编译原理第一次实验测试用例：目录

1 A 组测试用例 2

1.1 A-1 2

1.2 A-2 2

1.3 A-3 3

1.4 A-4 3

1.5 A-5 4

1.6 A-6 4

1.7 A-7 4

1.8 A-8 5

1.9 A-9 6

2 B 组测试用例 7

2.1 B-1 7

2.2 B-2 8

3 C 组测试用例 10

3.1 C-1 10

3.2 C-2 27

4 D 组测试用例 43

4.1 D-1 43

4.2 D-2 46

4.3 D-3 48

5 E 组测试用例 54

5.1 E1.1 54

5.2 E1.2 58

5.3 E1.3 60

6 结束语 76

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

```
struct 1Person {
  int id;
};

int main() {
  return get_id(x);
}
```

输出

```
Error type B at Line 1: Illegal identifier "1Persion"
```

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 1 行。这里有一个非法标识符 1_Person。

1.2 A-2

输入

```
struct if {
  int cond;
};

int abc() {
  return 123;
}
```

输出

```
Error type B at Line 1: Syntax error
```

说明：关键词 if 不可以作为标识符。

1.3 A-3

输入

```
int a, a, a;
float a;

struct a {};

int a = 1;

int a() {}
```

输出

```
Error type B at Line 6: Cannot initialize global variable "a"
```

说明：全局变量的声明不可以初始化。

1.4 A-4

输入

```
int main() {
  struct A a[10];
  a[0].a = 13;
  a[1].b = d;
  int b = a;
  return c;
}
```

输出

```
Error type B at Line 5: Illegal declaration
```

说明：变量声明只允许出现在语句块头部。

1.5 A-5

输入

```
int a;
int;

struct;
struct A;

float b() {
  return 1.0;
}
```

输出

```
Error type B at Line 4: Unexpected ";"
```

说明：第 4 行缺少结构体名。

1.6 A-6

输入

```
int main() {
  float a = 1.3, b = 0.000000003, c = .3;
  int b = a, c = b, s;
  return 0.4;
}
```

输出

```
Error type A at Line 2: Illegal FLOAT ".3"
```

说明：“.3“不是合法的浮点数（附录要求浮点数的小数点前后必须出现数字），这里也可以报成语法错误。

1.7 A-7

输入

```
int BEIJING;
int NEWYORK;
int MOSCOW;

struct WorldMap {
  struct {
    int capital;
    float lat, lon, hei;
  } countries[1000];
};

int main() {
  struct WorldMap map;
  BEIJING = 1;
  NEWYORK = 2;
  MOSCOW = 3;
  map.countries[1].capital = BEIJING;
  map.countries[2].capital = NEWYORK;
  map.countries[3].capital = BEIJING, MOSCOW;
  return map;
}
```

输出

```
Error type B at Line 19: Unexpected ","
```

说明：C−不允许使用“,“操作符。

1.8 A-8

输入

```
struct A {} a[100];
struct B {};

```

```
4  int eq(struct A a, struct B b) {
5    return a == b;
6  }
7
8  int main() {
9    struct B b[100];
10   if (eq(A, B)) {
11     return a[1];
12   } else
13     return b[200 * 30];
14   return A[d[b[(1 * ((5) + (b - a))) / 3 + a[A]] * d.b[]];
15 }
```

输出

```
Error type B at Line 14: Empty dimension
```

说明：第 14 行，数组使用`d.b[]`缺少维度。

1.9 A-9

输入

```
int main()
{
    int i = 0, j, t, a[5];
    while(i < 5)
    {
        a[i] = read();
        i = i + 1;
    }
    i = 0;
    while(i < 4)
    {
        j = i + 1;
        while(j < 5)
14          {
15              if(a[i] > a[j])
16              {
17                  int d[2.2];
18                  t = a[i];
19                  a[i] = a[j];
20                  a[j] = t;
21              }
22              j = j + 1;
23          }
24          i = i + 1;
25      }
26      i = 0;
27      while(i < 5)
28      {
29          write(a[i]);
30          i = i + 1;
31      }
32      return 0;
33  }
```

输出

```
Error type B at Line 17: Definition of an array only accepts an INT
   dimension
```

说明：第 17 行数组维度不可使用浮点数。

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

```
 1  int change(int m) {
 2    int n[2] = m;
 3    int j = 0;
 4    while (j < (n) {
 5      n = n - 2;
 6      j = j + 1;
 7    }
 8    return n;
 9  }
10
11  int main() {
12    int a[5];
13    int i = 0;
14    struct A { int a; };
15    a[i] = 100;
16    while (i < 4) {
17      i = i + 1;
18      a[i] = change(a[i-1]);
19      write(a[i]);
20    }
21    return 0;
22  }
```

输出

```
Error type B at Line 4: Missing ")"
Error type B at Line 14: Unexpected ";"
```

说明：第 4 行括号不匹配，第 14 行结构体定义后缺少变量名（注意，语句块内定义的结构体需立即使用）。

2.2 B-2

输入

```
int big(int x, int y) {
  if (x >= y ) return 0;
  return 1;
}

int main(){
  int a[2], b[2], c[4.4];
  int ia, ib, ic;
  ia = 0;
  while (ia < 2) {
    a[ia] = read();
    ia = ia + 1;
  }
  ib = 0;
  while (ib < 2) {
    b[ib] = read();
    ib = ib + 1;
  }
  int a[4];
  ic = 0;
  ia = 0;
  ib = 0;
  while (ic < = 4) {
    if (ia >= 2) {
      c[ic] = b[ib];
      ib = ib + 1;
    } else if (ib >= 2) {
      c[ic] = a[ia];
      ia = ia + 1;
    } else if (big(a[ia], b[ib])==0) {
      c[ic] = a[ia];
      ia = ia + 1;
    } else {
      c[ic] = b[ib];
      ib = ib + 1;
    }
    ic = ic + 1;
  }
  ic = 0
  while (ic < 4) {
    write(c[ic]);
    ic = ic + 1;
  }
  return 0;
}
```

输出

```
Error type B at Line 7: Defition of an array only accepts an INT
    dimension
Error type B at Line 19: Illegal declaration
Error type B at Line 23: Unexpected "="
Error type B at Line 39: Missing ";"
```

