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 20 个，测试用例 1-17 分别对应语义错误 1-17，之后三个测试用例对应于语义错误 15,9,16。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”的错误类型是必须报出来的，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。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 Vector {
    int x;
    int y;
};
int setVector(int x1, int y1, int z1)
{
    struct Vector a, b;
    a.x = x1;
    a.y = y1;
    z = z1;
}
```

输出

```
Error type 1 at line 10: Undefined variable "z"
```

说明：z = z1 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（= 两边类型不匹配）。

1.2 A-2

输入

```
int sort()
{
    int a[10];
    int i = 0;
```

```
    while(i<10){
        int j = 0;
        while(j<10){
            if(a[j]<=a[j+1])
                a[j] = 0;
        else
            swap(a,j,j+1);
        }
    }
    return 0;
}
```

输出

```
Error type 2 at line 11: Undefined function "swap"
```

说明：swap 函数未定义。

1.3 A-3

输入

```
struct name {
    int z;
}name1;

struct Vector {
    int x;
    int y;
};

int main()
{
    struct Vector vector;
    int name = vector.x;

```

```
    vector.x = 2;
    name1.z = 5;

    vector.y = name1.z;

    if(vector.y==5){
        vector.y = name1.z;
    }


    return 0;
}
```

输出

```
Error type 3 at line 13: Redefined variable "name"
```

说明：重复定义的变量 name，这里如果错误位置写为第 1 行也算对。

1.4 A-4

输入

```
int math_function(int a, int b, int c){
    int result = a + b + c;
    return result;
}

int math_function(int a1, int b1){
    int result1 = a1 + b1 * a1;
    return result1;
}

int main()
{
    return math_function(5,4,7);
```

```
}
```

输出

```
Error type 4 at line 6: Redefined function "math_function"
```

说明：重复定义的函数 math_function。这里如果没有把重复定义的函数放入符号表，会在第 13 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 Food {
    int type;
    int weight;
    float price;
};

struct Purchase{
    struct Food food;
    int time;
    int sum;
};

int main(){
    struct Food a;
    struct Purchase b;
    b.food = a;
    b.time = 13;
    b.sum = a;
}
```

输出

```
Error type 5 at line 18: Type mismatched
```

说明：赋值号两边类型不匹配（结构体赋值给整型）。

1.6 A-6

输入

```
int returnSmallerOne(int x, int y)
{
    if(x >= y)
        return y;
    else
        return x;
}

int main()
{
    int min = 14;
    int a = 12, b = 15;
    if( returnSmallerOne(a, b) < min)
        min = returnSmallerOne(a, b);
    else
        returnSmallerOne(a, b) = min;
}
```

输出

```
Error type 6 at line 16: The left-hand side of an assignment must be
   a variable
```

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;
    float array[5];
};
```

```
5
6  int main()
7  {
8      struct Vector A;
9      float array1[10],b;
10     A.x = 12;
11     A.y = 13;
12
13     b = A.array[2];
14     array1[1] = A.array[2] * 2;
15 }
```

输出

```
1  Error type 7 at line 14: Operands type mismatched
```

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），必须在 14 行。

1.8 A-8

输入

```
1  struct Vector{
2      float x;
3      float y;
4      float z;
5  };
6  int structMutipleFunction(struct Vector A, struct Vector B)
7  {
8      float c = A.x * B.x + A.y * B.y + A.z * B.z;
9      return c;
10 }
11
12 float structAddFunction(struct Vector A1)
13 {
```

```
14      float c1 = A1.x + A1.y + A1.z;
15      return c1;
16  }
```

输出

```
1  Error type 8 at line 9: The return type mismatched
```

说明：返回值实际类型与函数定义不一致，报在第 6 行也是对的。

1.9 A-9

输入

```
1  int split(int first, int last)
2  {
3      int x[10];
4      int pivot = first;
5      int split_point = first;
6      int i = first + 1;
7      int temp,temp2;
8      while(i <= last)
9      {
10         if(x[i] < x[pivot])
11         {
12             split_point = split_point + 1;
13             temp = x[i];
14             x[i] = x[split_point];
15             x[split_point] = temp;
16         }
17         i = i + 1;
18     }
19     temp2 = x[pivot];
20     x[pivot] = x[split_point];
21     x[split_point] = temp2;
22     return split_point;
```

```
}

int main()
{
    split(1,9,10);
}
```

输出

```
Error type 9 at line 27: The method "split" is not applicable for the
    arguments
```

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;
    float array[5];
};

int main()
{
    struct Vector v1, v2;
    v1.array[1] = 1.0;
    v2.x[1] = 2;
    return 0;
}
```

