

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	4
1.3	A-3	4
1.4	A-4	5
1.5	A-5	6
1.6	A-6	7
1.7	A-7	7
1.8	A-8	8
1.9	A-9	9
1.10	A-10	10
1.11	A-11	11
1.12	A-12	11
1.13	A-13	12
1.14	A-14	13
1.15	A-15	14
1.16	A-16	15
1.17	A-17	16
1.18	A-18	16
1.19	A-19	17
1.20	A-20	18
2	B 组测试用例	19
2.1	B-1	19
2.2	B-2	21
3	C 组测试用例	22
3.1	C-1	23
3.2	C-2	25
4	D 组测试用例	28
4.1	D-1	28
4.2	D-2	30
4.3	D-3	31

5	E 组测试用例	33
5.1	E-1	33
5.2	E-2	35
5.3	E-3	37
6	结束语	38

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 7, 9, 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”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

1.1.1 输入

```
1 struct Product {
2     int sku;
3     float price;
4     int stock;
5 };
6
7 int main() {
8     struct Product item1, item2;
9
10    item1.sku = 1001;
11    item1.price = 29.99;
12    item1.stock = 50;
13
14    item2.sku = 1002;
15    item2.price = discount_price;
16
17    return 0;
18 }
```

1.1.2 输出

```
1 Error type 1 at Line 15: Undefined variable.
```

1.1.3 说明

第 15 行中，`discount_price` 这个变量没有被定义过。这里可以多报一个 5 型错误。

1.2 A-2

1.2.1 输入

```
1 struct Inventory {
2     int item_id;
3     float unit_cost;
4 };
5
6 int main() {
7     struct Inventory box;
8     int total_quantity = 10;
9     float total = 500.0;
10
11     box.item_id = 3005;
12     box.unit_cost = 48.75;
13
14     total = calculate_total(box.unit_cost, total_quantity);
15
16     return 0;
17 }
```

1.2.2 输出

```
1 Error type 2 at Line 14: using a undefined function.
```

1.2.3 说明

第 14 行中，函数 `calculate_total` 没有被定义过。这里可以多报一个 5 型错误。

1.3 A-3

1.3.1 输入

```
1 struct Sensor {
2     int type;
3     float precision;
4     int sampling_rate;
5 };
```

```

6
7
8 int main() {
9     int temperature = 25;
10    float humidity = 0.65;
11
12    struct Sensor device;
13    float device = 3.14;
14 }

```

1.3.2 输出

```

1 Error type 3 at Line 13: Redefined variable.

```

1.3.3 说明

第 13 行局部变量的名称device和第 12 行的重复了。错误也可以报在第 12 行。

1.4 A-4

1.4.1 输入

```

1 struct Vector3D {
2     int x;
3     int y;
4     int z;
5 };
6
7 int calculate_magnitude(int xx, int yy, int zz) {
8     return xx * xx + yy * yy + zz * zz;
9 }
10
11 int main() {
12     struct Vector3D vec;
13     int result;
14     vec.x = 3;
15     vec.y = 4;
16     vec.z = 5;

```

```

17     result = calculate_magnitude(vec.x, vec.y, vec.z);
18     return result;
19 }
20
21 int calculate_magnitude(int a, int b, int c) {
22     return a * a + b * b + c * c;
23 }

```

1.4.2 输出

```

1 Error type 4 at Line 21: Redefined function.

```

1.4.3 说明

第 21 行定义的函数 `calculate_magnitude` 和第 7 行定义的函数重名了。错误也可以报在第 7 行。

1.5 A-5

1.5.1 输入

```

1 struct Temperature {
2     int celsius;
3     float fahrenheit;
4 };
5
6 int main() {
7     int warning_level;
8     struct Temperature t1;
9     float current_temp = 36.6;
10
11     t1.fahrenheit = 98.8;
12     warning_level = t1.fahrenheit;
13
14     return t1.celsius;
15 }