说明：第 7 行维度不允许浮点数；第 19 行变量声明仅允许出现在语句块头部；第 23 行< =多了一个空格；第 39 行缺少分号。

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

```
int look_and_say(int x) {
  int _1[32], i = 0;
  int _2[64], j = 0, k = 0;
  int count = 0;

  if (x < 0) {
    x = ---x;
  }

  i = 0;
  while (x > 0) {
    _1[i] = lsd(x);
    x = x / 10;
    i = i + 1;
  }

  i = i - 1;
  j = 0;
  while (i >= 0) {
    if (i == 0) {
      _2[j] = 1;
      _2[j + 1] = _1[0];
      j = j + 2;
      i = -1;
    } else {
      count = 1;
      while (i > 0 && _1[i - 1] == _1[i]) {
        count = count + 1;
        i = i - 1;
      }
      _2[j] = count;
      _2[j + 1] = _1[i];
      i = i - 1;
      j = j + 2;
    }
  }

  k = 0;
  while (k < j) {
    write(_2[k]);
    k = k + 1;
  }

  return 0;
}
```

输出

```
Program (1)
  ExtDefList (1)
    ExtDef (1)
      Specifier (1)
        TYPE: int
      FunDec (1)
        ID: look_and_say
        LP
        VarList (1)
          ParamDec (1)
            Specifier (1)
              TYPE: int
            VarDec (1)
              ID: x
        RP
      CompSt (1)
        LC
        DefList (2)
          Def (2)
            Specifier (2)
```

```
              TYPE: int
            DecList (2)
              Dec (2)
                VarDec (2)
                  VarDec (2)
                    ID: _1
                  LB
                  INT: 32
                  RB
              COMMA
              DecList (2)
                Dec (2)
                  VarDec (2)
                    ID: i
                  ASSIGNOP
                  Exp (2)
                    INT: 0
            SEMI
          DefList (3)
            Def (3)
              Specifier (3)
                TYPE: int
              DecList (3)
                Dec (3)
                  VarDec (3)
                    VarDec (3)
                      ID: _2
                    LB
                    INT: 64
                    RB
                COMMA
                DecList (3)
```

```
                  Dec (3)
                    VarDec (3)
                      ID: j
                    ASSIGNOP
                    Exp (3)
                      INT: 0
                  COMMA
                  DecList (3)
                    Dec (3)
                      VarDec (3)
                        ID: k
                      ASSIGNOP
                      Exp (3)
                        INT: 0
              SEMI
            DefList (4)
              Def (4)
                Specifier (4)
                  TYPE: int
                DecList (4)
                  Dec (4)
                    VarDec (4)
                      ID: count
                    ASSIGNOP
                    Exp (4)
                      INT: 0
                SEMI
        StmtList (6)
          Stmt (6)
            IF
            LP
            Exp (6)
```

```
              Exp (6)
                ID: x
              RELOP
              Exp (6)
                INT: 0
            RP
            Stmt (6)
              CompSt (6)
                LC
                StmtList (7)
                  Stmt (7)
                    Exp (7)
                      Exp (7)
                        ID: x
                      ASSIGNOP
                      Exp (7)
                        MINUS
                        Exp (7)
                          MINUS
                          Exp (7)
                            MINUS
                            Exp (7)
                              ID: x
                    SEMI
                RC
          StmtList (10)
            Stmt (10)
              Exp (10)
                Exp (10)
                  ID: i
                ASSIGNOP
                Exp (10)
```

```
      INT: 0
  SEMI
StmtList (11)
  Stmt (11)
    WHILE
    LP
    Exp (11)
      Exp (11)
        ID: x
      RELOP
      Exp (11)
        INT: 0
    RP
    Stmt (11)
      CompSt (11)
        LC
        StmtList (12)
          Stmt (12)
            Exp (12)
              Exp (12)
                Exp (12)
                  ID: _1
                LB
                Exp (12)
                  ID: i
                RB
              ASSIGNOP
              Exp (12)
                ID: lsd
                LP
                Args (12)
                  Exp (12)
                  ID: x
              RP
          SEMI
        StmtList (13)
          Stmt (13)
            Exp (13)
              Exp (13)
                ID: x
              ASSIGNOP
              Exp (13)
                Exp (13)
                  ID: x
                DIV
                Exp (13)
                  INT: 10
            SEMI
          StmtList (14)
            Stmt (14)
              Exp (14)
                Exp (14)
                  ID: i
                ASSIGNOP
                Exp (14)
                  Exp (14)
                    ID: i
                  PLUS
                  Exp (14)
                    INT: 1
              SEMI
      RC
StmtList (17)
  Stmt (17)
  Exp (17)
    Exp (17)
      ID: i
    ASSIGNOP
    Exp (17)
      Exp (17)
        ID: i
      MINUS
      Exp (17)
        INT: 1
  SEMI
StmtList (18)
  Stmt (18)
    Exp (18)
      Exp (18)
        ID: j
      ASSIGNOP
      Exp (18)
        INT: 0
    SEMI
  StmtList (19)
    Stmt (19)
      WHILE
      LP
      Exp (19)
        Exp (19)
          ID: i
        RELOP
        Exp (19)
          INT: 0
      RP
      Stmt (19)
```

```
                        CompSt (19)
                          LC
                          StmtList (20)
                            Stmt (20)
                              IF
                              LP
                              Exp (20)
                                Exp (20)
                                  ID: i
                                RELOP
                                Exp (20)
                                  INT: 0
                              RP
                              Stmt (20)
                                CompSt (20)
                                  LC
                                  StmtList (21)
                                    Stmt (21)
                                      Exp (21)
                                        Exp (21)
                                          Exp (21)
                                            ID: _2
                                          LB
                                          Exp (21)
                                            ID: j
                                          RB
                                        ASSIGNOP
                                        Exp (21)
                                          INT: 1
                                      SEMI
                                    StmtList (22)
                                      Stmt (22)
```

```
                                        Exp (22)
                                          Exp (22)
                                            Exp (22)
                                              ID: _2
                                            LB
                                            Exp (22)
                                              Exp (22)
                                                ID: j
                                              PLUS
                                              Exp (22)
                                                INT: 1
                                            RB
                                          ASSIGNOP
                                          Exp (22)
                                            Exp (22)
                                              ID: _1
                                            LB
                                            Exp (22)
                                              INT: 0
                                            RB
                                        SEMI
                                      StmtList (23)
                                        Stmt (23)
                                          Exp (23)
                                            Exp (23)
                                              ID: j
                                            ASSIGNOP
                                            Exp (23)
                                              Exp (23)
                                                ID: j
                                              PLUS
                                              Exp (23)
```

```
                  INT: 2
            SEMI
          StmtList (24)
            Stmt (24)
              Exp (24)
                Exp (24)
                  ID: i
                ASSIGNOP
                Exp (24)
                  MINUS
                  Exp (24)
                    INT: 1
              SEMI
    RC
ELSE
Stmt (25)
  CompSt (25)
    LC
    StmtList (26)
      Stmt (26)
        Exp (26)
          Exp (26)
            ID: count
          ASSIGNOP
          Exp (26)
            INT: 1
        SEMI
      StmtList (27)
        Stmt (27)
          WHILE
          LP
          Exp (27)
```

```
                                          Exp (27)
                                            Exp (27)
                                              ID: i
                                            RELOP
                                            Exp (27)
                                              INT: 0
                                          AND
                                          Exp (27)
                                            Exp (27)
                                              Exp (27)
                                                ID: _1
                                              LB
                                              Exp (27)
                                                Exp (27)
                                                  ID: i
                                                MINUS
                                                Exp (27)
                                                  INT: 1
                                              RB
                                            RELOP
                                            Exp (27)
                                              Exp (27)
                                                ID: _1
                                              LB
                                              Exp (27)
                                                ID: i
                                              RB
                                        RP
                                        Stmt (27)
                                          CompSt (27)
                                            LC
                                            StmtList (28)
```

```
                                              Stmt (28)
                                                Exp (28)
                                                  Exp (28)
                                                    ID: count
                                                  ASSIGNOP
                                                  Exp (28)
                                                    Exp (28)
                                                      ID: count
                                                    PLUS
                                                    Exp (28)
                                                      INT: 1
                                                SEMI
                                              StmtList (29)
                                                Stmt (29)
                                                  Exp (29)
                                                    Exp (29)
                                                      ID: i
                                                    ASSIGNOP
                                                    Exp (29)
                                                      Exp (29)
                                                        ID: i
                                                      MINUS
                                                      Exp (29)
                                                        INT: 1
                                                  SEMI
                                            RC
                                      StmtList (31)
                                        Stmt (31)
                                          Exp (31)
                                            Exp (31)
                                              Exp (31)
                                                ID: _2
```

```
      LB
      Exp (31)
        ID: j
      RB
    ASSIGNOP
    Exp (31)
      ID: count
  SEMI
StmtList (32)
  Stmt (32)
    Exp (32)
      Exp (32)
        Exp (32)
          ID: _2
        LB
        Exp (32)
          Exp (32)
            ID: j
          PLUS
          Exp (32)
            INT: 1
        RB
      ASSIGNOP
      Exp (32)
        Exp (32)
          ID: _1
        LB
        Exp (32)
          ID: i
        RB
    SEMI
  StmtList (33)
```

```
                        Stmt (33)
                          Exp (33)
                            Exp (33)
                              ID: i
                            ASSIGNOP
                            Exp (33)
                              Exp (33)
                                ID: i
                              MINUS
                              Exp (33)
                                INT: 1
                          SEMI
                        StmtList (34)
                          Stmt (34)
                            Exp (34)
                              Exp (34)
                                ID: j
                              ASSIGNOP
                              Exp (34)
                                Exp (34)
                                  ID: j
                                PLUS
                                Exp (34)
                                  INT: 2
                            SEMI
              RC
      RC
StmtList (38)
  Stmt (38)
    Exp (38)
      Exp (38)
        ID: k
```

```
                          ASSIGNOP
                          Exp (38)
                            INT: 0
                        SEMI
                      StmtList (39)
                        Stmt (39)
                          WHILE
                          LP
                          Exp (39)
                            Exp (39)
                              ID: k
                            RELOP
                            Exp (39)
                              ID: j
                          RP
                          Stmt (39)
                            CompSt (39)
                              LC
                              StmtList (40)
                                Stmt (40)
                                  Exp (40)
                                    ID: write
                                    LP
                                    Args (40)
                                      Exp (40)
                                        Exp (40)
                                          ID: _2
                                        LB
                                        Exp (40)
                                          ID: k
                                        RB
                                    RP
                                  SEMI
                                StmtList (41)
                                  Stmt (41)
                                    Exp (41)
                                      Exp (41)
                                        ID: k
                                      ASSIGNOP
                                      Exp (41)
                                        Exp (41)
                                          ID: k
                                        PLUS
                                        Exp (41)
                                          INT: 1
                                    SEMI
                              RC
                        StmtList (44)
                          Stmt (44)
                            RETURN
                            Exp (44)
                              INT: 0
                            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