输出

```
Error type 10 at line 10: Illegal use of "[]"
```

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 multiple_function(int len)
{
    int array[10];
    int result = 1;
    int i = 0;
    while(i < 10){
        result = result * array[i];
        i = i + 1;
    }
    return result;
}

int main()
{
    int t = multiple_function(5);
    t(5);
}
```

输出

```
Error type 11 at line 16: "t" must be a function
```

说明：对非函数的标识符使用 () 操作符。

1.12 A-12

输入

```
int bubblesort(int n)
{
    int exchange;
    int i, j, temp;
    int p[50];
```

```
    while(i < n)
    {
        i = i + 1;
        exchange = 1;

        while(j < n)
        {
            j = j + 1;
            if(p[j] > p[j+1])
            {
                temp = p[j+1];
                p[j+1] = p[j];
                p[0.5] = temp;
                exchange = 0;
            }
        }
        if(exchange == 0)
            p[i] = -1;
    }
}
```

输出

```
Error type 12 at line 18: Operands type mistaken
```

说明：数组下标非整数。

1.13 A-13

输入

```
struct {
    int a;
    int b;
} v;
int test_function()
```

```
{
    int temp = 2;
    v.a = temp + v.b * v.a;
    v.b = temp.b;
}
```

输出

```
Error type 13 at line 9: 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 5。

1.14 A-14

输入

```
struct _Vector_1 {
    float x, y;
};
struct _Vector_2 {
    float a, b, c;
};

int main()
{
    struct _Vector_1 v1;
    struct _Vector_2 v2;
    float p,q;
    p = v1.x + v1.y * v1.z;
    q = v2.a + v2.b * v2.c;
    return 0;
}
```

输出

```
Error type 14 at line 13: Un-existed field "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 和 7。

1.15 A-15

输入

```
struct Food {
    int num, weight, price;
    float price;
    float time;
};

int main()
{
    struct Food food;
    food.weight = 20;
}
```

输出

```
Error type 15 at line 3: Redefined field "price"
```

说明：结构体内部有重复定义的域。有的同学由于 Food 定义错误，就没有将其放入符号表，因此会在第 9 行第 10 行报 Food 未定义，这个不影响得分。

1.16 A-16

输入

```
struct Food {
    int price, weight;
};

int main()
{
    struct Food meat;
    meat.price = 20;
    meat.weight = 34;
    return 0;
```

```
}

struct Food {
    int price1;
    float weight1;
};
```

输出

```
Error type 16 at line 13: Duplicated name "Food"
```

说明：重复定义的结构体 Food。

1.17 A-17

输入

```
struct Food_purchase {
    int price, weight;
};
struct Drink_purchase {
    int sweet;
    int price2;
};

int main()
{
    struct Food_purchase meat;
    struct Food_purchase2 vegetable;
    struct Drink_purchase orange;
    return 0;
}
```

输出

```
Error type 17 at line 12: Undefined struct "Food_purchase2"
```

说明：使用了未定义的结构体 Food_purchase2。

1.18 A-18

输入

```
struct Food {
    int price;
    float weight = 0.19;
};

int main()
{
    struct Food food;
    food.price = 25;
    food.weight = 0.5;
}
```

输出

```
Error type 15 at line 3: "weight" is initialized when defined
```

说明：在结构体中不能初始化变量。

1.19 A-19

输入

```
struct Food {
    int price;
    float weight;
};

struct Food2 {
    int price1;
    int weight1;
}food;

int set_Item(struct Food temp){
```

```
12      temp.price = 10;
13      temp.weight = 0.5;
14      return 0;
15  }
16  int main()
17  {
18      int temp1 = 20;
19      struct Food food2;
20      food2.price = temp1;
21      food2.weight = 0.5;
22      set_Item(food);
23      set_Item(food2);
24
25  }
```

输出

```
1  Error type 9 at line 22: The method "set_Item" is not applicable for
       the arguments
```

说明：函数实参与形参类型不一致。

1.20 A-20

输入

```
1  struct Food {
2      int price;
3      float weight;
4  } food;
5
6  int main()
7  {
8      int temp = 20;
9      struct Food food2;
10     food2.price = 25;
```

```
11      food2.weight = 0.5;
12
13      food.price = temp;
14
15  }
16
17  struct temp{
18      int price1;
19      float weight1;
20  };
```

输出

