编译原理第二次实验测试用例：目录

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

```
int add(int a, int b)
{
    int x, y;
    x = a;
    y = b;
    z = x + y;
}
```

输出

```
Error Type 1 at line 6: Semantic Error, Undefined variable
        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

```
int print(int a)
{
    int x = 0;
    int y = a + x;
    printf(y);
}
```

输出

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

说明：printf 未定义。

1.3 A-3

输入

```
struct a {
    int x;
    int y;
};

int main()
{
    int a = 3;
}
```

输出

```
Error Type 3 at line 8: Semantic Error, Redefined variable
        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

```
int multiply(int a, int b)
{
    int c = a * b;
    return c;
}

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

```
Error Type 4 at line 14: Semantic Error, Redefined function
        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

```
struct Male {
    int age;
    int weight;
};

struct Female {
    int circumference[3];
    float w;
};

int main()
{
    struct Male a;
    struct Female b;
    a = b;
}
```

输出

```
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

```
int Exchange(int x, int y)
{
    int i = 0;
    if(x >= 0){
        while(i < x)
            i = i + 1;
        x = y;
        y = i;
    }
    else{
        while(i > x)
            i = i - 1;
        x = y;
        y = i;
    }
    return 0;
}

int main()
{
    Exchange(4,9) = 0;
}
```

输出

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

```
struct Vector {
    int x, y;
};

int main()
{
    struct Vector A;
    float b;
    A.x = 12;
    A.y = 13;
    b = 2 * A;
}
```

输出

```
Error Type 7 at line 11: Semantic Error, Incompatible operands type
        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

```
Error Type 8 at line 5: Semantic Error, Incompatible return type
        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

```
int Exchange(int x, int y)
{
    int i = 0;
    if(x >= 0){
        while(i < x)
            i = i + 1;
        x = y;
        y = i;
    }
    else{
        while(i > x)
            i = i - 1;
        x = y;
        y = i;
    }
    return 0;
}

int main()
{
    Exchange(1,2,3);
}
```

输出

```
Error Type 9 at line 21: Semantic Error, Invalid arguments
        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

```
struct Vector {
    int x, y;
};

int main()
{
    struct Vector v1, v2;
    v1[1] = 1;
    return 0;
}
```

输出

```
Error Type 10 at line 8: Semantic Error, Invalid array
        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

```
int fetch(int m)
{
    int a[6];
    int i = 0;
    int temp;
    while (i < 6) {
        a[i] = i + 1;
    }
    if (m == 0) return 0;
    temp = m;
    i = 0;
    while (i < m){
        temp = m(i);
        i = i + 1;
    }
    return temp;
}

int main()
{
    int t = 0;
    fetch(t);
}
```

输出

```
Error Type 11 at line 13: Semantic Error, Invalid function
        '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

```
int main()
{
    int a[10];
    int i = 0;
    int max = 0;
    while (i < 10){
        a[i] = i * i - i;
        i = i + 1;
    }
    i = 0;
    while (i < 10){
        if (max < a[i]){
            max = a[1.5];
            i = i + 1;
        }
    }
}
```

输出

```
Error Type 12 at line 13: Semantic Error, Operands type mistaken in
   array
        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

```c
int add()
{
    int a;
    int b = 1;
    int c = 2;
    int d = b + c;
    return d.c;
}
```

输出

```
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

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

int main()
{
    struct Vector v;
    float f;
    f = v.x + v.y - v.z;
    return 0;
}
```

输出

```
Error Type 14 at line 9: Semantic Error, Un-existed field
        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

```c
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

```
Error Type 15 at line 3: Semantic Error, Redefined variable or
    initialize variable in struct
        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

```c
struct Human {
    int age, weight;
};

int main()
{
    struct Human Lucious;
    Lucious.age = 48;
    Lucious.weight = 80;
    return 0;
}

struct Human {
    int age1;
    float weight1;
};
```

输出

```
Error Type 16 at line 13: Semantic Error, Redefined struct
        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

```
struct Male {
    int age, weight;
};

int main()
{
    struct Male Jason;
    struct Female Becky;
    return 0;
}
```

输出

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

说明：使用了未定义的结构体 Female。

1.18 A-18

输入

```
struct Male {
    int age = 18;
    float weight;
};

int main()
{
    struct Male Jeff;
    Jeff.age = 25;
    Jeff.weight = 70.5;
}
```

输出

