

编译原理第二次实验测试用例：目录

1	A 组测试用例	3
1.1	A-1	3
1.2	A-2	3
1.3	A-3	4
1.4	A-4	4
1.5	A-5	5
1.6	A-6	6
1.7	A-7	7
1.8	A-8	7
1.9	A-9	8
1.10	A-10	9
1.11	A-11	10
1.12	A-12	11
1.13	A-13	11
1.14	A-14	12
1.15	A-15	13
1.16	A-16	13
1.17	A-17	14
1.18	A-18	15
2	B 组测试用例	15
2.1	B-1	15
3	C 组测试用例	18
3.1	C-1	18
3.2	C-2	20
4	D 组测试用例	21
4.1	D-1	21
4.2	D-2	23
4.3	D-3	25

5	E 组测试用例	27
5.1	E2.1	27
5.2	E2.2	29
5.3	E2.3	31
6	结束语	33

1 A 组测试用例

本组测试用例共 18 个，测试用例 1-17 分别对应语义错误 1-17，第 18 个测试用例对应于语义错误 15。每个用例仅在其中一行含有语义错误。某些语义错误可能会产生连锁反应。测试用例 A-i 对应的“本质错误”就是错误类型 i，因此错误类型 i 是必须报出来的，如果报出其他错误，只要是由本质错误连带引发的（包括但不限于下面明确给出的情况），我们都不会扣分。

1.1 A-1

输入

```
1 int add(int a, int b)
2 {
3     int x, y;
4     x = a;
5     y = b;
6     z = x + y;
7 }
```

输出

```
1 Error Type 1 at line 6: Semantic Error, Undefined variable
2     Variable 'z' is previously undefined
```

说明： $z = x + y$ 这一句包含未定义的变量 z ，这里也可以另外报出错误类型 5（加号两边类型不匹配）。

1.2 A-2

输入

```
1 int print(int a)
2 {
3     int x = 0;
4     int y = a + x;
5     printf(y);
6 }
```

输出

```
1 Error Type 2 at line 5: Semantic Error, Undefined function
2     Function 'printf' is previously undefined
```

说明：printf 未定义。

1.3 A-3

输入

```
1 struct a {
2     int x;
3     int y;
4 };
5
6 int main()
7 {
8     int a = 3;
9 }
```

输出

```
1 Error Type 3 at line 8: Semantic Error, Redefined variable
2     Variable 'a' is previously defined / Variable name 'a'
      conflicts struct 'a' previously defined
```

说明：重复定义的变量 a，这里如果错误位置写为第 1 行也算对。

1.4 A-4

输入

```
1 int multiply(int a, int b)
2 {
3     int c = a * b;
4     return c;
5 }
6
7 int main()
```

```

8 {
9     int x = 1;
10    int y = -1;
11    return multiply(x, y);
12 }
13
14 int multiply(int u, int v)
15 {
16     int z = u + v;
17     return z;
18 }

```

输出

```

1 Error Type 4 at line 14: Semantic Error, Redefined function
2     Function 'multiply' is previously defined

```

说明：重复定义的函数 `multiply`。这里如果没有把重复定义的函数放入符号表，会在第 11 行报了错误类型 2，是否报出这个错误，不影响得分。

1.5 A-5

输入

```

1 struct Male {
2     int age;
3     int weight;
4 };
5
6 struct Female {
7     int circumference[3];
8     float w;
9 };
10
11 int main()
12 {

```

```

13     struct Male a;
14     struct Female b;
15     a = b;
16 }

```

输出

```

1 Error Type 5 at line 15: Semantic Error, Incompatible types when
   assigning

```

说明：赋值号两边类型不匹配（无论结构等价还是名等价）

1.6 A-6

输入

```

1 int Exchange(int x, int y)
2 {
3     int i = 0;
4     if(x >= 0) {
5         while(i < x)
6             i = i + 1;
7         x = y;
8         y = i;
9     }
10    else{
11        while(i > x)
12            i = i - 1;
13        x = y;
14        y = i;
15    }
16    return 0;
17 }
18
19 int main()
20 {

```

```
21     Exchange(4,9) = 0;
22 }
```

输出

```
1 Error Type 6 at line 21: Semantic Error, L-value required as left
  operand of assignment
2     Expected a L-value as left operand of assignment
```

