编译原理第一次实验测试用例：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 6

1.8 A-8 7

1.9 A-9 8

1.10 A-10 9

1.11 A-11 9

1.12 A-12 10

1.13 A-13 11

1.14 A-14 12

1.15 A-15 12

1.16 A-16 13

1.17 A-17 14

2 B 组测试用例 **14**

3 C 组测试用例 **16**

3.1 C-1 17

3.2 C-2 18

4 D 组测试用例 **19**

4.1 D-1 19

4.2 D-2 20

4.3 D-3 21

5 E 组测试用例 **22**

5.1 E2.1 23

5.2 E2.2 24
5.3 E2.3 25

6 结束语 **27**

1　A 组测试用例

本组测试用例共 17 个，分别对应语义错误 1-17，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 main()
{

    int a, b;
    c = a;
}
```

输出

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

说明：c=a 这一句包含未定义的变量 c，这里也可以另外报出错误类型 5（赋值号两边类型不匹配）。

1.2　A-2

输入

```

int main()
{
    int a, b;

    readInt(a);
}
```

输出

```
Error Type 2 at line 6: Semantic Error, Undefined function
        Function 'readInt' is previously undefined
```

说明：readInt 未定义。

1.3 A-3

输入

```
int main()
{
    int a, b;
    int a, c;
}
```

输出

```
Error Type 3 at line 4: Semantic Error, Redefined variable
        Variable 'a' is previously defined / Variable name 'a'
           conflicts function 'a' previously defined
```

说明：重复定义的变量 a，这里如果错误位置写为第 3 行，或者 3-4 行，都算对。

1.4 A-4

输入

```
int duplicated_function(int x)
{
    if(x>=0)     return x;
    else     return -x;
}

int duplicated_function(int x2)
{
}

```

```
int main()
{
    int a = 0;
    return duplicated_function(a);
}
```

输出

```
Error Type 4 at line 7: Semantic Error, Redefined function
        Function 'duplicated_function' is previously defined
```

说明：重复定义的函数 duplicated_function。这里如果没有把重复定义的函数放入符号表，会在第 14 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 A
{
    int x, y;
};

struct B
{
    float a, b;
};

int main()
{
    struct A Aa;
    struct B Bb;
    Aa = Bb;
}
```

输出

```
Error Type 5 at line 15: Semantic Error, Incompatible types when
   assigning
        Expected type '{int,int}'
```

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

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

float dotmultiply(struct Vector v1, struct Vector v2)
{
    return v1.x*v2.y - v1.y*v2.x;
}

int main()
{
    struct Vector sv1, sv2;
    dotmultiply(sv1, sv2) = 1.434;
}
```

输出

```
Error Type 6 at line 14: 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
{
    float x, y;
};

int main()
{
    struct Vector v1, v2;
    int a, b;
    b = a + v1;
}
```

输出

```
Error Type 7 at line 10: Semantic Error, Incompatible operands type
        Invalid operands to binary + (have 'int' and '{float,float}')
```

说明：加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），因为 a+v1 的类型未知。

1.8 A-8

输入

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

int dotmultiply(struct Vector v1, struct Vector v2)
{
    return v1.x*v2.y - v1.y*v2.x;
}

int main()
```

```
{
    return 0;
}
```

输出

```
Error Type 8 at line 8: 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

1.9 A-9

输入

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

float dotmultiply(struct Vector v1, struct Vector v2)
{
    return v1.x*v2.y - v1.y*v2.x;
}

int main()
{
    struct Vector sv1, sv2;
    int a = 5;
    dotmultiply(sv1, a);
}
```

输出

```
Error Type 9 at line 15: Semantic Error, Invalid arguments
        Incompatible arguments to function 'dotmultiply', expected
           type '({float,float},{float,float})'
```

说明：函数实参与形参类型不一致

1.10 A-10

输入

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

int func(float px)
{
    return 0;
}
int main()
{
    struct Vector v1, v2;
    func(v1[0]);
}
```

输出

```
Error Type 10 at line 13: 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 9，因为实参的类型可以算作是“未知”。

1.11 A-11

输入

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

```

```
float dotmultiply(struct Vector v1, struct Vector v2)
{
    return v1(v2);
}


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

输出

```
Error Type 11 at line 8: Semantic Error, Invalid function
        'v1' 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

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


int main()
{
    struct Vector v1, v2;
    int a[10];
    int i = 0, j;
    while(i<10) {
        j = a[v1.x];
        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

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


int main()
{
    struct Vector v1, v2;
    float a, b;
    b = a.x;
}
```