```
Error Type 15 at line 2: Semantic Error, Redefined variable or
   initialize variable in struct
        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

```
struct HouseGuest {
    int strength;
    int EQ;
};
```

```c

int HOHCompetition(struct HouseGuest h1, struct HouseGuest h2)
{
    int HOH = 0;
    if (h1.strength > h2.stregth)
        HOH = 1;
    if (h1.strength < h2.strength)
        HOH = 2;
    if (h1.strength == h2.strength){
        if (h1.EQ > h2.EQ)
            HOH = 1;
        if (h1.EQ < h2.EQ)
            HOH = 2;
        if (h1.EQ == h2.EQ)
            HOH = 1;
    }
    return HOH;
}

struct HouseGuestA {
    int strength1;
    int EQ1;
    int weight1 = 65;
    float weight1;
};

struct HeadofHouse {
    int strength2;
    int EQ2;
};

int main()
```

```
{
    struct HouseGuest Clay;
    struct HouseGuest James;
    int HeadofHouse;
    int Head = 1;
    int a[5], b[5];
    float power = 100.0;
    int i = 0;
    while (i < 5) {
        Clay.strength = a[i];
        Clay.EQ = idiot;
        James.strength = power;
        James.EQ = b[i];
        i = i + 1;
    }
    if (HOHCompetition(James, Clay) == 1)
        Head = 1;
    else if (HOHCompetition(James, Clay) = 2)
        Head = 1;
    else Head = 1;
    return 1;
}
```

输出

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

```
        Variable name 'HeadofHouse' conflicts 'struct␣HeadofHouse'
           previously defined
Error Type 1 at line 47: Semantic Error, Undefined variable
        Variable 'idiot' is previously undefined
Error Type 5 at line 48: Semantic Error, Incompatible types when
   assigning
        Expected type 'float'
Error Type 6 at line 54: Semantic Error, L-value required as left
   operand of assignment
        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

```c
struct HouseGuest {
    int strength;
    int EQ;
};

int HOHCompetition(struct HouseGuest h1, struct HouseGuest h2)
{
    int HOH = 0;
    if (h1.strength > h2.strength)
        HOH = 1;
    if (h1.strength < h2.strength)
        HOH = 2;
    if (h1.strength == h2.strength){
        if (h1.EQ > h2.EQ)
            HOH = 1;
        if (h1.EQ < h2.EQ)
            HOH = 2;
        if (h1.EQ == h2.EQ)
            HOH = 1;
    }
    return HOH;
}

struct HouseGuestA {
    int strength1;
    int EQ1;
    int age;
};

int main()
{
    struct HouseGuest Clay;
    struct HouseGuest James;
    int Head = 1;
    int a[5], b[5];
    int power = 100;
    int idiot = 0;
    int i = 0;
    while (i < 5) {
        Clay.strength = a[i];
        Clay.EQ = idiot;
        James.strength = power;
        James.EQ = b[i];
        i = i + 1;
    }
    if (HOHCompetition(James, Clay) == 1)
        Head = 1;
    else if (HOHCompetition(James, Clay) == 2)
        Head = 1;
    else Head = 1;
    return 1;
}
```

输出

```
//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

```
struct Human {
    int age;
    int weight;
} p1;

struct {
    int old;
    int overweight;
} p2;

struct Male {int age1, weight1;} test1()
{
    struct Male p3;
    p3.age1 = 20;
    p3.weight1 = 65;
    return p3;
}
```

```

int test2(struct Female {int age3, weight3;} p4)
{
    struct Female p5 = p4;
    return p5.age3 + p5.age3;
}

float main()
{
    int a[10], b[10];
    int i = 0;
    while (i < 10) {
        struct {int age4, weight4;} p6;
        p6.age4 = i + 70;
        p6.weight4 = i * 5 + 18;
        i = i + 1;
    }
    return 1.0;
}
```

输出

```
//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

```
struct Node {
    int num;
    int value;
    int next;
};

int link(struct Node n1, struct Node n2);

int link(struct Node n1, struct Node n2)
{
    if (n1.value >= n2.value) {
        n1.next = n2.num;
        return 0;
    }
    else {
        n2.next = n1.num;
        return 1;
    }
}

int main()
{
    int i = 0;
    int s = 0;
    struct Node a[10];
    while (i < 10)
    {
        a[i].num = i;
        a[i].value = i * i - 5 * i;
        a[i].next = i + 1;
        i = i + 1;
    }
    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

```
struct Node {
    int num;
    int value;
    int next;
};

int link(struct Node i, struct Node s)
{
    if (i.value >= s.value) {
        i.next = s.num;
        return 0;
    }
    else {
        s.next = i.num;
        return 1;
    }
}

int main()
{
    int i = 0;
    int s = 0;
    struct Node a[10];
    while (i < 10)
    {
        a[i].num = i;
        a[i].value = i * i - 5 * i;
        a[i].next = i + 1;
        i = i + 1;
    }
    i = 0;

    while (i < 10)
    {
        while (s < 10)
        {
            if (link(a[i],a[s]) == 1)
            {
                a[i].value = a[s].value;
                a[i].next = i + 1;
            }
```

```
                s = s + 1;
            }

            s = 0;
            i = i + 1;
        }
        return 0;
}
```