说明：赋值号左边是一个不能为左值的类型（函数）

1.7 A-7

输入

```
1 struct Vector {
2     int x, y;
3 };
4
5 int main()
6 {
7     struct Vector A;
8     float b;
9     A.x = 12;
10    A.y = 13;
11    b = 2 * A;
12 }
```

输出

```
1 Error Type 7 at line 11: Semantic Error, Incompatible operands type
2     Invalid operands to binary * (have 'int' and 'float')
```

说明：乘号操作符两边类型不匹配，这里可以另外报错误类型 5（赋值号两边错误类型不匹配）。

1.8 A-8

输入

```

1 struct Vector {
2     float x, y;
3 };
4
5 int Multiplication(struct Vector A, struct Vector B)
6 {
7     float c = A.x * B.y + A.y * B.x;
8     return c;
9 }

```

输出

```

1 Error Type 8 at line 5: Semantic Error, Incompatible return type
2     Expected return type 'int'

```

说明：返回值实际类型与函数声明不一致，报在第 8 行也是对的。

1.9 A-9

输入

```

1 int Exchange(int x, int y)
2 {
3     int i = 0;
4     if(x >= 0) {
5         while(i < x)
6             i = i + 1;
7         x = y;
8         y = i;
9     }
10    else{
11        while(i > x)
12            i = i - 1;
13        x = y;
14        y = i;

```



```

15     }
16     return 0;
17 }
18
19 int main()
20 {
21     Exchange(1,2,3);
22 }

```

输出

```

1 Error Type 9 at line 21: Semantic Error, Invalid arguments
2     Incompatible arguments to function 'Exchange', expected type
      '(int, int)'

```

说明：函数实参与形参数目不一致

1.10 A-10

输入

```

1 struct Vector {
2     int x, y;
3 };
4
5 int main()
6 {
7     struct Vector v1, v2;
8     v1[1] = 1;
9     return 0;
10 }

```

输出

```

1 Error Type 10 at line 8: Semantic Error, Invalid array
2     It is NOT an array

```

说明：对非数组变量使用 [] 操作符，这里会连带报出错误类型 5，因为赋值号左边的类型可以算作是“未知”。

1.11 A-11

输入

```
1 int fetch(int m)
2 {
3     int a[6];
4     int i = 0;
5     int temp;
6     while (i < 6) {
7         a[i] = i + 1;
8     }
9     if (m == 0) return 0;
10    temp = m;
11    i = 0;
12    while (i < m) {
13        temp = m(i);
14        i = i + 1;
15    }
16    return temp;
17 }
18
19 int main()
20 {
21     int t = 0;
22     fetch(t);
23 }
```

输出

```
1 Error Type 11 at line 13: Semantic Error, Invalid function
2     'm' is NOT a function
```

说明：对非函数的标识符使用 () 操作符，同时会连带产生错误类型 8，因为函数返回值类型实际上是未知的。

1.12 A-12

输入

```
1 int main()
2 {
3     int a[10];
4     int i = 0;
5     int max = 0;
6     while (i < 10){
7         a[i] = i * i - i;
8         i = i + 1;
9     }
10    i = 0;
11    while (i < 10){
12        if (max < a[i]){
13            max = a[1.5];
14            i = i + 1;
15        }
16    }
17 }
```

输出

```
1 Error Type 12 at line 13: Semantic Error, Operands type mistaken in
   array
2     Array subscript is NOT an integer
```

说明：数组下标非整数，这里可以报出错误类型 5，因为赋值号变量右边类型可以认为是未知的。

1.13 A-13

输入

```

1 int add()
2 {
3     int a;
4     int b = 1;
5     int c = 2;
6     int d = b + c;
7     return d.c;
8 }

```

输出

```

1 Error Type 13 at line 7: Semantic Error, Illegal use of '.'

