测试报告

一、注释符

测试样例：

```
#include<iostream>
#include<math.h>
using namespace std;

int main() {
    // 测试1
    // 测试2
    /*
    * 测试3，多行测试
    */
    cout << "The value of c is " << c << endl;
    return 0;
}
```

测试结果：

```
# 特殊符号
include 关键字
<iostream> 特殊符号
# 特殊符号
include 关键字
<math.h> 特殊符号
using 关键字
namespace 关键字
std 关键字
; 特殊字符
int 关键字
main 关键字
( 特殊字符
) 特殊字符
{ 特殊字符
// 测试1 注释
// 测试2 注释
/*
    * 测试3，多行测试
    */ 注释
cout 关键字
<< 运算符
"The value of c is " 字符串
<< 运算符
c 标识符
<< 运算符
endl 标识符
```

```
; 特殊字符
return 关键字
0 数字
; 特殊字符
} 特殊字符
```

测试截图：

测试结论：

测试注释全部正常通过

二、运算符

测试样例：

```cpp
#include <iostream>

using namespace std;

int main() {
    int a = 10;
    int b = 5;

    // Arithmetic operators
```

```
    int sum = a + b;
    int difference = a - b;
    int product = a * b;
    int division = a / b;
    int modulus = a % b;

    // Compound assignment operators
    sum += 5;
    difference -= 2;
    product *= 3;
    division /= 2;

    // Increment and decrement operators
    int x = 7;
    x++;
    int y = 10;
    y--;

    // Relational operators
    bool isEqual = (a == b);
    bool isNotEqual = (a != b);
    bool isLess = (a < b);
    bool isLessOrEqual = (a <= b);
    bool isGreater = (a > b);
    bool isGreaterOrEqual = (a >= b);

    // Logical operators
    bool logicalAnd = (true && false);
    bool logicalOr = (true || false);
    bool logicalNot = !true;

    // Conditional operator (ternary)
    int max = (a > b) ? a : b;

    // Bitwise operators (not commonly used)
    int bitwiseAnd = a & b;
    int bitwiseOr = a | b;
    int bitwiseXor = a ^ b;
    int bitwiseNot = ~a;

    // Shift operators (left and right shift)
    int leftShift = a << 2;
    int rightShift = a >> 1;

    // Commenting examples
    // This is a single-line comment

    /*
    This is a
    multi-line comment
    */

    // Division by zero (causes a runtime error)
    // int result = a / 0;
```

```
    return 0;
}
```

测试结果：

```
# 特殊符号
include  关键字
<iostream> 特殊符号
using 关键字
namespace 关键字
std 关键字
; 特殊字符
int 关键字
main 关键字
( 特殊字符
) 特殊字符
{ 特殊字符
int 关键字
a 标识符
= 运算符
10 数字
; 特殊字符
int 关键字
b 标识符
= 运算符
5 数字
; 特殊字符
// Arithmetic operators 注释
int 关键字
sum 标识符
= 运算符
a 标识符
+ 运算符
b 标识符
; 特殊字符
int 关键字
difference 标识符
= 运算符
a 标识符
- 运算符
b 标识符
; 特殊字符
int 关键字
product 标识符
= 运算符
a 标识符
* 运算符
b 标识符
; 特殊字符
int 关键字
division 标识符
= 运算符
```