```
1  Error type 16 at line 17: Duplicated name "temp"
```

说明：结构体与之前定义的变量重名，错误报在第 8 行也对。

2 B 组测试用例

本组测试用例共 2 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 Triangle{
2      int edge1;
3      int edge2;
4      int edge3;
5  };
6
7  struct Circle{
8      int r;
9  };
```

```
10
11 struct Rectangle{
12     int long_edge;
13     int short_edge;
14 };
15 struct Triangle setTriangle(int a,int b, int c){
16     struct Triangle triangle_set;
17     triangle_set.edge1 = a;
18     triangle_set.edge2 = b;
19     triangle_set.edge3 = d;
20     return triangle_set;
21 }
22 struct Circle setCircle(int d){
23     struct Circle circle_set;
24     circle_set.r = d;
25     return circle_set;
26 }
27 struct Rectangle setRectangle(int l, int s){
28     struct Rectangle rectangle_set;
29     rectangle_set.long_edge = l;
30     rectangle_set.short_edge = s;
31     return rectangle_set;
32 }
33 int compareReturnShortest(struct Triangle triangle, struct Circle
    circle, struct Rectangle rectangle){
34     int perimeter1 = triangle.edge1 + triangle.edge2 + triangle.edge3
       ;
35     int perimeter2 = circle.r * 3 * 2;
36     int perimeter3 = 2 * (rectangle.long_edge + rectangle.short_edge)
       ;
37     if(perimeter1 < perimeter2){
38         if(perimeter1 <= perimeter3)
```

```
39              return 1;
40          else return 3;
41      }
42      else {
43          if(perimeter2 <=perimeter3)
44              return 2;
45          else return 3;
46      }
47      return 0;
48  }
49
50  int main(){
51      struct Triangel temp1 = setTriangle(3,4,5);
52      struct Circle temp2 = setCircle(2);
53      struct Rectangle temp3 = setTriangle(3,4,1);
54      return compareReturnShortest(temp1,temp1,temp3);
55
56  }
```

输出

```
1  Error type 1 at line 19: Undefined variable "d"
2  Error type 17 at line 51: Undefined struct "Triangel"
3  Error type 5 at line 53: Type mismatched
4  Error type 9 at line 54: The method "compareReturnShortest" is not
       applicable for the arguments
```

说明：输出中的 4 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 19 行的错误可能会导致错误类型 5，因为 d 的类型未知；第 51 行的变量 Triangel 没有定义，Triangel 的类型可以看作未知，因此可能会报出一个类型 5 错误，连锁不仅限与此，合理即可。

2.2 B-2

输入

```
1  struct Array_Vector{
```

```
    int length = 0;
    int array[100];
};
struct {
    int len;
    int array2[5];
}vector_define;

int main(){
    struct Array_Vector vector_array;
    int i, j;
    vector_array.length = 100;
    i = 0;
    while(i<vector_array.length){
        if(i<vector_define.len)
            vector_array.array[i] = vector_define.array2[i];
        else
            vector_array.array[i] = i * i[i] + 1;
        i = i + 1;
    }
    j = 0;
    while(j<vector_define.len){
        vector_define.array[i] = j * 2 + 1;
        vector_define.array2[i] + 0 = vector_define.array2[i] * j;
    }

}
```

输出

```
Error type 15 at line 2: "length" is initialized when defined
Error type 10 at line 19: Illegal use of "[]"
Error type 14 at line 24: Un-existed field "array"
```

```
Error type 6 at line 25: The left-hand side of an assignment must be
    a variable
```

说明：输出中的 4 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 19 行的错误可能会导致错误类型 5 和 7，因为 i[i] 的类型未知；第 24 行可能会报出一个类型 5 错误，连锁不仅限与此，合理即可。