```

说明：对非结构体变量使用“.”操作符，同时可以报出错误类型 8。

1.14 A-14

输入

```

1 struct Vector {
2     float x, y;
3 };
4
5 int main()
6 {
7     struct Vector v;
8     float f;
9     f = v.x + v.y - v.z;
10    return 0;
11 }

```

输出

```

1 Error Type 14 at line 9: Semantic Error, Un-existed field
2     Struct has no member named 'z'

```

说明：使用了结构体中未定义的域 z，这里可以报出错误类型 5，因为赋值号变量右边类型可以认为是未知的。

1.15 A-15

输入

```
1 struct Human {
2     int age, weight;
3     float weight;
4 };
5
6 int main()
7 {
8     struct Human Tom;
9     Tom.age = 20;
10 }
```

输出

```
1 Error Type 15 at line 3: Semantic Error, Redefined variable or
   initialize variable in struct
2     Variable 'weight' is previously defined in the struct
```

说明：结构体内部有重复定义的域。有的同学由于 **Human** 定义错误，就没有将其放入符号表，因此会在第 8 行报 **Human** 未定义，这个不影响得分。

1.16 A-16

输入

```
1 struct Human {
2     int age, weight;
3 };
4
5 int main()
6 {
7     struct Human Lucious;
8     Lucious.age = 48;
9     Lucious.weight = 80;
```

```

10     return 0;
11 }
12
13 struct Human {
14     int age1;
15     float weight1;
16 };

```

输出

```

1 Error Type 16 at line 13: Semantic Error, Redefined struct
2     Name 'Human' used in the previous defined struct

```

说明：重复定义的结构体 Human。

1.17 A-17

输入

```

1 struct Male {
2     int age, weight;
3 };
4
5 int main()
6 {
7     struct Male Jason;
8     struct Female Becky;
9     return 0;
10 }

```

输出

```

1 Error Type 17 at line 8: Semantic Error, Undefined struct
2     Struct 'Female' is previously undefined

```

说明：使用了未定义的结构体 Female。

1.18 A-18

输入

```
1 struct Male {
2     int age = 18;
3     float weight;
4 };
5
6 int main()
7 {
8     struct Male Jeff;
9     Jeff.age = 25;
10    Jeff.weight = 70.5;
11 }
```

输出

```
1 Error Type 15 at line 2: Semantic Error, Redefined variable or
   initialize variable in struct
2     Cannot initialize the variable in struct
```

说明：在结构体 **Male** 中不能初始化变量。

2 B 组测试用例

本组测试用例共 1 个，其中包含多个语义错误。每一行的语义错误会分别算分，同一个语义错误可能会有连锁反应，其处理方式与 A 类用例相同，只要是合理的（包括但不限于下面明确给出的情况），都不会影响得分。

2.1 B-1

输入

```
1 struct HouseGuest {
2     int strength;
3     int EQ;
4 };
```

```

5
6 int HOHCompetition(struct HouseGuest h1, struct HouseGuest h2)
7 {
8     int HOH = 0;
9     if (h1.strength > h2.stregth)
10         HOH = 1;
11     if (h1.strength < h2.strength)
12         HOH = 2;
13     if (h1.strength == h2.strength) {
14         if (h1.EQ > h2.EQ)
15             HOH = 1;
16         if (h1.EQ < h2.EQ)
17             HOH = 2;
18         if (h1.EQ == h2.EQ)
19             HOH = 1;
20     }
21     return HOH;
22 }
23
24 struct HouseGuestA {
25     int strength1;
26     int EQ1;
27     int weight1 = 65;
28     float weight1;
29 };
30
31 struct HeadofHouse {
32     int strength2;
33     int EQ2;
34 };
35
36 int main()

```



```

37 {
38     struct HouseGuest Clay;
39     struct HouseGuest James;
40     int HeadofHouse;
41     int Head = 1;
42     int a[5], b[5];
43     float power = 100.0;
44     int i = 0;
45     while (i < 5) {
46         Clay.strength = a[i];
47         Clay.EQ = idiot;
48         James.strength = power;
49         James.EQ = b[i];
50         i = i + 1;
51     }
52     if (HOHCompetition(James, Clay) == 1)
53         Head = 1;
54     else if (HOHCompetition(James, Clay) = 2)
55         Head = 1;
56     else Head = 1;
57     return 1;
58 }