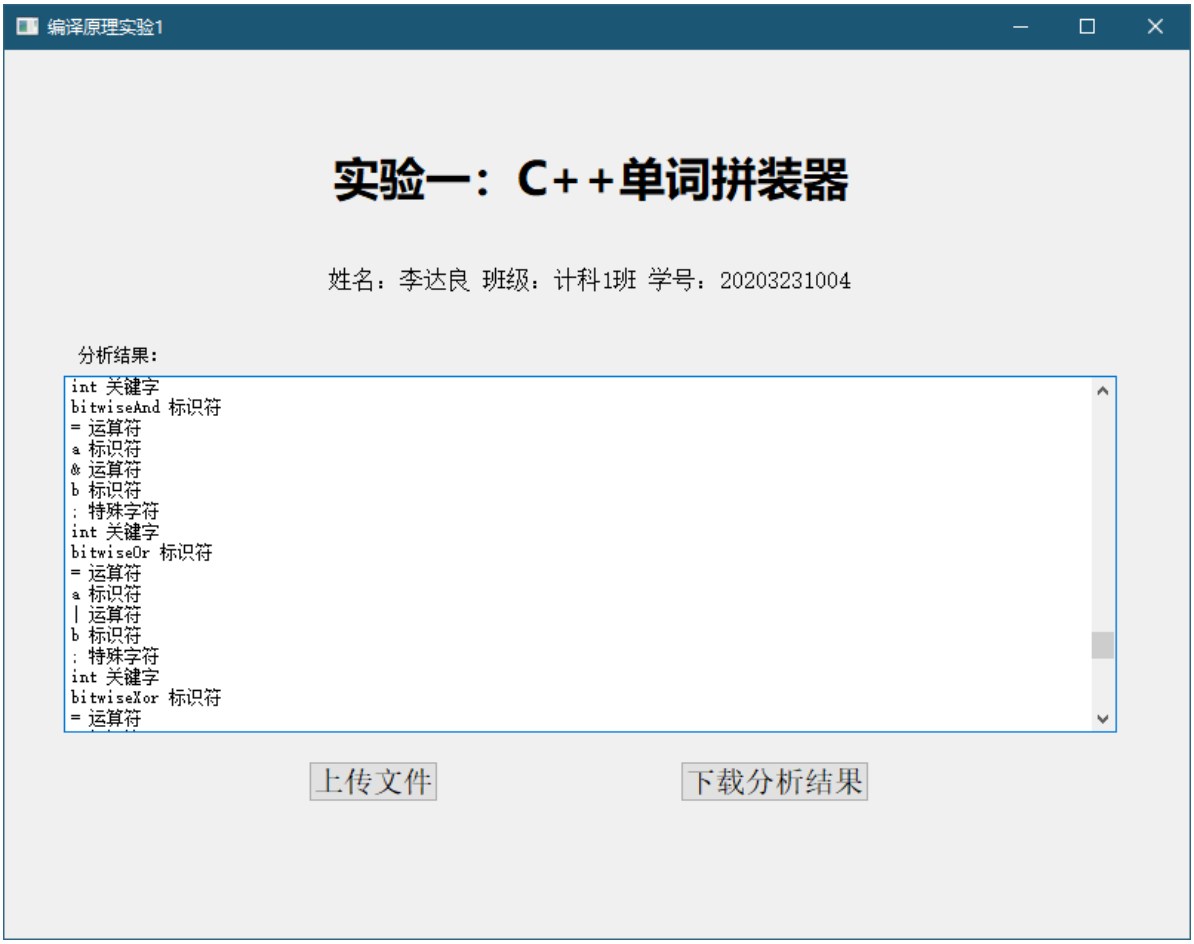
```
a 标识符
/ 运算符
b 标识符
; 特殊字符
int 关键字
modulus 标识符
= 运算符
a 标识符
% 运算符
b 标识符
; 特殊字符
// Compound assignment operators 注释
sum 标识符
+= 运算符
5 数字
; 特殊字符
difference 标识符
-= 运算符
2 数字
; 特殊字符
product 标识符
*= 运算符
3 数字
; 特殊字符
division 标识符
/= 运算符
2 数字
; 特殊字符
// Increment and decrement operators 注释
int 关键字
x 标识符
= 运算符
7 数字
; 特殊字符
x 标识符
++ 运算符
; 特殊字符
int 关键字
y 标识符
= 运算符
10 数字
; 特殊字符
y 标识符
-- 运算符
; 特殊字符
// Relational operators 注释
bool 关键字
isEqual 标识符
= 运算符
( 特殊字符
a 标识符
== 运算符
b 标识符
) 特殊字符
; 特殊字符
```

```
bool 关键字
isNotEqual 标识符
= 运算符
( 特殊字符
a 标识符
!= 运算符
b 标识符
) 特殊字符
; 特殊字符
bool 关键字
isLess 标识符
= 运算符
( 特殊字符
a 标识符
< 运算符
b 标识符
) 特殊字符
; 特殊字符
bool 关键字
isLessOrEqual 标识符
= 运算符
( 特殊字符
a 标识符
<= 运算符
b 标识符
) 特殊字符
; 特殊字符
bool 关键字
isGreater 标识符
= 运算符
( 特殊字符
a 标识符
> 运算符
b 标识符
) 特殊字符
; 特殊字符
bool 关键字
isGreaterOrEqual 标识符
= 运算符
( 特殊字符
a 标识符
>= 运算符
b 标识符
) 特殊字符
; 特殊字符
// Logical operators 注释
bool 关键字
logicalAnd 标识符
= 运算符
( 特殊字符
true 关键字
&& 运算符
false 关键字
) 特殊字符
; 特殊字符
```

```
bool 关键字
logicalOr 标识符
= 运算符
( 特殊字符
true 关键字
|| 运算符
false 关键字
) 特殊字符
; 特殊字符
bool 关键字
logicalNot 标识符
= 运算符
! 运算符
true 关键字
; 特殊字符
// Conditional operator (ternary) 注释
int 关键字
max 标识符
= 运算符
( 特殊字符
a 标识符
> 运算符
b 标识符
) 特殊字符
? 运算符
a 标识符
: 特殊字符
b 标识符
; 特殊字符
// Bitwise operators (not commonly used) 注释
int 关键字
bitwiseAnd 标识符
= 运算符
a 标识符
& 运算符
b 标识符
; 特殊字符
int 关键字
bitwiseOr 标识符
= 运算符
a 标识符
| 运算符
b 标识符
; 特殊字符
int 关键字
bitwiseXor 标识符
= 运算符
a 标识符
^ 运算符
b 标识符
; 特殊字符
int 关键字
bitwiseNot 标识符
= 运算符
~ 运算符
```

```
a 标识符
; 特殊字符
// Shift operators (left and right shift) 注释
int 关键字
leftShift 标识符
= 运算符
a 标识符
<< 运算符
2 数字
; 特殊字符
int 关键字
rightShift 标识符
= 运算符
a 标识符
>> 运算符
1 数字
; 特殊字符
// Commenting examples 注释
// This is a single-line comment 注释
/*
    This is a
    multi-line comment
    */ 注释
// Division by zero (causes a runtime error) 注释
// int result = a / 0; 注释
return 关键字
0 数字
; 特殊字符
} 特殊字符
```