```

1.5.2 输出

```
1 Error type 5 at Line 12: Type mismatched for assignment.
```

1.5.3 说明

第 12 行中，赋值表达式两边的变量类型不一致，不能把一个浮点数变量赋值给一个整数变量。

1.6 A-6

1.6.1 输入

```
1 struct Circle {  
2     float radius;  
3     float center_x;  
4     float center_y;  
5 };  
6  
7 int main() {  
8     struct Circle ball;  
9  
10    float pi = 3.14159;  
11    pi * ball.radius = 15.7;  
12  
13    return 0;  
14 }
```

1.6.2 输出

```
1 Error type 6 at Line 11: LHS are a right-value-only Expression.
```

1.6.3 说明

第 11 行中，两个浮点数之积不能放在赋值号的左边。

1.7 A-7

1.7.1 输入

```

1 struct Matrix {
2     int data[3][3];
3     int rows;
4     int cols;
5 };
6
7 int main() {
8     struct Matrix m1;
9     int vector[3];
10
11     m1.rows = 3;
12     m1.cols = 3;
13
14     m1.data[0][0] = vector + m1.rows;
15
16     return 0;
17 }

```

1.7.2 输出

```

1 Error type 7 at Line 14: only support arithmic operation on int and
   float.

```

1.7.3 说明

第 14 行中，不能把一个浮点型变量与一个数组相加。这里可以多报一个 5 型错误。

1.8 A-8

1.8.1 输入

```

1 struct Sensor {
2     float temperature;
3     int status_code;
4 };
5
6 int read_temperature() {
7     struct Sensor device;

```



```

8     device.temperature = 36.5;
9     return device.temperature;
10 }
11
12 int main() {
13     return 0;
14 }

```

1.8.2 输出

```

1 Error type 8 at Line 9: Type mismatched for return.

```

1.8.3 说明

第 9 行中，实际的返回值类型 `float` 和声明的返回值类型 `int` 不一致。

1.9 A-9

1.9.1 输入

```

1 struct Vector2D {
2     int x;
3     int y;
4 };
5
6 int multiply(int a, int b) {
7     return a * b;
8 }
9
10 int main() {
11     struct Vector2D vec;
12     int product;
13     vec.x = 3;
14     vec.y = 4;
15
16     product = multiply(vec.x);
17
18     return product;

```

```
19 }
```

1.9.2 输出

```
1 Error type 9 at Line 16: Funtion args mismatch.
```

1.9.3 说明

第 16 行中，函数multiply的实参数量与形参数量不符。

1.10 A-10

1.10.1 输入

```
1 int sort() {
2     int n = 5;
3     int data[5];
4     int i = 0, j = 0;
5     while (i < n - 1) {
6         while (j < n - i - 1) {
7             if (data[j] > data[j+1]) {
8                 int temp = data[j];
9                 data[j] = data[j+1];
10                data[j+1] = temp;
11            }
12            j = j + 1;
13        }
14        i = i + 1;
15    }
16    return n[0];
17 }
18
19 int main() {
20     int arr[5];
21     int result = sort();
22     return result;
23 }
```

1.10.2 输出

```
1 Error type 10 at Line 16: Apply [] to non-array variable.
```

1.10.3 说明

第 16 行中，对非数组类型的变量`n`使用了数组索引符号`[]`。这里可以多报一个 8 型错误。

1.11 A-11

1.11.1 输入

```
1 struct Calculator {
2     int result;
3 };
4
5 int compute(int a, int b) {
6     return a + b;
7 }
8
9 int main() {
10     struct Calculator calc;
11     calc.result = 0;
12
13     calc(5, 3);
14     return 0;
15 }
```

1.11.2 输出

```
1 Error type 11 at Line 13: using (..) on a non-function variable.
```

1.11.3 说明

第 13 行中，对非函数类型的变量`calc`使用了函数调用符号`(...)`。

1.12 A-12

1.12.1 输入

```
1 int main() {
2     int temperature[24];
3     float hour = 3.1415;
4
5     temperature[hour] = 25;
6
7     return 0;
8 }
```