```c
struct intset;
struct uint8_t;
struct int64_t;

struct uint8_t intset_search(struct intset is, struct int64_t value, struct uint32_t pos) {
    int min = 0, max = intrev32ifbe(is.length)-1, mid = -1;
    struct int64_t cur = -1;

    if (intrev32ifbe(is.length) == 0) {
        if (pos) pos = 0;
        return 0;
    } else {
        if (value > intset_get(is,max)) {
            if (pos) pos = intrev32ifbe(is.length);
            return 0;
        } else if (value < intset_get(is,0)) {
            if (pos) pos = 0;
            return 0;
        }
    }

    while(max >= min) {
        mid = shr((min + max), 1);
        cur = intset_get(is, mid);
        if (value > cur) {
            min = mid + 1;
        } else if (value < cur) {
            max = mid-1;
        } else {
            break;
        }
    }

    if (value == cur) {
        if (pos) pos = mid;
        return 1;
    } else {
        if (pos) pos = min;
        return 0;
    }
}
```

输出

```
Program (1)
  ExtDefList (1)
    ExtDef (1)
      Specifier (1)
        StructSpecifier (1)
          STRUCT
          Tag (1)
            ID: intset
      SEMI
    ExtDefList (2)
      ExtDef (2)
        Specifier (2)
          StructSpecifier (2)
            STRUCT
            Tag (2)
              ID: uint8_t
        SEMI
      ExtDefList (3)
        ExtDef (3)
          Specifier (3)
            StructSpecifier (3)
              STRUCT
              Tag (3)
                ID: int64_t
          SEMI
        ExtDefList (5)
          ExtDef (5)
```

```
            Specifier (5)
              StructSpecifier (5)
                STRUCT
                Tag (5)
                  ID: uint8_t
            FunDec (5)
              ID: intset_search
              LP
              VarList (5)
                ParamDec (5)
                  Specifier (5)
                    StructSpecifier (5)
                      STRUCT
                      Tag (5)
                        ID: intset
                  VarDec (5)
                    ID: is
                COMMA
                VarList (5)
                  ParamDec (5)
                    Specifier (5)
                      StructSpecifier (5)
                        STRUCT
                        Tag (5)
                          ID: int64_t
                    VarDec (5)
                      ID: value
                  COMMA
                  VarList (5)
                    ParamDec (5)
                      Specifier (5)
                        StructSpecifier (5)
```