3 C 组测试用例

本组测试用例共 2 个，不包含任何错误。

3.1 C-1

输入

```
struct Triangle{
    int edge1;
    int edge2;
    int edge3;
};

struct Circle{
    int r;
};

struct Rectangle{
    int long_edge;
    int short_edge;
};
struct Triangle setTriangle(int a,int b, int c){
    struct Triangle triangle_set;
    triangle_set.edge1 = a;
    triangle_set.edge2 = b;
    triangle_set.edge3 = c;
    return triangle_set;
```

```
}
struct Circle setCircle(int d){
    struct Circle circle_set;
    circle_set.r = d;
    return circle_set;
}
struct Rectangle setRectangle(int l, int s){
    struct Rectangle rectangle_set;
    rectangle_set.long_edge = l;
    rectangle_set.short_edge = s;
    return rectangle_set;
}
int compareReturnShortest(struct Triangle triangle, struct Circle
   circle, struct Rectangle rectangle){
    int perimeter1 = triangle.edge1 + triangle.edge2 + triangle.edge3
       ;
    int perimeter2 = circle.r * 3 * 2;
    int perimeter3 = 2 * (rectangle.long_edge + rectangle.short_edge)
       ;
    if(perimeter1 < perimeter2){
        if(perimeter1 <= perimeter3)
            return 1;
        else return 3;
    }
    else {
        if(perimeter2 <=perimeter3)
            return 2;
        else return 3;
    }
    return -1;
}

```

```
int main(){
    struct Triangle temp1 = setTriangle(3,4,5);
    struct Circle temp2 = setCircle(2);
    struct Rectangle temp3= setRectangle(3,4);
    return compareReturnShortest(temp1,temp2,temp3);

}
```

输出

```
//正常返回，没有任何输出。
```

说明：本测试用例是 B_1 类测试用例的改正版。

3.2 C-2

输入

```
struct Array_Vector{
    int length;
    int array[100];
};
struct {
    int len;
    int array2[5];
}vector_define;

int main(){
    struct Array_Vector vector_array;
    int i, j;
    vector_array.length = 100;
    i = 0;
    while(i<vector_array.length){
        if(i<vector_define.len)
            vector_array.array[i] = vector_define.array2[i];
        else
```

```
            vector_array.array[i] = i * i + 1;
        i = i + 1;
    }
    j = 0;
    while(j<vector_define.len){
        vector_define.array2[i] = j * 2 + 1;
        vector_define.array2[i] = vector_define.array2[i] * j;
    }

}
```

输出

```
//正常返回，没有任何输出。
```

说明：本测试用例是 B_2 类测试用例的改正版。

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，将会倒扣分。

4.1 D-1

输入

```
struct Node{
    int no;
    int name[5];
    float value;
    int nextno;
};

int initial_Node(struct Node temp1){
    int i = 0;
    while(i<5){
```

```
        temp1.name[i] = i;
        i = i + 1;
    }
    temp1.value = 0.0;
    temp1.no = temp1.nextno = -1;
    return 1;
}
int add_Node(struct Node former, struct Node later);



int main(){
    struct Node a, b;
    initial_Node(a);
    initial_Node(b);
    return add_Node(a,b);

}
int add_Node(struct Node former, struct Node later){
    if(former.nextno!=-1 && later.nextno == -1){
        later.nextno = former.nextno;
        former.nextno = later.no;
        return 0;
    }
    else if(former.nextno == -1 && later.nextno != -1){
        former.nextno = later.no;
        return 1;
    }
    return -1;
}
```

输出

```
//正常返回，没有任何输出。
```

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 18 行报出有语法错误 (Error type B at line 18)。

4.2 D-2

输入

```
struct DepartmentGuest{
    int ageG;
    int isMaleG;
    int EQG;
    int scoreG[5];
};
struct DepartmentOwner{
    int ageO;
    int isMaleO;
}owner;

int guestComparison(struct DepartmentGuest guest_a,struct
   DepartmentGuest guest_b){
    int result = 0;
    if(guest_a.ageG > guest_b.ageG)
        result = guest_a.ageG - guest_b.ageG;

    if(guest_a.EQG > guest_b.EQG)
        result = result * 2;
    return result - owner.ageO;
}

int totalEQ(struct DepartmentGuest guest_a, struct DepartmentGuest
   guest_b){
    return guest_a.EQG + guest_b.EQG;
}
```

```
int main(){
    struct DepartmentGuest guest1, guest2;
    int i = 0;
    int result = 0;
    while(i<5){
        result = result + guest1.scoreG[i] + guest2.scoreG[i];
        i = i + 1;
    }

    return result + totalEQ(guest1,guest2) + guestComparison(guest1,
       guest2);

}
```

输出

```
//正常返回，没有任何输出。
```

说明：对于 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（guest_a，guest_b 和 result)。

4.3 D-3

输入

```
struct Vector1{
    int a1;
    int b1;
};
struct Vector2{
    int a2;
    int b2;
};
struct Vector3{
    int a3[5];
    float b3[2][1];
```

```
};
struct Vector4{
    int a4[6];
    float b4[5][1];
};

int main(){
    struct Vector1 v1;
    struct Vector2 v2;
    struct Vector3 v3;
    struct Vector4 v4;
    v1 = v2;
    v3 = v4;


}
```