1.12.2 输出

```
1 Error type 12 at Line 5: Non-integer index.
```

1.12.3 说明

第 5 行中，不能使用float类型的变量作为数组的索引。可以多报一个 5 型错误。

1.13 A-13

1.13.1 输入

```
1 struct Point {
2     float x;
3     float y;
4 };
5
6 float calculate_distance(struct Point p1, struct Point p2) {
7     float dx = p1.x - p2.x;
8     float dy = p1.y - p2.y;
9     return dx * dx + dy * dy;
10 }
11
12 int main() {
13     struct Point a;
14     struct Point b;
15     int counter = 0;
16     float result;
17 }
```

```

18     a.x = 1.0;
19     a.y = 2.0;
20     b.x = 4.0;
21     b.y = 6.0;
22
23     while (counter < 3) {
24         if (counter / 2 == 0) {
25             result = calculate_distance(a, b);
26         } else {
27             float temp = result;
28             temp.value = temp * 0.5;
29         }
30         counter = counter + 1;
31     }
32
33     return 0;
34 }

```

1.13.2 输出

```

1 Error type 13 at Line 28: using . on a non-structure variable.

```

1.13.3 说明

第 28 行中，对浮点数变量使用了 `.` 操作符。这里可以多报一个 5 型错误。

1.14 A-14

1.14.1 输入

```

1 struct Sensor {
2     float temperature;
3     int status;
4 };
5
6 int checkStatus(struct Sensor s) {
7     if (s.status > 0) {
8         return 1;

```

```

9      } else {
10         return s.error_code;
11     }
12 }
13
14 int main() {
15     struct Sensor device;
16     int result;
17     device.temperature = 36.5;
18     device.status = 1;
19
20     result = checkStatus(device);
21
22     while (result == 0) {
23         device.status = device.status + 1;
24         result = checkStatus(device);
25     }
26
27     return 0;
28 }

```

1.14.2 输出

```

1 Error type 14 at Line 10: Non-existent field.

```

1.14.3 说明

第 10 行中，访问了未定义的域 `error_code`。这里可以多报一个 8 型错误。

1.15 A-15

1.15.1 输入

```

1 struct InventoryItem {
2     int id;
3     float price;
4     float id;
5     int category[20];

```

```
6 };
7
8 int main() {
9     struct InventoryItem book;
10    int updated;
11    book.id = 1001;
12    book.price = 45.8;
13
14    return 0;
15 }
```

1.15.2 输出

```
1 Error type 15 at Line 4: Redefined field.
```

1.15.3 说明

第 4 行中，id 与第 2 行重复。该错误可以报在第 2 行。

1.16 A-16

1.16.1 输入

```
1 struct Coordinate {
2     float x;
3     float y;
4 };
5
6 struct Coordinate {
7     float z;
8     int axis;
9 };
10
11 int main() {
12     struct Coordinate point;
13     point.x = 1.5;
14     point.y = 3.2;
15 }
```

```
16     return 0;
17 }
```

1.16.2 输出

```
1 Error type 16 at Line 6: Redefined structure.
```

1.16.3 说明

第 6 行中，定义的结构体Coordinate和已经定义过的结构体重名了，也可以报在第 7 行。可以多报与struct Coordinate相关的错误。

1.17 A-17

1.17.1 输入

```
1 struct DefinedStruct {
2     int value;
3 };
4
5 int main() {
6     struct UndefinedStruct s;
7     struct DefinedStruct ds;
8     ds.value = 10;
9     return 0;
10 }
```

1.17.2 输出

```
1 Error type 17 at Line 6: Missing previous structure definition.
```

1.17.3 说明

第 6 行中，使用了未被定义的结构体类型UndefinedStruct。

1.18 A-18

1.18.1 输入


```

1 struct Order {
2     int id;
3     struct {
4         float weight = 3.5;
5         int unit[4];
6     } details;
7 };
8
9 int main() {
10     struct Order myOrder;
11     myOrder.id = 2023;
12     return 0;
13 }

```

1.18.2 输出

```

1 Error type 15 at Line 4: can't initialize variable inside struct body
.