```

输出

```

1 Error Type 14 at line 9: Semantic Error, Un-existed field
2     Struct has no member named 'stregth'
3 Error Type 15 at line 27: Semantic Error, Redefined variable or
4     initialize variable in struct
5     Cannot initialize the variable in struct
6 Error Type 15 at line 28: Semantic Error, Redefined variable or
7     initialize variable in struct
8     Variable 'weight1' is previously defined in the struct
9 Error Type 3 at line 40: Semantic Error, Redefined variable

```

```

8      Variable name 'HeadofHouse' conflicts 'struct_HeadofHouse'
      previously defined
9 Error Type 1 at line 47: Semantic Error, Undefined variable
10     Variable 'idiot' is previously undefined
11 Error Type 5 at line 48: Semantic Error, Incompatible types when
      assigning
12     Expected type 'float'
13 Error Type 6 at line 54: Semantic Error, L-value required as left
      operand of assignment
14     Expected a L-value as left operand of assignment

```

说明：输出中的 7 个错误为本质错误，是必须要报出来的，这些错误可能会有连锁反应：第 9 行的错误可能会导致错误类型 7，因为 `stregth` 的类型未知；第 47 行的变量 `idiot` 没有定义，`idiot` 的类型可以看作未知，因此可能会报出一个类型 5 错误。

3 C 组测试用例

本组测试用例共 2 个，不包含语义错误，程序应该正常终止且没有任何错误提示。

3.1 C-1

输入

```

1 struct HouseGuest {
2     int strength;
3     int EQ;
4 };
5
6 int HOHCompetition(struct HouseGuest h1, struct HouseGuest h2)
7 {
8     int HOH = 0;
9     if (h1.strength > h2.strength)
10         HOH = 1;
11     if (h1.strength < h2.strength)
12         HOH = 2;

```

```

13     if (h1.strength == h2.strength) {
14         if (h1.EQ > h2.EQ)
15             HOH = 1;
16         if (h1.EQ < h2.EQ)
17             HOH = 2;
18         if (h1.EQ == h2.EQ)
19             HOH = 1;
20     }
21     return HOH;
22 }
23
24 struct HouseGuestA {
25     int strength1;
26     int EQ1;
27     int age;
28 };
29
30 int main()
31 {
32     struct HouseGuest Clay;
33     struct HouseGuest James;
34     int Head = 1;
35     int a[5], b[5];
36     int power = 100;
37     int idiot = 0;
38     int i = 0;
39     while (i < 5) {
40         Clay.strength = a[i];
41         Clay.EQ = idiot;
42         James.strength = power;
43         James.EQ = b[i];
44         i = i + 1;

```

```

45     }
46     if (HOHCompetition(James, Clay) == 1)
47         Head = 1;
48     else if (HOHCompetition(James, Clay) == 2)
49         Head = 1;
50     else Head = 1;
51     return 1;
52 }

```

输出

```

1 // 正常返回，无任何输出

```

说明：本测试用例是 B 类测试用例的改正版。

3.2 C-2

输入

```

1 struct Human {
2     int age;
3     int weight;
4 } p1;
5
6 struct {
7     int old;
8     int overweight;
9 } p2;
10
11 struct Male {int age1, weight1;} test1()
12 {
13     struct Male p3;
14     p3.age1 = 20;
15     p3.weight1 = 65;
16     return p3;
17 }

```

```

18
19 int test2(struct Female {int age3, weight3;} p4)
20 {
21     struct Female p5 = p4;
22     return p5.age3 + p5.age3;
23 }
24
25 float main()
26 {
27     int a[10], b[10];
28     int i = 0;
29     while (i < 10) {
30         struct {int age4, weight4;} p6;
31         p6.age4 = i + 70;
32         p6.weight4 = i * 5 + 18;
33         i = i + 1;
34     }
35     return 1.0;
36 }

```

输出

```
1 // 正常返回，无任何输出
```

说明：考察几类特殊的结构体定义方式。

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。需要能够识别其语言特性，如果提示错误则不得分；其他分组的同学需要识别出其中的错误，如果没有报错，则将视为违规，会倒扣分。