输出

```
//正常返回，没有任何输出。
```

说明：对于分组 2.3 的同学，应该没有任何输出，其他分组的同学应该在 23 行 24 行识别出类型不匹配。（函数参数类型 Error type 5）

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

5.1 E-1

这组测试用例针对 2.1 分组的同学

输入

```
struct Node{
    int no;
    int name[5];
```

```
4       float value;
5       int nextno;
6   };
7   struct Node initial_Node(struct Node temp1);
8
9   int add_Node(struct Node former, struct Node later, int m);
10
11  int delete_Node(struct Node temp);
12
13  int main(){
14      struct Node a, b;
15      initial_Node(a);
16      initial_Node(b);
17      delete_Node(a);
18      return add_Node(a,b);
19
20  }
21  int add_Node(struct Node former, struct Node later){
22      if(former.nextno!=-1 && later.nextno == -1){
23          later.nextno = former.nextno;
24          former.nextno = later.no;
25          return 0;
26      }
27      else if(former.nextno == -1 && later.nextno != -1){
28          former.nextno = later.no;
29          return 1;
30      }
31      return -1;
32  }
33
34  int initial_Node(struct Node temp1){
35      int i = 0;
```

```
36      while(i<5){
37          temp1.name[i] = i;
38          i = i + 1;
39      }
40      temp1.value = 0.0;
41      temp1.no = temp1.nextno = -1;
42      return 1;
43  }
```

输出

```
1  Error type 19 at line 21: Inconsistent declaration of function "
       add_Node"
2  Error type 19 at line 34: Inconsistent declaration of function "
       initial_Node"
3  Error type 18 at line 11: Undefined function "delete_Node"
```

说明：仅 2.1 分组同学需要测试该用例，需要输出上述的错误信息，其中错误类型 19 也可以输出在第 18 行，第 18 行也可以多报一个错误类型 9。

5.2 E-2

这组测试用例针对 2.2 分组的同学

输入

```
1  struct DepartmentGuest{
2      int ageG;
3      int isMaleG;
4      int EQG;
5      int scoreG[5];
6  };
7  struct DepartmentOwner{
8      int ageO;
9      int isMaleO;
10 }owner;
11
```

```
12 int guestComparison(struct DepartmentGuest guest_a,struct
       DepartmentGuest guest_b){
13     int result = 0;
14     if(guest_a.ageG > guest_b.ageG)
15         result = guest_a.ageG - guest_b.ageG;
16
17     if(guest_a.EQG > guest_b.EQG)
18         result = result * 2;
19     return result - owner.ageO;
20 }
21
22 int totalEQ(struct DepartmentGuest guest_a, struct DepartmentGuest
       guest_b){
23     struct DepartmentGuest guest_a;
24     return guest_a.EQG + guest_b.EQG;
25 }
26 int main(){
27     struct DepartmentGuest guest1, guest2;
28     int i = 0;
29     int result = 0;
30     while(i<5){
31         int j = 0;
32         result = result + guest1.scoreG[i] + guest2.scoreG[i];
33         i = i + 1;
34         j = i;
35     }
36     j = owner.ageO;
37     return result + totalEQ(guest1,guest2) + guestComparison(guest1,
           guest2);
38
39 }
```

输出

```
Error type 3 at Line 23: Redefined variable "guest_a"
Error type 1 at Line 36: Undefined variable "j"
```

说明：仅 2.2 分组同学需要测试该用例，需要输出上述的错误信息。

5.3 E-3

这组测试用例针对 2.3 分组的同学

输入

```
struct Vector1{
    int a1;
    int b1;
};
struct Vector2{
    int a2;
    int b2;
};
struct Vector3{
    int a3[5];
    float b3[2][1];
};
struct Vector4{
    int a4[6];
    float b4[5][1];
};
struct Vector5{
    int a5[6];
    float b5[5];
};
int main(){
    struct Vector1 v1;
    struct Vector2 v2;
    struct Vector3 v3;
```

```
25      struct Vector4 v4;
26      struct Vector5 v5;
27      v1 = v2;
28      v2 = v3;
29      v3 = v4;
30      v4 = v5;
31  }
```

输出

```
1  Error type 5 at line 28: Type mismatched for assignment
2  Error type 5 at line 30: Type mismatched for assignment
```

说明：仅 2.3 分组同学需要测试该用例，需要输出上述的错误信息。

6 结束语

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