```

1.18.3 说明

第 4 行中，结构体在定义时不能对域进行初始化。

1.19 A-19

1.19.1 输入

```

1 struct Matrix {
2     int data[3][3];
3     int rows;
4     int cols;
5 };
6
7
8 int main() {
9     struct Matrix m1,m2;
10    m1.rows = 3;
11    m1.cols = 3;

```

```

12     m2.rows = 4;
13     m2.cols = 4;
14     m2 = m1 + m2;
15     return 0;
16 }

```

1.19.2 输出

```

1 Error type 7 at Line 14: Type mismatched for operands.

```

1.19.3 说明

第 14 行中，不能让结构体相加。也可以报出错误类型 5。

1.20 A-20

1.20.1 输入

```

1 struct Dish {
2     float price;
3     struct Comment {
4         int count;
5         int positive;
6         int negative;
7     } comments[10];
8 };
9
10 struct Comment getComment(struct Dish dish) {
11     return dish.comments;
12 }
13
14 int main() {
15     struct Comment myComment;
16     struct Dish myDish;
17     myComment = getComment(myDish);
18     return 0;
19 }

```

1.20.2 输出

```
1 Error type 8 at Line 11: Type mismatched for return.
```

1.20.3 说明

第 11 行中，返回值类型与声明的类型不匹配。

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

2.1.1 输入

```
1 struct Point {
2     int x;
3     int y;
4 };
5
6 int distanceSquared(struct Point px, struct Point py) {
7     int dx = px.x - py.x;
8     int dy = px.y - py.y;
9     return dx * dx + dy * dy;
10 }
11
12 int main() {
13     struct Point p1, p2;
14     int x_values[5];
15     int i = 0;
16     int total_distance = 0;
17
18     p1.x = 3;
19     p1.y = 4;
20     p2.x = 6;
21     p2.y = 8;
```

```

22
23     while (i < 5) {
24         total_distance = total_distance + x_values[i];
25         i = i + 1;
26     }
27
28     total_distance = x_values + p1;
29
30     distanceSquared(p1, p2) = total_distance;
31
32     if (total_distance > 10) {
33         p1.x = p1.x + 1;
34     } else {
35         p1.x = p1.x - 1;
36     }
37
38     total_distance = total_distance.x;
39
40     p1.y = p1.y + 2;
41
42     total_distance = p1.z;
43
44     return total_distance;
45 }

```

2.1.2 输出

```

1 Error type 7 at Line 28: Type mismatched for operands.
2 Error type 6 at Line 30: Invalid left value.
3 Error type 13 at Line 38: Applying . to non-structure variable 0.
4 Error type 14 at Line 42: Non-existent field.

```

2.1.3 说明

第 28 行中，操作数类型不匹配，此处可以多报一个 5 型错误。第 30 行中，函数的返回值是右值，不能放在赋值表达式的左边。第 38 行中，对非结构体变量使用“.”操作符，此处可以多报一个 5 型错误。第 42 行中，访问了不存在的域，此处可以多报一个 5 型错误。

2.2 B-2

2.2.1 输入

```
1 struct StudentInfo {
2     int student_id;
3     int student_score;
4 };
5
6 int computeTotalMarks(struct StudentInfo first_stu, struct
    StudentInfo second_stu, struct StudentInfo third_stu) {
7     return first_stu.student_score + second_stu.student_score +
        third_stu.student_score;
8 }
9
10 int determineTopStudent(struct StudentInfo stu1, struct StudentInfo
    stu2) {
11     int best_id = stu1.student_id;
12     int highest_score = stu1.student_score;
13
14     if (stu2.student_score > highest_score) {
15         highest_score = stu2.student_score;
16         best_id = stu2.student_id;
17     }
18
19     return best_id;
20 }
21
22 int displayFinalOutcome(float total_marks, int best_stu_id) {
23     if (total_marks > 90) {
24         best_stu_id[0] = 5;
25     }
26
27     return 0.0;
28 }
29
30 int main() {
31     struct StudentInfo student_one, student_two, student_three;
```

```

32     int final_score;
33     int top_student_id;
34
35     student_one.student_id = 101;
36     student_one.student_score = 85;
37     student_two.student_id = 102;
38     student_two.student_score = 92;
39     student_three.student_id = 103;
40     student_three.student_score = 78;
41
42     final_score = computeTotalMarks(student_one, student_two,
43                                     student_three);
44
45     top_student_id = determineTopStudent(student_one, student_two);
46
47     displayFinalOutcome(final_score, top_student_id);
48
49     return final_score;
50 }