测试截图：



测试结论：

对于大部分运算符，基本上没有问题，除了三目运算符`？：`中的`：`识别为了特殊字符，其他运算符均识别正确

三、数字

测试程序：

```
#include<iostream>

using namespace std;

int main() {
    int a = 0xffff;
    a = 0XFFFF;
    int b = 20;
    float c = 3.14;
    float d = 0.123;
    float e = 12.34;
    float f = 3E+4;
    float g = 3.0e+10;
    int h = 0;
    float i = 0.0;
    float temp = -1.542e-3;
    return 0;
```

```
}
```

测试结果：

```
# 特殊符号
include 关键字
<iostream> 特殊符号
using 关键字
namespace 关键字
std 关键字
; 特殊字符
int 关键字
main 关键字
( 特殊字符
) 特殊字符
{ 特殊字符
int 关键字
a 标识符
= 运算符
0xffff 数字
; 特殊字符
a 标识符
= 运算符
0XFFFF 数字
; 特殊字符
int 关键字
b 标识符
= 运算符
20 数字
; 特殊字符
float 关键字
c 标识符
= 运算符
3.14 数字
; 特殊字符
float 关键字
d 标识符
= 运算符
0.123 数字
; 特殊字符
float 关键字
e 标识符
= 运算符
12.34 数字
; 特殊字符
float 关键字
f 标识符
= 运算符
3E+4 数字
; 特殊字符
float 关键字
g 标识符
= 运算符
```

```
3.0e+10 数字
; 特殊字符
int 关键字
h 标识符
= 运算符
0 数字
; 特殊字符
float 关键字
i 标识符
= 运算符
0.0 数字
; 特殊字符
float 关键字
temp 标识符
= 运算符
- 运算符
1.542e-3 数字
; 特殊字符
return 关键字
0 数字
; 特殊字符
} 特殊字符
```