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

```
struct Node {
    int num;
    int value;
    int next;
};

struct Node2 {
    int num2;
    int value2;
    int next2;
};

int link(struct Node n1, struct Node n2)
{
    if (n1.value >= n2.value) {
        n1.next = n2.num;
        return 0;
    }
    else {
        n2.next = n1.num;
        return 1;
    }
}

int main()
{
    int i = 0;
    int s = 0;
    struct Node a[10];
    struct Node2 b[10];
    while (i < 10)
    {
        a[i].num = i;
        a[i].value = i * i - 5 * i;
        a[i].next = i + 1;
        b[i].num2 = i;
        b[i].value2 = i * i - 5 * i;
        b[i].next2 = i + 1;
        i = i + 1;
    }
    i = 0;

    while (i < 10)
    {
        while (s < 10)
        {
            if (link(a[i],b[s]) == 1)
            {
                a[i].value = a[s].value;
                a[i].next = i + 1;
            }
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

```
struct Node {
    int num;
    int value;
    int next;
};

struct Node1 {
    int next1;
};

int link(struct Node N1, struct Node1 N2);

int insert(struct Node s1, struct Node s2);

```

```c
int link(struct Node n1, struct Node n2)
{
    if (n1.value >= n2.value) {
        n1.next = n2.num;
        return 0;
    }
    else {
        n2.next = n1.num;
        return 1;
    }
}

int main()
{
    int i = 0;
    int s = 0;
    struct Node a[10];
    while (i < 10)
    {
        a[i].num = i;
        a[i].value = i * i - 5 * i;
        a[i].next = i + 1;
        i = i + 1;
    }
    i = 0;

    while (i < 10)
    {
        while (s < 10)
        {
            if (link(a[i],a[s]) == 1)
            {
                a[i].value = a[s].value;
                a[i].next = i + 1;
            }
            s = s + 1;
        }

        s = 0;
        i = i + 1;
    }
    return 0;
}
```

输出

```
Error Type 18 at line 11: Semantic Error, Function declared but undefined
        Function 'link' is declared but undefined
Error Type 18 at line 13: Semantic Error, Function declared but undefined
        Function 'insert' is declared but undefined
Error Type 19 at line 15: Semantic Error, Function inconsistent between declaration and definition
        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

```c
struct Node {
    int num;
    int value;
    int next;
};
```

```c

int link(struct Node i, struct Node s)
{
    if (i.value >= s.value) {
        i.next = s.num;
        return 0;
    }
    else {
        s.next = i.num;
        return 1;
    }
}

int main()
{
    int i = 0;
    int s = 0;
    struct Node s;
    struct Node a[10];
    while (i < 10)
    {
        a[i].num = i;
        a[i].value = i * i - 5 * i;
        a[i].next = i + 1;
        i = i + 1;
    }
    i = 0;

    while (i < 10)
    {
        while (s < 10)
        {
            if (link(a[i],a[s]) == 1)
            {
                a[i].value = a[s].value;
                a[i].next = i + 1;
            }
            s = s + 1;
        }

        s = 0;
        i = i + 1;
    }
    return 0;
}
```

输出

```
Error Type 3 at line 23: Semantic Error, Redefined variable
        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

```
struct Male {
    int age;
    float circumference[3];
};

struct Female {
    int a;
    float c[3];
};
```

```c

struct Transgender {
    int ori;
    int new;
};

int test1(struct Male James)
{
    return 0;
}

struct Male test2(struct Female Meg)
{
    struct Female Venesa;
    Venesa.a = Meg.a;
    return Venesa;
}

struct Male test3(struct Transgender Audrey)
{
    struct Female Liz;
    Liz.a = Audrey.ori;
    return Liz;
}

int main()
{
    struct Male Jace;
    struct Female Julia;
    struct Transgender Clay;

    test1(Julia);
    test1(Clay);
    test2(Jace);
    test2(Clay);
    test3(Clay);
    return 0;
}
```

输出

```
Error Type 5 at line 42: Semantic Error, Incompatible types when
   assigning
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。