```

2.2.2 输出

```

1 Error type 7 at Line 23: unmatched operands.
2 Error type 10 at Line 24: using [...] on a non-array variable.
3 Error type 8 at Line 27: the return value contradicts the definition
  of the function.
4 Error type 9 at Line 46: unmatched parameters when calling function:
  displayFinalOutcome .

```

2.2.3 说明

第 23 行中，操作数类型不匹配。第 24 行中，使用 `[]` 操作符时，操作数必须是数组类型。第 27 行中，返回值类型与声明的类型不匹配。第 46 行中，函数实参类型与声明的类型不匹配。

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

3.1.1 输入

```
1 struct StudentData {
2     int student_id;
3     int student_score;
4 };
5
6 struct ProcessedStudent {
7     int processed_id;
8     int updated_score;
9 };
10
11 int sumScores(int score_array[5], int total_count) {
12     int index_counter = 0;
13     int overall_sum = 0;
14     while (index_counter < total_count) {
15         overall_sum = overall_sum + score_array[index_counter];
16         index_counter = index_counter + 1;
17     }
18     return overall_sum;
19 }
20
21 int getMaxScore(int score_collection[5], int entry_size) {
22     int position_marker = 0;
23     int highest_value = score_collection[0];
24
25     while (position_marker < entry_size) {
26         if (score_collection[position_marker] > highest_value) {
27             highest_value = score_collection[position_marker];
28         }
29         position_marker = position_marker + 1;
30     }
31     return highest_value;
32 }
33
34 struct ProcessedStudent updateStudentData(struct StudentData
```

```

    single_record, int extra_points) {
35     struct ProcessedStudent modified_entry;
36     modified_entry.processed_id = single_record.student_id;
37     modified_entry.updated_score = single_record.student_score +
        extra_points;
38     return modified_entry;
39 }
40
41 int main() {
42     struct StudentData entry_one, entry_two, entry_three;
43     struct ProcessedStudent final_one, final_two;
44     int score_list[3];
45     int id_list[3];
46     int final_sum, top_score, iteration_var;
47
48     iteration_var = 0;
49     while (iteration_var < 3) {
50         id_list[iteration_var] = 201 + iteration_var;
51         score_list[iteration_var] = 75 + iteration_var * 6;
52         iteration_var = iteration_var + 1;
53     }
54
55     entry_one.student_id = id_list[0];
56     entry_one.student_score = score_list[0];
57     entry_two.student_id = id_list[1];
58     entry_two.student_score = score_list[1];
59     entry_three.student_id = id_list[2];
60     entry_three.student_score = score_list[2];
61
62     final_sum = sumScores(score_list, 3);
63     top_score = getMaxScore(score_list, 3);
64
65     iteration_var = 0;
66     while (iteration_var < 3) {
67         if (score_list[iteration_var] == top_score) {
68             final_one = updateStudentData(entry_one, 4);
69         } else {

```



```

70         final_two = updateStudentData(entry_two, 2);
71     }
72     iteration_var = iteration_var + 1;
73 }
74
75 if (final_one.updated_score > final_two.updated_score) {
76     if (final_one.updated_score > entry_three.student_score) {
77         return final_one.processed_id;
78     } else {
79         return entry_three.student_id;
80     }
81 } else {
82     if (final_two.updated_score > entry_three.student_score) {
83         return final_two.processed_id;
84     } else {
85         return entry_three.student_id;
86     }
87 }
88 }