测试截图：

测试结论：

对所有类型的数字进行测试，包括整数、浮点数、科学计数法、带正负号的科学计数法，科学计数法中含有浮点数，十六进制等类型，均测试通过。

四、分界符号或特殊符号

测试程序：

```
#include <iostream>

using namespace std;

int main() {
    int num1 = 5;
    int num2 = 10;

    // If statement with comparison operators
    if (num1 != num2) {
        cout << "num1 is not equal to num2." << endl;
    }
    else {
        cout << "num1 is equal to num2." << endl;
    }

    // Block of code enclosed in curly braces
    {
        int x = 20;
        int y = 30;
        int result = x + y;
        cout << "Result inside the block: " << result << endl;
    }

    // Array declaration and usage with square brackets
    int numbers[3] = { 1, 2, 3 };
    cout << "First element of the array: " << numbers[0] << endl;

    // Function call with parentheses
    cout << "Hello, World!" << endl;

    // Comma operator in a single statement
    int sum = (num1 + num2, num1 - num2, num1 * num2); // Only the last value is
assigned to 'sum'
    cout << "Sum: " << sum << endl;

    // Comma operator in a for loop
    for (int i = 0, j = 10; i < 5; i++, j -= 2) {
        cout << "i: " << i << ", j: " << j << endl;
    }

    // Semicolon as a statement terminator
    cout << "This is the end of the program." << endl;

    return 0;
}
```

测试结果：

```
# 特殊符号
include  关键字
<iostream> 特殊符号
using 关键字
namespace 关键字
std 关键字
; 特殊字符
int 关键字
main 关键字
( 特殊字符
) 特殊字符
{ 特殊字符
int 关键字
num1 标识符
= 运算符
5 数字
; 特殊字符
int 关键字
num2 标识符
= 运算符
10 数字
; 特殊字符
// If statement with comparison operators 注释
if 关键字
( 特殊字符
num1 标识符
!= 运算符
num2 标识符
) 特殊字符
{ 特殊字符
cout 关键字
<< 运算符
"num1 is not equal to num2." 字符串
<< 运算符
endl 标识符
; 特殊字符
} 特殊字符
else 关键字
{ 特殊字符
cout 关键字
<< 运算符
"num1 is equal to num2." 字符串
<< 运算符
endl 标识符
; 特殊字符
} 特殊字符
// Block of code enclosed in curly braces 注释
{ 特殊字符
int 关键字
```

```
x 标识符
= 运算符
20 数字
; 特殊字符
int 关键字
y 标识符
= 运算符
30 数字
; 特殊字符
int 关键字
result 标识符
= 运算符
x 标识符
+ 运算符
y 标识符
; 特殊字符
cout 关键字
<< 运算符
"Result inside the block: " 字符串
<< 运算符
result 标识符
<< 运算符
endl 标识符
; 特殊字符
} 特殊字符
// Array declaration and usage with square brackets 注释
int 关键字
numbers 标识符
[ 特殊字符
3 数字
] 特殊字符
= 运算符
{ 特殊字符
1 数字
, 特殊字符
2 数字
, 特殊字符
3 数字
} 特殊字符
; 特殊字符
cout 关键字
<< 运算符
"First element of the array: " 字符串
<< 运算符
numbers 标识符
[ 特殊字符
0 数字
] 特殊字符
<< 运算符
endl 标识符
; 特殊字符
// Function call with parentheses 注释
cout 关键字
<< 运算符
"Hello, World!" 字符串
```

```
<< 运算符
endl 标识符
; 特殊字符
// Comma operator in a single statement 注释
int 关键字
sum 标识符
= 运算符
( 特殊字符
num1 标识符
+ 运算符
num2 标识符
, 特殊字符
num1 标识符
- 运算符
num2 标识符
, 特殊字符
num1 标识符
* 运算符
num2 标识符
) 特殊字符
; 特殊字符
// Only the last value is assigned to 'sum' 注释
cout 关键字
<< 运算符
"Sum: " 字符串
<< 运算符
sum 标识符
<< 运算符
endl 标识符
; 特殊字符
// Comma operator in a for loop 注释
for 关键字
( 特殊字符
int 关键字
i 标识符
= 运算符
0 数字
, 特殊字符
j 标识符
= 运算符
10 数字
; 特殊字符
i 标识符
< 运算符
5 数字
; 特殊字符
i 标识符
++ 运算符
, 特殊字符
j 标识符
-= 运算符
2 数字
) 特殊字符
{ 特殊字符
cout 关键字
```

```
<< 运算符
"i: " 字符串
<< 运算符
i 标识符
<< 运算符
", j: " 字符串
<< 运算符
j 标识符
<< 运算符
endl 标识符
; 特殊字符
} 特殊字符
// Semicolon as a statement terminator 注释
cout 关键字
<< 运算符
"This is the end of the program." 字符串
<< 运算符
endl 标识符
; 特殊字符
return 关键字
0 数字
; 特殊字符
} 特殊字符
```