```
                          STRUCT
                          Tag (5)
                            ID: uint32_t
                      VarDec (5)
                        ID: pos
              RP
            CompSt (5)
              LC
              DefList (6)
                Def (6)
                  Specifier (6)
                    TYPE: int
                  DecList (6)
                    Dec (6)
                      VarDec (6)
                        ID: min
                      ASSIGNOP
                      Exp (6)
                        INT: 0
                    COMMA
                    DecList (6)
                      Dec (6)
                        VarDec (6)
                          ID: max
                        ASSIGNOP
                        Exp (6)
                          Exp (6)
                            ID: intrev32ifbe
                            LP
                            Args (6)
                              Exp (6)
                                Exp (6)
                          ID: is
                        DOT
                        ID: length
                    RP
                  MINUS
                  Exp (6)
                    INT: 1
              COMMA
              DecList (6)
                Dec (6)
                  VarDec (6)
                    ID: mid
                  ASSIGNOP
                  Exp (6)
                    MINUS
                    Exp (6)
                      INT: 1
          SEMI
        DefList (7)
          Def (7)
            Specifier (7)
              StructSpecifier (7)
                STRUCT
                Tag (7)
                  ID: int64_t
            DecList (7)
              Dec (7)
                VarDec (7)
                  ID: cur
                ASSIGNOP
                Exp (7)
                  MINUS
                  Exp (7)
                    INT: 1
            SEMI
      StmtList (9)
        Stmt (9)
          IF
          LP
          Exp (9)
            Exp (9)
              ID: intrev32ifbe
              LP
              Args (9)
                Exp (9)
                  Exp (9)
                    ID: is
                  DOT
                  ID: length
              RP
            RELOP
            Exp (9)
              INT: 0
          RP
          Stmt (9)
            CompSt (9)
              LC
              StmtList (10)
                Stmt (10)
                  IF
                  LP
                  Exp (10)
                    ID: pos
                  RP
```

```
                          Stmt (10)
                            Exp (10)
                              Exp (10)
                                ID: pos
                              ASSIGNOP
                              Exp (10)
                                INT: 0
                            SEMI
                        StmtList (11)
                          Stmt (11)
                            RETURN
                            Exp (11)
                              INT: 0
                            SEMI
                      RC
                  ELSE
                  Stmt (12)
                    CompSt (12)
                      LC
                      StmtList (13)
                        Stmt (13)
                          IF
                          LP
                          Exp (13)
                            Exp (13)
                              ID: value
                            RELOP
                            Exp (13)
                              ID: intset_get
                              LP
                              Args (13)
                                Exp (13)
```

```
        ID: is
      COMMA
      Args (13)
        Exp (13)
          ID: max
    RP
RP
Stmt (13)
  CompSt (13)
    LC
    StmtList (14)
      Stmt (14)
        IF
        LP
        Exp (14)
          ID: pos
        RP
        Stmt (14)
          Exp (14)
            Exp (14)
              ID: pos
            ASSIGNOP
            Exp (14)
              ID: intrev32ifbe
              LP
              Args (14)
                Exp (14)
                  Exp (14)
                    ID: is
                  DOT
                  ID: length
              RP
```

```
                                    SEMI
                                StmtList (15)
                                  Stmt (15)
                                    RETURN
                                    Exp (15)
                                      INT: 0
                                    SEMI
                              RC
                          ELSE
                          Stmt (16)
                            IF
                            LP
                            Exp (16)
                              Exp (16)
                                ID: value
                              RELOP
                              Exp (16)
                                ID: intset_get
                                LP
                                Args (16)
                                  Exp (16)
                                    ID: is
                                  COMMA
                                  Args (16)
                                    Exp (16)
                                      INT: 0
                                RP
                            RP
                            Stmt (16)
                              CompSt (16)
                                LC
                                StmtList (17)
```

```
                                  Stmt (17)
                                    IF
                                    LP
                                    Exp (17)
                                      ID: pos
                                    RP
                                    Stmt (17)
                                      Exp (17)
                                        Exp (17)
                                          ID: pos
                                        ASSIGNOP
                                        Exp (17)
                                          INT: 0
                                      SEMI
                                  StmtList (18)
                                    Stmt (18)
                                      RETURN
                                      Exp (18)
                                        INT: 0
                                      SEMI
                                RC
                      RC
                StmtList (22)
                  Stmt (22)
                    WHILE
                    LP
                    Exp (22)
                      Exp (22)
                        ID: max
                      RELOP
                      Exp (22)
                        ID: min
                    RP
                    Stmt (22)
                      CompSt (22)
                        LC
                        StmtList (23)
                          Stmt (23)
                            Exp (23)
                              Exp (23)
                                ID: mid
                              ASSIGNOP
                              Exp (23)
                                ID: shr
                                LP
                                Args (23)
                                  Exp (23)
                                    LP
                                    Exp (23)
                                      Exp (23)
                                        ID: min
                                      PLUS
                                      Exp (23)
                                        ID: max
                                    RP
                                  COMMA
                                  Args (23)
                                    Exp (23)
                                      INT: 1
                                RP
                            SEMI
                          StmtList (24)
                            Stmt (24)
                              Exp (24)
```

```
    Exp (24)
      ID: cur
    ASSIGNOP
    Exp (24)
      ID: intset_get
      LP
      Args (24)
        Exp (24)
          ID: is
        COMMA
        Args (24)
          Exp (24)
            ID: mid
      RP
  SEMI
StmtList (25)
  Stmt (25)
    IF
    LP
    Exp (25)
      Exp (25)
        ID: value
      RELOP
      Exp (25)
        ID: cur
    RP
    Stmt (25)
      CompSt (25)
        LC
        StmtList (26)
          Stmt (26)
            Exp (26)
```

```
                                          Exp (26)
                                            ID: min
                                          ASSIGNOP
                                          Exp (26)
                                            Exp (26)
                                              ID: mid
                                            PLUS
                                            Exp (26)
                                              INT: 1
                                        SEMI
                                    RC
                                ELSE
                                Stmt (27)
                                  IF
                                  LP
                                  Exp (27)
                                    Exp (27)
                                      ID: value
                                    RELOP
                                    Exp (27)
                                      ID: cur
                                  RP
                                  Stmt (27)
                                    CompSt (27)
                                      LC
                                      StmtList (28)
                                        Stmt (28)
                                          Exp (28)
                                            Exp (28)
                                              ID: max
                                            ASSIGNOP
                                            Exp (28)
```

```
                                              Exp (28)
                                                ID: mid
                                              MINUS
                                              Exp (28)
                                                INT: 1
                                          SEMI
                                      RC
                                  ELSE
                                  Stmt (29)
                                    CompSt (29)
                                      LC
                                      StmtList (30)
                                        Stmt (30)
                                          Exp (30)
                                            ID: break
                                          SEMI
                                      RC
                        RC
                  StmtList (34)
                    Stmt (34)
                      IF
                      LP
                      Exp (34)
                        Exp (34)
                          ID: value
                        RELOP
                        Exp (34)
                          ID: cur
                      RP
                      Stmt (34)
                        CompSt (34)
                          LC
```