4.1 D-1

输入

```
1 struct Node {
```

```

2     int num;
3     int value;
4     int next;
5 };
6
7 int link(struct Node n1, struct Node n2);
8
9 int link(struct Node n1, struct Node n2)
10 {
11     if (n1.value >= n2.value) {
12         n1.next = n2.num;
13         return 0;
14     }
15     else {
16         n2.next = n1.num;
17         return 1;
18     }
19 }
20
21 int main()
22 {
23     int i = 0;
24     int s = 0;
25     struct Node a[10];
26     while (i < 10)
27     {
28         a[i].num = i;
29         a[i].value = i * i - 5 * i;
30         a[i].next = i + 1;
31         i = i + 1;
32     }
33     i = 0;

```

```

34
35     while (i < 10)
36     {
37         while (s < 10)
38         {
39             if (link(a[i],a[s]) == 1)
40             {
41                 a[i].value = a[s].value;
42                 a[i].next = i + 1;
43             }
44             s = s + 1;
45         }
46
47         s = 0;
48         i = i + 1;
49     }
50     return 0;
51 }

```

输出说明：对于 2.1 分组的同学，应该没有任何输出，对于其他分组的同学，应该在第 7 行报出有语法错误。

4.2 D-2

输入

```

1 struct Node {
2     int num;
3     int value;
4     int next;
5 };
6
7 int link(struct Node i, struct Node s)
8 {
9     if (i.value >= s.value) {

```

```

10         i.next = s.num;
11         return 0;
12     }
13     else {
14         s.next = i.num;
15         return 1;
16     }
17 }
18
19 int main()
20 {
21     int i = 0;
22     int s = 0;
23     struct Node a[10];
24     while (i < 10)
25     {
26         a[i].num = i;
27         a[i].value = i * i - 5 * i;
28         a[i].next = i + 1;
29         i = i + 1;
30     }
31     i = 0;
32
33     while (i < 10)
34     {
35         while (s < 10)
36         {
37             if (link(a[i],a[s]) == 1)
38             {
39                 a[i].value = a[s].value;
40                 a[i].next = i + 1;
41             }

```



```

42         s = s + 1;
43     }
44
45     s = 0;
46     i = i + 1;
47 }
48 return 0;
49 }

```

输出说明：2.2 分组的同学应该没有任何输出，其他分组的同学应该会识别出大量的重复定义变量（i 和 s）。

4.3 D-3

输入

```

1 struct Node {
2     int num;
3     int value;
4     int next;
5 };
6
7 struct Node2 {
8     int num2;
9     int value2;
10    int next2;
11 };
12
13 int link(struct Node n1, struct Node n2)
14 {
15     if (n1.value >= n2.value) {
16         n1.next = n2.num;
17         return 0;
18     }
19     else {

```

```

20         n2.next = n1.num;
21         return 1;
22     }
23 }
24
25 int main()
26 {
27     int i = 0;
28     int s = 0;
29     struct Node a[10];
30     struct Node2 b[10];
31     while (i < 10)
32     {
33         a[i].num = i;
34         a[i].value = i * i - 5 * i;
35         a[i].next = i + 1;
36         b[i].num2 = i;
37         b[i].value2 = i * i - 5 * i;
38         b[i].next2 = i + 1;
39         i = i + 1;
40     }
41     i = 0;
42
43     while (i < 10)
44     {
45         while (s < 10)
46         {
47             if (link(a[i],b[s]) == 1)
48             {
49                 a[i].value = a[s].value;
50                 a[i].next = i + 1;
51             }

```

```

52         s = s + 1;
53     }
54
55     s = 0;
56     i = i + 1;
57 }
58 return 0;
59 }

```

输出说明：2.3 分组应该没有任何输出，其他分组的同学应该在 47 行识别出类型不匹配（函数参数类型）

5 E 组测试用例

本组测试用例共 3 个，针对不同分组进行测试。下面给出的输出开始对应分组的同学的期望输出，其他分组同学的期望输出见说明。

5.1 E2.1

输入

```

1 struct Node {
2     int num;
3     int value;
4     int next;
5 };
6
7 struct Node1 {
8     int next1;
9 };
10
11 int link(struct Node N1, struct Node1 N2);
12
13 int insert(struct Node s1, struct Node s2);
14