输出

```
Error Type 13 at line 11: Semantic Error, Illegal use of '.'
        Request for member 'x' in something NOT a structure
```

说明：对非结构体变量使用“.”操作符，同时可以报出错误类型 5，原因同上。

1.14 A-14

输入

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


float function()
{
    struct Vector v1, v2;
    float a = 0.0, b;
    b = v1.z;
    return a;
}
```

输出

```
Error Type 14 at line 11: Semantic Error, Un-existed field
        Struct has no member named 'z'
```

说明：试用了结构体中为定义的域 z，这里可以报出错误类型 5，原因同上。

1.15 A-15

输入

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


int main()
```

```
{
    struct Vector v1;
    return 0;
}
```

输出

```
Error Type 15 at line 4: Semantic Error, Redefined variable or
   initialize variable in struct
        Variable 'y' 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 Vector 定义错误，就没有将其放入符号表，因此会在第 10 行报 Vector 未定义，这个不影响得分。

1.16 A-16

输入

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

struct Vector
{
    float z;
};

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

输出

```
Error Type 16 at line 6: Semantic Error, Redefined struct
        Name 'Vector' used in the previous defined struct
```

说明：重复定义的结构体 Vector。

1.17 A-17

输入

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


int main()
{
    struct Vector v1, v2;
    struct Vector1 v3;

    return 0;
}
```

输出

```
Error Type 17 at line 10: Semantic Error, Undefined struct
        Struct 'Vector1' is previously undefined
```

说明：使用了未定义的结构体 Vector1。

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。 输入

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

```c


float dotMultiply(struct Vector v1, struct Vector v2)
{
    return v1.x*v2.y-v1.y*v2.z;
}

struct Vector2
{
    float x2 = 1.0, y2;
    float y2;
};

struct Vector3
{
    int x3;
};

int main()
{
    float a[10], c[10];

    int Vector3;
    int d[10];
    int i = 0;
    struct Vector sv1;
    struct Vector sv2;
    while(i<10)
    {
        sv1.x = a[i];
        sv1.y = b;
        sv2.x = c[i];
        sv2.y = d[i];
        i = i+1;

        dotMultiply(sv1, sv2);
    }
    return 0;
}
```

输出

```
Error Type 14 at line 9: Semantic Error, Un-existed field
        Struct has no member named 'z'
Error Type 15 at line 14: Semantic Error, Redefined variable or initialize variable in struct
        Cannot initialize the variable in struct
Error Type 15 at line 15: Semantic Error, Redefined variable or initialize variable in struct
        Variable 'y2' is previously defined in the struct
Error Type 3 at line 27: Semantic Error, Redefined variable
        Variable name 'Vector3' conflicts 'struct␣Vector3' previously defined
Error Type 1 at line 35: Semantic Error, Undefined variable
        Variable 'b' is previously undefined
Error Type 5 at line 37: Semantic Error, Incompatible types when assigning
        Expected type 'float'
```

说明：输出中的 6 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 和 8，因为 z 的类型未知；读 35 行的变量 b 没有定义，b 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

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


float dotMultiply(struct Vector v1, struct Vector v2)
{
    float ret = v1.x*v2.y-v1.y*v2.x;
    return ret;
}

struct Vector2
{
    float x1, y1;
};

int main()
{
    float a[10], b, c[10];
    float d[10];
    int i = 0;
    struct Vector sv1;
    struct Vector sv2;
    while(i<10)
    {
        sv1.x = a[i];
        sv1.y = b;
        sv2.x = c[i];
        sv2.y = d[i];
        i = i+1;

        dotMultiply(sv1, sv2);
    }
    return 0;
}
```

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

```
float func1()
{
    struct {float a, b;} vv;
    vv.a = 1.0;
    vv.b = 1.0;
    return vv.a;
}

int func2(struct Vector1 {int y;} v1)
{
    struct Vector1 v11 = v1;
    return v1.y;
}


struct Vector2 {int x;} main()
{
    struct Vector2 v2;
    v2.x = 1;
    return v2;
}
```

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 Vector
{
    float x, y;
};


float dotMultiply(struct Vector vv1, struct Vector vv2);

float dotMultiply(struct Vector v1, struct Vector v2)
{
    return v1.x*v2.y-v1.y*v2.x;
}

int main()
{
    float a[10], c[10], b[10], d[10];
    int i = 0;
    struct Vector sv1;
    struct Vector sv2;
    while(i<10)
    {
        sv1.x = a[i];
        sv1.y = b[i];
        sv2.x = c[i];
        sv2.y = d[i];
        i = i+1;

        dotMultiply(sv1, sv2);
    }
    return 0;
}
```