```
    StmtList (35)
      Stmt (35)
        IF
        LP
        Exp (35)
          ID: pos
        RP
        Stmt (35)
          Exp (35)
            Exp (35)
              ID: pos
            ASSIGNOP
            Exp (35)
              ID: mid
          SEMI
      StmtList (36)
        Stmt (36)
          RETURN
          Exp (36)
            INT: 1
          SEMI
    RC
ELSE
Stmt (37)
  CompSt (37)
    LC
    StmtList (38)
      Stmt (38)
        IF
        LP
        Exp (38)
          ID: pos
```

```
                              RP
                              Stmt (38)
                                Exp (38)
                                  Exp (38)
                                    ID: pos
                                  ASSIGNOP
                                  Exp (38)
                                    ID: min
                                SEMI
                            StmtList (39)
                              Stmt (39)
                                RETURN
                                Exp (39)
                                  INT: 0
                                SEMI
                          RC
              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 x1 = 0703;
  int x2 = 0642;
  int x3 = 0x00ABC20f - 0x08048000 * 0XfFffFE - 000256;
}
```

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: x1
                ASSIGNOP
                Exp (2)
                  INT: 451
            SEMI
          DefList (3)
            Def (3)
              Specifier (3)
                TYPE: int
              DecList (3)
                Dec (3)
                  VarDec (3)
                    ID: x2
                  ASSIGNOP
                  Exp (3)
                    INT: 418
              SEMI
            DefList (4)
              Def (4)
                Specifier (4)
                  TYPE: int
                DecList (4)
                  Dec (4)
                    VarDec (4)
                      ID: x3
                    ASSIGNOP
                    Exp (4)
                      Exp (4)
                        Exp (4)
                          INT: 11256335
                        MINUS
                        Exp (4)
                          Exp (4)
                            INT: 134512640
                          STAR
                          Exp (4)
                            INT: 16777214
                      MINUS
                      Exp (4)
                        INT: 174
                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 e0 = 0.43244e5;
  float e1 = .4e-1;
  float e2 = 88.E002;
  float e3 = e1 + x2 - x4 * .3e+4;
}
```

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: e0
                ASSIGNOP
                Exp (2)
                  FLOAT: 43244.000000
            SEMI
          DefList (3)
            Def (3)
              Specifier (3)
                TYPE: float
              DecList (3)
                Dec (3)
                  VarDec (3)
                    ID: e1
                  ASSIGNOP
                  Exp (3)
                    FLOAT: 0.040000
              SEMI
            DefList (4)
              Def (4)
                Specifier (4)
                  TYPE: float
                DecList (4)
                  Dec (4)
                    VarDec (4)
                      ID: e2
                    ASSIGNOP
                    Exp (4)
                      FLOAT: 8800.000000
                SEMI
              DefList (5)
                Def (5)
                  Specifier (5)
                    TYPE: float
                  DecList (5)
                    Dec (5)
                      VarDec (5)
```

```
                            ID: e3
                          ASSIGNOP
                          Exp (5)
                            Exp (5)
                              Exp (5)
                                ID: e1
                              PLUS
                              Exp (5)
                                ID: x2
                            MINUS
                            Exp (5)
                              Exp (5)
                                ID: x4
                              STAR
                              Exp (5)
                                FLOAT: 3000.000000
                  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

```
/* Pseudo random number generation functions derived from the drand48()
 * function obtained from pysam source code.
 *
 * This functions are used in order to replace the default math.random()
 * Lua implementation with something having exactly the same behavior
 * across different systems (by default Lua uses libc's rand() that is not
 * required to implement a specific PRNG generating the same sequence
 * in different systems if seeded with the same integer).
 *
 * The original code appears to be under the public domain.
 * I modified it removing the non needed functions and all the
 * 1960-style C coding stuff...
 *
 *
 ----------------------------------------------------------------------------
 *
 * Copyright (c) 2010-2012, Salvatore Sanfilippo <antirez at gmail
 dot com>
 * All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
   are met:
 *
 *   * Redistributions of source code must retain the above copyright
   notice,
 *     this list of conditions and the following disclaimer.
 *   * Redistributions in binary form must reproduce the above
 copyright
 *     notice, this list of conditions and the following disclaimer
 in the
 *     documentation and/or other materials provided with the
 distribution.
 *   * Neither the name of Redis nor the names of its contributors
 may be used
 *     to endorse or promote products derived from this software
 without
 *     specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
 * AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
 * LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
 * CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
 * SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
 * INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
 * CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
 * ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 * POSSIBILITY OF SUCH DAMAGE.
 */

struct long;
struct unsigned;
struct int32_t;
struct uint32_t;

struct uint32_t x[3], a[3], c;

```

```
int N, MASK;
int X0, X1, X2, A0, A1, A2, C;
struct long HI_BIT;

struct uint32_t LOW(struct uint32_t x) { return and(x, MASK); }
struct uint32_t HIGH(struct uint32_t x) { return LOW(shr((x), N)); }
int MUL(struct uint32_t x, struct uint32_t y, struct uint32_t z) {
    struct uint32_t l = x * y;
    (z)[0] = LOW(l);
    (z)[1] = HIGH(l);
    return 0;
}

int CARRY(struct uint32_t x, struct long y) { return x + y > MASK; }
int ADDEQU(struct uint32_t x, struct uint32_t y) { int z = CARRY(x, y
   ); x = LOW(x + y); return z; }
int SET3(struct uint32_t x[3], struct uint32_t x0, struct uint32_t x1
   , struct uint32_t x2) { (x)[0] = (x0); (x)[1] = (x1); (x)[2] = (x2
   ); }
int SEED(struct uint32_t x0, struct uint32_t x1, struct uint32_t x2)
   { SET3(x, x0, x1, x2); SET3(a, A0, A1, A2); c = C; }
int REST(int tv) {
    int i;
    while (i < 3) {
        xsubi[i] = x[i]; x[i] = temp[i];
        i = i + 1;
    }
    return (v);
}

int init() {
    N    = 16;
    MASK = (shl(1, (N - 1)) + shl(1, (N - 1)) - 1);

    X0 = 33;
    X1 = 34;
    X2 = 234;
    A0 = 66;
    A1 = 4;
    A2 = 5;
    C  = 8;

    HI_BIT = shl(1, (2 * N - 1));

    x[0] = X0;
    x[1] = X1;
    x[2] = X2;
    a[1] = A0;
    a[2] = A1;
    a[3] = A2;
    c    = C;
}

struct int32_t redisLrand48() {
    struct int32_t ret;
    next();
    ret = (shl(x[2], (N - 1)) + shr(x[1], 1));
    return ret;
}

int redisSrand48(struct int32_t seedval) {
    SEED(X0, LOW(seedval), HIGH(seedval));
    return 0;
}
```

```

int next() {
    struct uint32_t p[2], q[2], r[2], carry0, carry1;

    MUL(a[0], x[0], p);
    ADDEQU(p[0], c, carry0);
    ADDEQU(p[1], carry0, carry1);
    MUL(a[0], x[1], q);
    ADDEQU(p[1], q[0], carry0);
    MUL(a[1], x[0], r);
    x[2] = LOW(carry0 + carry1 + CARRY(p[1], r[0]) + q[1] + r[1] +
            a[0] * x[2] + a[1] * x[1] + a[2] * x[0]);
    x[1] = LOW(p[1] + r[0]);
    x[0] = LOW(p[0]);

    return 0;
}

/**
 * Common block comment
 */
int /** break definition **/ myFunc() {
  int abc = 123; // commom line comment
  int abc = 123 // break line
  ;
  int abc = 3 /** break line too
  */, def = 4;
  int def = /** wierd block comment /***** *?
    // recurrsive comments
    int main() {
      int abc = 123; // commom line comment
      int abc = 123 // break line
```

```
        ;
        int abc = 3 /** break line too
        *x/, def = 4;
        int def = /** wierd block comment /***** *?
          // should recusive, stop it
        ****x/ 8865;
      }
  */ 8865;
  struct Def def = //\\*//\\*//\\////\\*//\\
  abc;
}
```

