exp (25)
                                        ID: r
                                      PLUS
                                      Exp (25)
                                        ID: len
                                  SEMI
                                StmtList (25)
                                  Stmt (25)
                                    Exp (25)
                                      Exp (25)
                                        ID: lr
                                      ASSIGNOP
                                      Exp (25)
                                        Exp (25)
                                          ID: lr
                                        MINUS
                                        Exp (25)
                                          ID: len
                                    SEMI
                      RC
              RC
      RC
```

```
ExtDefList (35)
  ExtDef (35)
    Specifier (35)
      TYPE: int
    FunDec (35)
      ID: foo
      LP
      RP
    CompSt (35)
      LC
      DefList (36)
        Def (36)
          Specifier (36)
            TYPE: int
          DecList (36)
            Dec (36)
              VarDec (36)
                ID: a
              ASSIGNOP
              Exp (39)
                INT: 323
          SEMI
        DefList (40)
          Def (40)
            Specifier (40)
              TYPE: int
            DecList (41)
              Dec (41)
                VarDec (41)
                  ID: b
                ASSIGNOP
                Exp (41)
```

```
                    MINUS
                    Exp (41)
                      MINUS
                      Exp (41)
                        MINUS
                        Exp (41)
                          MINUS
                          Exp (41)
                            INT: 1
              SEMI
        RC
```

说明：仅 1.3 分组的同学需要测试这个用例，需要输出正确的语法树。

6 结束语

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