```

3.1.2 输出

```

1 // 正常返回，没有任何输出。

```

3.2 C-2

3.2.1 输入

```

1 struct CalculationData {
2     int input_value;
3     int factorial_result;
4 };
5
6 struct ComputedResult {
7     int gcd_result;
8     int sum_result;
9 };
10

```

```

11 int computeFactorial(int number_input) {
12     if (number_input == 1) {
13         return 1;
14     }
15     return number_input * computeFactorial(number_input - 1);
16 }
17
18 int findGreatestCommonDivisor(int first_number, int second_number) {
19     while (second_number != 0) {
20         int temp_value = second_number;
21         second_number = first_number - (first_number / second_number)
22             * second_number;
23         first_number = temp_value;
24     }
25     return first_number;
26 }
27
28 int computeArraySum(int num_array[5], int array_size) {
29     int index_marker = 0;
30     int total_sum = 0;
31     while (index_marker < array_size) {
32         total_sum = total_sum + num_array[index_marker];
33         index_marker = index_marker + 1;
34     }
35     return total_sum;
36 }
37
38 struct ComputedResult processComputation(int first_input, int
39     second_input, int values_list[5], int list_size) {
40     struct ComputedResult final_results;
41     final_results.gcd_result = findGreatestCommonDivisor(first_input,
42         second_input);
43     final_results.sum_result = computeArraySum(values_list, list_size
44         );
45     return final_results;
46 }
47

```

```

44 int main() {
45     struct CalculationData factorial_info;
46     struct ComputedResult computed_values;
47     int num_series[3];
48     int input_one, input_two, factorial_output, sum_total, gcd_output
        ;
49     int loop_counter;
50
51     loop_counter = 0;
52     while (loop_counter < 3) {
53         num_series[loop_counter] = loop_counter + 2;
54         loop_counter = loop_counter + 1;
55     }
56
57     input_one = num_series[0] + 5;
58     input_two = num_series[1] + 3;
59
60     factorial_info.input_value = input_one;
61     factorial_info.factorial_result = computeFactorial(input_one);
62
63     computed_values = processComputation(input_one, input_two,
        num_series, 3);
64
65     sum_total = computed_values.sum_result;
66     gcd_output = computed_values.gcd_result;
67
68     if (factorial_info.factorial_result > sum_total) {
69         return factorial_info.factorial_result;
70     } else {
71         if (gcd_output > sum_total) {
72             return gcd_output;
73         } else {
74             return sum_total;
75         }
76     }
77 }

```

3.2.2 输出

```
1 // 正常返回，没有任何输出。
```

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

4.1.1 输入

```
1 struct Computation {
2     int total_sum;
3     int max_value;
4 };
5
6 int calculateSum(int sum_list[3], int sum_count);
7
8 int calculateSum(int input_numbers[3], int entry_limit) {
9     int index1 = 0, sum_result = 0;
10    while (index1 < entry_limit) {
11        sum_result = sum_result + input_numbers[index1];
12        index1 = index1 + 1;
13    }
14    return sum_result;
15 }
16
17 int determineMax(int value_series[3], int count_entries) {
18     int index2 = 0, max_found = value_series[0];
19     while (index2 < count_entries) {
20         if (value_series[index2] > max_found) {
21             max_found = value_series[index2];
22         }
23         index2 = index2 + 1;
24     }
25     return max_found;
```

```

26 }
27
28 struct Computation executeProcess(int first_input, int second_input,
    int num_series[3], int series_length) {
29     struct Computation result_data;
30     result_data.total_sum = calculateSum(num_series, series_length);
31     result_data.max_value = determineMax(num_series, series_length);
32
33     if (first_input > second_input) {
34         result_data.max_value = result_data.max_value + first_input;
35     } else {
36         result_data.total_sum = result_data.total_sum + second_input;
37     }
38
39     return result_data;
40 }
41
42 int main() {
43     struct Computation final_result;
44     int num_list[3];
45     int input_x, input_y;
46     int loop_counter = 0;
47
48     while (loop_counter < 3) {
49         num_list[loop_counter] = loop_counter * 5 + 3;
50         loop_counter = loop_counter + 1;
51     }
52
53     input_x = num_list[0] + 2;
54     input_y = num_list[1] + 4;
55
56     final_result = executeProcess(input_x, input_y, num_list, 3);
57
58     if (final_result.total_sum > final_result.max_value) {
59         return final_result.total_sum;
60     } else {
61         return final_result.max_value;