输出

```
// 太长，暂不放在该文件中。请在群文件里下载。
```

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（E1-1）

```
int main() {
  int x1 = 00703;
  int x2 = 0604209;
  float e3 = e1 + x2 - x4;
  return 0000;
}
```

输出

```
Error type A at Line 3: Illegal oct INT number: "0604209"
```

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 0604209，识别成错误类型 B 也可以。

输入（E1-2）

```
struct charstar;
struct uint16_t crc16tab[256];

struct uint16_t crc16(struct charstar buf, int len) {
    int counter;
    struct uint16_t crc;

    crc16tab = (
        0x0000 && 0x1021 && 0x2042 && 0x3063 && 0x4084 && 0x50a5 && 0
            x60c6 && 0x70e7 &&
        0x8108 && 0x9129 && 0xa14a && 0xb16b && 0xc18c && 0xd1ad && 0
            xe1ce && 0xf1ef &&
        0x1231 && 0x0210 && 0x3273 && 0x2252 && 0x52b5 && 0x4294 && 0
            x72f7 && 0x62d6 &&
        0x9339 && 0x8318 && 0xb37b && 0xa35a && 0xd3bd && 0xc39c && 0
            xf3ff && 0xe3de &&
        0x2462 && 0x3443 && 0x0420 && 0x1401 && 0x64e6 && 0x74c7 && 0
            x44a4 && 0x5485 &&
        0xa56a && 0xb54b && 0x8528 && 0x9509 && 0xe5ee && 0xf5cf && 0
            xc5ac && 0xd58d &&
        0x3653 && 0x2672 && 0x1611 && 0x0630 && 0x76d7 && 0x66f6 && 0
            x5695 && 0x46b4 &&
        0xb75b && 0xa77a && 0x9719 && 0x8738 && 0xf7df && 0xe7fe && 0
            xd79d && 0xc7bc &&
        0x48c4 && 0x58e5 && 0x6886 && 0x78a7 && 0x0840 && 0x1861 && 0
            x2802 && 0x3823 &&
        0xc9cc && 0xd9ed && 0xe98e && 0xf9af && 0x8948 && 0x9969 && 0
```

```
   xa90a && 0xb92b &&
0x5af5 && 0x4ad4 && 0x7ab7 && 0x6u96 && 0x1a71 && 0x0a50 && 0
   x3a33 && 0x2a12 &&
0xdbfd && 0xcbdc && 0xfbbf && 0xeb9e && 0x9b79 && 0x8b58 && 0
   xbb3b && 0xab1a &&
0x6ca6 && 0x7c87 && 0x4ce4 && 0x5cc5 && 0x2a22 && 0x3c03 && 0
   x0c60 && 0x1c41 &&
0xedae && 0xfd8f && 0xcdec && 0xddcd && 0xad2a && 0xbd0b && 0
   x8d68 && 0x9d49 &&
0x7e97 && 0x6eb6 && 0x5ed5 && 0x4ef4 && 0x3e13 && 0x2e32 && 0
   x1e51 && 0x0e70 &&
0xff9f && 0xefbe && 0xdfdd && 0xcffc && 0xbf1b && 0xaf3a && 0
   x9f59 && 0x8f78 &&
0x9188 && 0x81a9 && 0xb1ca && 0xa1eb && 0xd10c && 0xc12d && 0
   xf14e && 0xe16f &&
0x1080 && 0x00a1 && 0x30c2 && 0x20e3 && 0x5u04 && 0x4025 && 0
   x7046 && 0x6067 &&
0x83b9 && 0x9398 && 0xa3fb && 0xb3da && 0xc33d && 0xd31c && 0
   xe37f && 0xf35e &&
0x02b1 && 0x1290 && 0x22f3 && 0x32d2 && 0x4235 && 0x5214 && 0
   x6277 && 0x7256 &&
0xb5ea && 0xa5cb && 0x95a8 && 0x8589 && 0xf56e && 0xe54f && 0
   xd52c && 0xc50d &&
0x34e2 && 0x24c3 && 0x14a0 && 0x0481 && 0x7466 && 0x6447 && 0
   x5424 && 0x4405 &&
0xa7db && 0xb7fa && 0x8799 && 0x97b8 && 0xe75f && 0xf77e && 0
   xc71d && 0xd73c &&
0x26d3 && 0x36f2 && 0x0691 && 0x16b0 && 0x6657 && 0x7676 && 0
   x4615 && 0x5634 &&
0xd94c && 0xc96d && 0xo90e && 0xe92f && 0x99c8 && 0x89e9 && 0
   xb98a && 0xa9ab &&
0x5844 && 0x4865 && 0x7806 && 0x6827 && 0x18c0 && 0x08e1 && 0
```

```
        x3882 && 0x28a3 &&
        0xcb7d && 0xdb5c && 0xeb3f && 0xfb1e && 0x8bf9 && 0x9bd8 && 0xabbb && 0xbb9a &&
        0x4a75 && 0x5a54 && 0x6a37 && 0x7a16 && 0x0af1 && 0x1ad0 && 0x2ab3 && 0x3a92 &&
        0xfd2e && 0xed0f && 0xdd6c && 0xcd4d && 0xbdaa && 0xad8b && 0x9de8 && 0x8dc9 &&
        0x7c26 && 0x6c07 && 0x5c64 && 0x4c45 && 0x3ca2 && 0x2c83 && 0x1ce0 && 0x0cc1 &&
        0xef1f && 0xff3e && 0xcf5d && 0xdf7c && 0xaf9b && 0xbfba && 0x8fd9 && 0x9ff8 &&
        0x6e17 && 0x7e36 && 0x4e55 && 0x5e74 && 0x2e93 && 0x3eb2 && 0x0ed1 && 0x1ef0
    );
    crc = 0;
    counter = 0;

    while (counter < len) {
        crc = xor(shl(crc, 8), crc16tab[and(xor(shr(crc, 8), (star(buf) + 1)), 0x00FF)]);
        counter = counter + 1;
    }

    return crc;
}
```