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 Vector
{
    float x, y;
};


float dotMultiply(struct Vector v1, struct Vector v2)
{
    float d = v1.x*v2.y-v1.y;
    return d;
}

int main()
{
    float a[10], c[10], b[10], d[10];
    int i = 0;
    struct Vector v1;
    struct Vector v2;
    while(i<10)
    {
        v1.x = a[i];
        v1.y = b[i];
        v2.x = c[i];
        v2.y = d[i];
        i = i+1;

        dotMultiply(v1, v2);
    }
    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（v1、v2 和 d），同时也可以报出对 d 使用 [] 操作符，因为程序将 d 记录为 float 类型变量。

4.3 D-3

输入

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

struct Vector2
{
    float fa, fb;
};

```

```
11 float dotMultiply(struct Vector v1, struct Vector v2)
12 {
13     return v1.x*v2.y-v1.y*v2.x;
14 }
15 
16 int main()
17 {
18     float a[10], c[10], b[10], d[10];
19     int i = 0;
20     struct Vector sv1;
21     struct Vector2 sv2;
22     while(i<10)
23     {
24         sv1.x = a[i];
25         sv1.y = b[i];
26         sv2.fa = c[i];
27         sv2.fb = d[i];
28         i = i+1;
29 
30         sv2 = sv1;
31     }
32     dotMultiply(sv1, sv2);
33     return 0;
34 }
```

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 30 和 32 行识别出类型不匹配（分别是赋值号两边和函数参数类型）

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

这组测试用例针对 1.1 分组的同学

输入

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


float dotMultiply(struct Vector pv1 );

float getX();

float dotMultiply(struct Vector pdv1, struct Vector pdv2)
{
    return pdv1.x*pdv2.y-pdv1.y*pdv2.x;
}

int main()
{
    float a[10], c[10], b[10], d[10];
    int i = 0;
    struct Vector v1;
    struct Vector v2;
    while(i<10)
    {
        v1.x = a[i];
        v1.y = b[i];
        v2.x = c[i];
        v2.y = d[i];
        i = i+1;
    }
    return 0;
}
```

输出

```
Error Type 19 at line 11: Semantic Error, Function inconsistent
   between declaration and defination
        Conflicting type for function 'dotMultiply'
Error Type 18 at line 9: Semantic Error, Function declared but
   undefined
        Function 'getX' is declared but undefined
Error Type 18 at line 7: Semantic Error, Function declared but
   undefined
        Function 'dotMultiply' is declared but undefined
```

说明：2.1 分组同学需要输出上述的错误信息，其中第 7 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

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


float dotMultiply(struct Vector v1, struct Vector v2)
{
    int a = 0, b = 1, c = 2;
    float d = v1.x*v2.y-v1.y;
    return d;
}

int main()
```

```c
15 {
16     float a[10], c[10], b[10], d[10];
17     int i = 0;
18     struct Vector v1;
19     struct Vector v1, v2;
20     while(i<10)
21     {
22         v1.x = a[i];
23         v1.y = b[i];
24         v2.x = c[i];
25         v2.y = d[i];
26 
27         i = i + 1;
28         dotMultiply(v1, v2);
29     }
30     return 0;
31 }
```

输出

```
Error Type 3 at line 19: Semantic Error, Redefined variable
        Variable 'v1' is previously defined / Variable name 'v1'
           conflicts function 'v1' previously defined
```

说明：2.2 分组同学应该只识别出一个类型重复定义（这个错误会导致 22-23 行产生其他的语义错误，例如认为 v1 未定义）；其他分组的同学应该识别出大量的重复定义变量（a、b、c、d、v1、v2）。

5.3 E2.3

输入

```c
struct Vector1
{
    float x1, y1;
    int a1, b1;
};

struct Vector2
{
    float x2, y2;
    int a2, b2;
};

struct Vector3
{
    float x3, y3;
    float a3;
};

int main()
{
    struct Vector1 v1;
    struct Vector2 v2;
    struct Vector3 v3;

    v1 = v2;
    v2 = v3;
    v1 = v3;
}
```

输出

```
Error Type 5 at line 26: Semantic Error, Incompatible types when
   assigning
        Expected type '{float,float,int,int}'
Error Type 5 at line 27: Semantic Error, Incompatible types when
   assigning
        Expected type '{float,float,int,int}'
```

说明：2.3 分组的同学应该识别出上述的两组类型不匹配，其他分组的同学应该识别出三组（25、26、27 行）。

6 结束语

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