```

```

62     }
63 }
64
65 int determineMax(int largest_list1[3], int max_count1);

```

4.1.2 输出

```

1 // 正常返回，没有任何输出。

```

4.1.3 说明

3.1 分组的同学没有任何输出，其它分组的同学在第 6 行、第 65 行报语法错误。

```

1 Error Type B at Line 6: syntax error, unexpected SEMI, expecting LC.
2 Error Type B at Line 65: syntax error, unexpected SEMI, expecting LC.

```

4.2 D-2

4.2.1 输入

```

1 struct Coordinate {
2     int x;
3     int y;
4 };
5
6 float calculate_area(int radius) {
7     float pi = 3.14159;
8     return pi;
9 }
10
11 int main() {
12     int counter = 0;
13     int total = 0;
14
15     struct Coordinate pos;
16     int data[3];
17
18     {
19         int counter = 5;

```

```

20     float total = calculate_area(counter);
21     pos.x = counter;
22
23     while (counter > 0) {
24         int data = 99;
25         counter = counter - 1;
26     }
27 }
28
29 data[0] = 99;
30 return 0;
31 }

```

4.2.2 输出

```

1 // 正常返回，没有任何输出。

```

4.2.3 说明

3.2分组的同学没有任何输出。其它分组的同学应该识别出对于变量counter, total, data的重复定义。

```

1 Error type 3 at Line 19: Redefined variable "counter".
2 Error type 3 at Line 20: Redefined variable "total".
3 Error type 3 at Line 24: Redefined variable "data".

```

4.3 D-3

4.3.1 输入

```

1 struct S1 { int a; float b; } s1;
2 struct S2 { int c; float d; } s2;
3
4 struct M1 { int arr1[3][3]; } m1;
5 struct M2 { int arr2[2][4]; } m2;
6
7 struct Data {
8     int values[3];
9     struct { int idx; float scorex; } meta;

```

```

10 };
11
12 struct Info {
13     int numbers[4];
14     struct {
15         int id;
16         float score;
17     } detailed_info;
18 };
19
20 struct Info get_info(struct Data ret) {
21     return ret;
22 }
23
24 int main() {
25     float total;
26     struct Data data;
27     struct Info info;
28
29     s1 = s2;
30
31     m1 = m2;
32
33     data = get_info(data);
34
35     return 0;
36 }

```

4.3.2 输出

```

1 // 正常返回，没有任何输出。

```

4.3.3 说明

3.3 分组的同学没有任何输出，其它分组的同学应该在第 21 行、第 29 行、第 31 行、第 33 行报错。

```

1 Error type 8 at Line 21: the return value contradicts the definition
  of the function.

```



```
2 Error type 5 at Line 29: Type mismatched for assignment.
3 Error type 5 at Line 31: Type mismatched for assignment.
4 Error type 5 at Line 33: Type mismatched for assignment.
```