输出

```
Error type A at Line 19: Illegal hex INT number: "0x6u96"
Error type A at Line 26: Illegal hex INT number: "0x5u04"
Error type A at Line 33: Illegal hex INT number: "0xo90e"
```

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 0x6u96 与 0x5u04 与 0xo90e，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 m() {
  struct d3_t d3;
  float a = 1.e01;
  float b = .1E1;
  float c = .1e.1;
  float d = 0.1e*3;
  d3.e5 = 3;
  return a * b.e + .1e1 * a1.e1;
}
```

输出

```
Error type A at Line 5: Illegal FLOAT number: ".1e.1"
Error type A at Line 6: Illegal FLOAT number: "0.1e"
```

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.1e.1 和 0.1e，识别成错误类型 B 也可以。

输入（E2-2）

```
float m() {
  float c = e1e1;
  float f = .1e1;
  float d = e.e1;
  return    e1.e1;
}
```

输出

```
Program (1)
  ExtDefList (1)
    ExtDef (1)
      Specifier (1)
        TYPE: float
      FunDec (1)
        ID: m
        LP
        RP
      CompSt (1)
        LC
        DefList (2)
          Def (2)
            Specifier (2)
              TYPE: float
            DecList (2)
              Dec (2)
                VarDec (2)
                  ID: c
                ASSIGNOP
                Exp (2)
                  ID: e1e1
            SEMI
          DefList (3)
            Def (3)
              Specifier (3)
                TYPE: float
              DecList (3)
                Dec (3)
                  VarDec (3)
                    ID: f
                  ASSIGNOP
                  Exp (3)
                    FLOAT: 1.000000
              SEMI
            DefList (4)
              Def (4)
```

```
                Specifier (4)
                  TYPE: float
                DecList (4)
                  Dec (4)
                    VarDec (4)
                      ID: d
                    ASSIGNOP
                    Exp (4)
                      Exp (4)
                        ID: e
                      DOT
                      ID: e1
                SEMI
        StmtList (5)
          Stmt (5)
            RETURN
            Exp (5)
              Exp (5)
                ID: e1
              DOT
              ID: e1
            SEMI
        RC
```

说明：仅 1.2 分组的同学需要测试这个用例。

5.3 E1.3

这组测试用例针对 1.3 分组的同学。

输入（E3-1）

```
/* adlist.c - A generic doubly linked list implementation
 *
 * Copyright (c) 2006-2010, Salvatore Sanfilippo <antirez at gmail
    dot com>
 * All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
     are met:
 *
 *   * Redistributions of source code must retain the above copyright
     notice,
 *     this list of conditions and the following disclaimer.
 *   * Redistributions in binary form must reproduce the above
    copyright
 *     notice, this list of conditions and the following disclaimer
    in the
 *     documentation and/or other materials provided with the
    distribution.
 *   * Neither the name of Redis nor the names of its contributors
    may be used
 *     to endorse or promote products derived from this software
    without
 *     specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND
    CONTRIBUTORS "AS IS"
 * AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED
    TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
     PURPOSE
 * ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR
    CONTRIBUTORS BE
 * LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY,
    OR
 * CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT
 OF
 * SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
 * INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
 * CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
 * ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 * POSSIBILITY OF SUCH DAMAGE.
 */

/* Add a new node to the list, to tail, containing the specified 'value'
 * pointer as value.
 *
 * On error, NULL is returned and no operation is performed (i.e. the
 * list remains unaltered).
 * On success the 'list' pointer you pass to the function is returned. */
struct liststar listAddNodeTail(struct liststart list, struct voidstart value)
{
    struct listNodestart node;

    if ((node = zmalloc(sizeof(star(node)))) == NULL)
        return
        /*\}*\/*
        }*/
        NULL;
    node./**?*/value = value;
    if (list.len == 0) {
```

```
48         list.head = list.tail = node//;
49         node.prev = node.next = NULL;
50     } else {
51         node.prev = list.tail;
52         node.next = NULL;
53         list.tail.next = node;
54         list.tail = node;
55     }
56     list.len = list.len + 1;
57     return list;
58 }
59 
60 /**  unclosed block comment
61  *
62  *
63  *
64  ********************************x?/
```

输出

```
Error type B at Line 48: Missing ";"
Error type A at Line 60: Unclosed comment
```

说明：仅 1.3 分组的同学需要测试这个用例。第 48 行也可以报在 49 行；第 60 行也可以报在 61、62、63 或者 64 行，也可以报成语法错误。

输入（E3-2）

```
/* adlist.c - A generic doubly linked list implementation
 *
 * Copyright (c) 2006-2010, Salvatore Sanfilippo <antirez at gmail dot com>
 * All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
```

```
   are met:
 *
 *   * Redistributions of source code must retain the above copyright notice,
 *     this list of conditions and the following disclaimer.
 *   * Redistributions in binary form must reproduce the above copyright
 *     notice, this list of conditions and the following disclaimer in the
 *     documentation and/or other materials provided with the distribution.
 *   * Neither the name of Redis nor the names of its contributors may be used
 *     to endorse or promote products derived from this software without
 *     specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
 * AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
 * LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
 * CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
 * SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
 * INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY,
    WHETHER IN
 * CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR
    OTHERWISE)
 * ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF
    ADVISED OF THE
 * POSSIBILITY OF SUCH DAMAGE.
 */