```

```

15 int link(struct Node n1, struct Node n2)
16 {
17     if (n1.value >= n2.value) {
18         n1.next = n2.num;
19         return 0;
20     }
21     else {
22         n2.next = n1.num;
23         return 1;
24     }
25 }
26
27 int main()
28 {
29     int i = 0;
30     int s = 0;
31     struct Node a[10];
32     while (i < 10)
33     {
34         a[i].num = i;
35         a[i].value = i * i - 5 * i;
36         a[i].next = i + 1;
37         i = i + 1;
38     }
39     i = 0;
40
41     while (i < 10)
42     {
43         while (s < 10)
44         {
45             if (link(a[i],a[s]) == 1)
46                 {

```

```

47         a[i].value = a[s].value;
48         a[i].next = i + 1;
49     }
50     s = s + 1;
51 }
52
53     s = 0;
54     i = i + 1;
55 }
56 return 0;
57 }

```

输出

```

1 Error Type 18 at line 11: Semantic Error, Function declared but
   undefined
2     Function 'link' is declared but undefined
3 Error Type 18 at line 13: Semantic Error, Function declared but
   undefined
4     Function 'insert' is declared but undefined
5 Error Type 19 at line 15: Semantic Error, Function inconsistent
   between declaration and definition
6     Conflicting type for function 'link'

```

说明：2.1 分组同学需要输出上述的错误信息，其中第 11 行的错误类型 18 可以不输出，因为其本质错误还是函数声明不一致。其他分组的同学应该识别出有语法错误。

5.2 E2.2

输入

```

1 struct Node {
2     int num;
3     int value;
4     int next;
5 };

```

```

6
7 int link(struct Node i, struct Node s)
8 {
9     if (i.value >= s.value) {
10         i.next = s.num;
11         return 0;
12     }
13     else {
14         s.next = i.num;
15         return 1;
16     }
17 }
18
19 int main()
20 {
21     int i = 0;
22     int s = 0;
23     struct Node s;
24     struct Node a[10];
25     while (i < 10)
26     {
27         a[i].num = i;
28         a[i].value = i * i - 5 * i;
29         a[i].next = i + 1;
30         i = i + 1;
31     }
32     i = 0;
33
34     while (i < 10)
35     {
36         while (s < 10)
37         {

```

```

38         if (link(a[i],a[s]) == 1)
39         {
40             a[i].value = a[s].value;
41             a[i].next = i + 1;
42         }
43         s = s + 1;
44     }
45
46     s = 0;
47     i = i + 1;
48 }
49 return 0;
50 }

```

输出

```

1 Error Type 3 at line 23: Semantic Error, Redefined variable
2     Variable 's' is previously defined / Variable name 's'
      conflicts function 's' previously defined

```

说明：2.2 分组同学应该只识别出一个类型重复定义（这个错误可能会导致其他行产生其他的语义错误）；其他分组的同学应该识别出大量的重复定义变量（i、s）。

5.3 E2.3

输入

```

1 struct Male {
2     int age;
3     float circumference[3];
4 };
5
6 struct Female {
7     int a;
8     float c[3];
9 };

```

```

10
11 struct Transgender {
12     int ori;
13     int new;
14 };
15
16 int test1(struct Male James)
17 {
18     return 0;
19 }
20
21 struct Male test2(struct Female Meg)
22 {
23     struct Female Venesa;
24     Venesa.a = Meg.a;
25     return Venesa;
26 }
27
28 struct Male test3(struct Transgender Audrey)
29 {
30     struct Female Liz;
31     Liz.a = Audrey.ori;
32     return Liz;
33 }
34
35 int main()
36 {
37     struct Male Jace;
38     struct Female Julia;
39     struct Transgender Clay;
40
41     test1(Julia);

```



```
42     test1(Clay);  
43     test2(Jace);  
44     test2(Clay);  
45     test3(Clay);  
46     return 0;  
47 }
```

输出

```
1 Error Type 5 at line 42: Semantic Error, Incompatible types when  
   assigning  
2 Error Type 5 at line 44: Semantic Error, Incompatible types when  
   assigning
```

说明：2.3 分组的同学应该识别出上述的两组类型不匹配（或者函数参数类型不匹配），其他分组的同学应该识别出四组（41、42、43 和 44 行）。

6 结束语

如果对本测试用例有任何疑议，可以写邮件与杨文华助教联系，注意同时抄送给许老师。