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

5.1.1 输入

```
1 struct Computation {
2     int total_sum;
3     int max_value;
4 };
5
6 struct Calculation {
7     int overall_sum;
8     int max_result;
9 };
10
11 int calculateMulti(int sum_list[3], int sum_count);
12
13
14 int calculateSum(int input_numbers[3], int entry_limit) {
15     int index1 = 0, sum_result = 0;
16     while (index1 < entry_limit) {
17         sum_result = sum_result + input_numbers[index1];
18         index1 = index1 + 1;
19     }
20     return sum_result;
21 }
22
23 int determineMax(int value_series[3], int count_entries) {
24     int index2 = 0, max_found = value_series[0];
25     while (index2 < count_entries) {
26         if (value_series[index2] > max_found) {
```

```

27         max_found = value_series[index2];
28     }
29     index2 = index2 + 1;
30 }
31 return max_found;
32 }
33
34 struct Computation executeProcess(int first_input, int second_input,
    int num_series[3], int series_length) {
35     struct Computation result_data;
36     result_data.total_sum = calculateSum(num_series, series_length);
37     result_data.max_value = determineMax(num_series, series_length);
38
39     if (first_input > second_input) {
40         result_data.max_value = result_data.max_value + first_input;
41     } else {
42         result_data.total_sum = result_data.total_sum + second_input;
43     }
44
45     return result_data;
46 }
47
48 int main() {
49     struct Computation final_result;
50     int num_list[3];
51     int input_x, input_y;
52     int loop_counter = 0;
53
54     while (loop_counter < 3) {
55         num_list[loop_counter] = loop_counter * 5 + 3;
56         loop_counter = loop_counter + 1;
57     }
58
59     input_x = num_list[0] + 2;
60     input_y = num_list[1] + 4;
61
62     final_result = executeProcess(input_x, input_y, num_list, 3);

```

```

63
64     if (final_result.total_sum > final_result.max_value) {
65         return final_result.total_sum;
66     } else {
67         return final_result.max_value;
68     }
69 }
70
71 int determineMax(int largest_list1[3], int max_count1);
72
73 struct Calculation executeProcess(int first_inputx, int second_inputx
    , int num_seriesx[3], int series_lengthx);

```

5.1.2 输出

```

1 Error type 19 at Line 73: function definition is inconsistent with
    previous declaration.
2 Error type 18 at Line 11: function declared but not defined

```

5.1.3 说明

仅 3.1 分组的同学需要测试这个用例，并且报出错误。错误 19 也可以报在第 34 行。

若选做内容全部完成，则不需要报错误类型 19，这种情况的报错信息如下：

```

1 Error type 18 at Line 11: function declared but not defined

```

5.2 E-2

5.2.1 输入

```

1 struct Coordinate {
2     int x;
3     int y;
4 };
5
6 struct Area {
7     float area;
8 };
9

```

```

10 int calculate_area(int radius) {
11     if (radius > 0) {
12         return radius;
13     }
14     return 0;
15 }
16
17 int main() {
18     int counter = 0;
19     int total = 0;
20
21     struct Coordinate pos;
22     int data[3];
23
24     {
25         struct Area pos;
26         int counter;
27         float total;
28         float data[10];
29
30         total = calculate_area(counter);
31
32         while (counter > 0) {
33             int data = 99;
34             counter = counter - 1;
35             pos.x = counter;
36             data[counter] = 99;
37         }
38     }
39
40     data[0] = 99;
41     pos.x = 99;
42     return 0;
43 }

```

5.2.2 输出

```
1 Error type 5 at Line 30: Type mismatched for assignment.
2 Error type 14 at Line 35: Trying to visit a structure field which is
  undefined.
3 Error type 10 at Line 36: using [...] on a non-array variable.
```

5.2.3 说明

仅 3.2 分组的同学需要测试这个用例，并且报出错误。

5.3 E-3

5.3.1 输入

```
1 struct Data {
2     float values[3];
3     struct { float idx; int scorex; } meta[10];
4     float fvalue;
5 };
6
7 struct Info {
8     float numbers[4];
9     struct {
10         int id;
11         float score;
12     } detailed_info[10];
13 };
14
15 struct Info get_info(struct Data ret) {
16     return ret;
17 }
18
19 int main() {
20     struct Data data;
21     struct Info info;
22
23     data.meta = info.detailed_info;
24
25     return 0;
```

26 | }

5.3.2 输出

```
1 Error type 8 at Line 16: the return value contradicts the difinition
  of the function.
2 Error type 5 at Line 23: Type mismatched for assignment.
```

5.3.3 说明

仅 3.3 分组的同学需要测试这个用例，并且报出错误。

6 结束语

若对本文档有任何疑问，可写邮件与孙伟杰助教联系，注意同时抄送给许畅老师。