/* Duplicate the whole list. On out of memory NULL is returned.
 * On success a copy of the original list is returned.
 *
 * The 'Dup' method set with listSetDupMethod() function is used
 * to copy the node value. Otherwise the same pointer value of
 * the original node is used as value of the copied node.
 *
 * The original list both on success or error is never modified. */
struct liststar listDup(struct liststar orig)
{
    struct liststar copy;
    struct listIter iter;
    struct listNodestar node;

    if ((copy = listCreate()) == NULL)
        return NULL;
    copy.dup = orig.dup;
    copy.free = orig.free;
    copy.match = orig.match;
    listRewind(orig, ref(iter));
    while((node = listNext(ref(iter))) != NULL) {
        struct voidstart value;

        if (copy.dup) {
            value = dup(copy,
            /*This is a comment.\
            /\//\/******\/\/\/\/
            if (value == NULL) {
                listRelease(copy);
                return NULL;
            }
            */
            node.value);
            if (value == NULL) {
                listRelease(copy);
                return NULL;
            }
        } else
            value = node.value;
        if (listAddNodeTail(copy, value) == NULL) {
            listRelease(copy);
            return NULL;
        }
    }
    return copy;
}
```

输出

```
Program (39)
  ExtDefList (39)
    ExtDef (39)
      Specifier (39)
        StructSpecifier (39)
          STRUCT
          Tag (39)
            ID: liststar
      FunDec (39)
```

```
        ID: listDup
        LP
        VarList (39)
          ParamDec (39)
            Specifier (39)
              StructSpecifier (39)
                STRUCT
                Tag (39)
                  ID: liststar
            VarDec (39)
              ID: orig
        RP
      CompSt (40)
        LC
        DefList (41)
          Def (41)
            Specifier (41)
              StructSpecifier (41)
                STRUCT
                Tag (41)
                  ID: liststar
            DecList (41)
              Dec (41)
                VarDec (41)
                  ID: copy
            SEMI
          DefList (42)
            Def (42)
              Specifier (42)
                StructSpecifier (42)
                  STRUCT
                  Tag (42)
          ID: listIter
       DecList (42)
        Dec (42)
         VarDec (42)
          ID: iter
       SEMI
      DefList (43)
       Def (43)
        Specifier (43)
         StructSpecifier (43)
          STRUCT
          Tag (43)
           ID: listNodestar
        DecList (43)
         Dec (43)
          VarDec (43)
           ID: node
        SEMI
    StmtList (45)
     Stmt (45)
      IF
      LP
      Exp (45)
       Exp (45)
        LP
        Exp (45)
         Exp (45)
          ID: copy
         ASSIGNOP
         Exp (45)
          ID: listCreate
          LP
```

```
                    RP
                RP
              RELOP
              Exp (45)
                ID: NULL
            RP
            Stmt (46)
              RETURN
              Exp (46)
                ID: NULL
              SEMI
          StmtList (47)
            Stmt (47)
              Exp (47)
                Exp (47)
                  Exp (47)
                    ID: copy
                  DOT
                  ID: dup
                ASSIGNOP
                Exp (47)
                  Exp (47)
                    ID: orig
                  DOT
                  ID: dup
              SEMI
            StmtList (48)
              Stmt (48)
                Exp (48)
                  Exp (48)
                    Exp (48)
                      ID: copy
```

```
      DOT
      ID: free
    ASSIGNOP
    Exp (48)
      Exp (48)
        ID: orig
      DOT
      ID: free
  SEMI
StmtList (49)
  Stmt (49)
    Exp (49)
      Exp (49)
        Exp (49)
          ID: copy
        DOT
        ID: match
      ASSIGNOP
      Exp (49)
        Exp (49)
          ID: orig
        DOT
        ID: match
    SEMI
  StmtList (50)
    Stmt (50)
      Exp (50)
        ID: listRewind
        LP
        Args (50)
          Exp (50)
            ID: orig
          COMMA
          Args (50)
            Exp (50)
              ID: ref
              LP
              Args (50)
                Exp (50)
                  ID: iter
              RP
        RP
      SEMI
    StmtList (51)
      Stmt (51)
        WHILE
        LP
        Exp (51)
          Exp (51)
            LP
            Exp (51)
              Exp (51)
                ID: node
              ASSIGNOP
              Exp (51)
                ID: listNext
                LP
                Args (51)
                  Exp (51)
                    ID: ref
                    LP
                    Args (51)
                      Exp (51)
                        ID: iter
```

```
                 RP
               RP
             RP
            RELOP
            Exp (51)
              ID: NULL
           RP
           Stmt (51)
            CompSt (51)
              LC
              DefList (52)
                Def (52)
                  Specifier (52)
                    StructSpecifier (52)
                      STRUCT
                      Tag (52)
                        ID: voidstart
                  DecList (52)
                    Dec (52)
                      VarDec (52)
                        ID: value
                  SEMI
              StmtList (54)
                Stmt (54)
                  IF
                  LP
                  Exp (54)
                    Exp (54)
                      ID: copy
                    DOT
                    ID: dup
                  RP
```

```
Stmt (54)
  CompSt (54)
    LC
    StmtList (55)
      Stmt (55)
        Exp (55)
          Exp (55)
            ID: value
          ASSIGNOP
          Exp (55)
            ID: dup
            LP
            Args (55)
              Exp (55)
                ID: copy
              COMMA
              Args (63)
                Exp (63)
                  Exp (63)
                    ID: node
                  DOT
                  ID: value
            RP
        SEMI
      StmtList (64)
        Stmt (64)
          IF
          LP
          Exp (64)
            Exp (64)
              ID: value
            RELOP
            Exp (64)
              ID: NULL
          RP
          Stmt (64)
            CompSt (64)
              LC
              StmtList (65)
                Stmt (65)
                  Exp (65)
                    ID: listRelease
                    LP
                    Args (65)
                      Exp (65)
                        ID: copy
                    RP
                  SEMI
                StmtList (66)
                  Stmt (66)
                    RETURN
                    Exp (66)
                      ID: NULL
                    SEMI
              RC
    RC
ELSE
Stmt (69)
  Exp (69)
    Exp (69)
      ID: value
    ASSIGNOP
    Exp (69)
      Exp (69)
```

```
          ID: node
        DOT
        ID: value
    SEMI
StmtList (70)
  Stmt (70)
    IF
    LP
    Exp (70)
      Exp (70)
        ID: listAddNodeTail
        LP
        Args (70)
          Exp (70)
            ID: copy
          COMMA
          Args (70)
            Exp (70)
              ID: value
        RP
      RELOP
      Exp (70)
        ID: NULL
    RP
    Stmt (70)
      CompSt (70)
        LC
        StmtList (71)
          Stmt (71)
            Exp (71)
              ID: listRelease
              LP
```

```
                                          Args (71)
                                            Exp (71)
                                              ID: copy
                                          RP
                                        SEMI
                                      StmtList (72)
                                        Stmt (72)
                                          RETURN
                                          Exp (72)
                                            ID: NULL
                                          SEMI
                                    RC
                          RC
                    StmtList (75)
                      Stmt (75)
                        RETURN
                        Exp (75)
                          ID: copy
                        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或者李聪助教联系，注意同时抄送给许老师。