测试截图：

测试结论：

所有测试符号均通过，包括了： `(  )  ;  {}  [  ]  ,`

其中，`!=和==`已经归为了运算符上，就不显示特殊符号。

五、标识符和关键词

测试程序：

```
#include <iostream>

using namespace std;

int main() {
    int x = 5;
    int y = 10;
    int abc123 = 15;
    int xy = x + y;
    int _xy = x - y;
    int xy_xy23 = xy * _xy;

    cout << "x: " << x << endl;
    cout << "y: " << y << endl;
    cout << "abc123: " << abc123 << endl;
    cout << "xy: " << xy << endl;
    cout << "_xy: " << _xy << endl;
    cout << "xy_xy23: " << xy_xy23 << endl;

    if (x < y) {
        cout << "x is less than y." << endl;
    } else {
        cout << "x is not less than y." << endl;
    }

    int i = 0;
    while (i < 5) {
        cout << "Iteration " << i + 1 << endl;
        i++;
    }

    for (int j = 0; j < 5; j++) {
        if (j == 3) {
            break;
        }
        if (j == 4) {
            continue;
        }
        cout << "Inside the for loop, j: " << j << endl;
    }

    do {
        cout << "This will be executed at least once." << endl;
    } while (false);
```

```
    switch (x) {
        case 5:
            cout << "x is 5." << endl;
            break;
        case 10:
            cout << "x is 10." << endl;
            break;
        default:
            cout << "x is neither 5 nor 10." << endl;
            break;
    }

    return 0;
}
```

测试结果：

```
# 特殊符号
include  关键字
<iostream> 特殊符号
using 关键字
namespace 关键字
std 关键字
; 特殊字符
int 关键字
main 关键字
( 特殊字符
) 特殊字符
{ 特殊字符
int 关键字
x 标识符
= 运算符
5 数字
; 特殊字符
int 关键字
y 标识符
= 运算符
10 数字
; 特殊字符
int 关键字
abc123 标识符
= 运算符
15 数字
; 特殊字符
int 关键字
xy 标识符
= 运算符
x 标识符
+ 运算符
y 标识符
; 特殊字符
```

```
int 关键字
_xy 标识符
= 运算符
x 标识符
- 运算符
y 标识符
; 特殊字符
int 关键字
xy_xy23 标识符
= 运算符
xy 标识符
* 运算符
_xy 标识符
; 特殊字符
cout 关键字
<< 运算符
"x: " 字符串
<< 运算符
x 标识符
<< 运算符
endl 标识符
; 特殊字符
cout 关键字
<< 运算符
"y: " 字符串
<< 运算符
y 标识符
<< 运算符
endl 标识符
; 特殊字符
cout 关键字
<< 运算符
"abc123: " 字符串
<< 运算符
abc123 标识符
<< 运算符
endl 标识符
; 特殊字符
cout 关键字
<< 运算符
"xy: " 字符串
<< 运算符
xy 标识符
<< 运算符
endl 标识符
; 特殊字符
cout 关键字
<< 运算符
"_xy: " 字符串
<< 运算符
_xy 标识符
<< 运算符
endl 标识符
; 特殊字符
cout 关键字
```

<< 运算符
"xy_xy23: " 字符串
<< 运算符
xy_xy23 标识符
<< 运算符
endl 标识符
; 特殊字符
if 关键字
(特殊字符
x 标识符
< 运算符
y 标识符
) 特殊字符
{ 特殊字符
cout 关键字
<< 运算符
"x is less than y." 字符串
<< 运算符
endl 标识符
; 特殊字符
} 特殊字符
else 关键字
{ 特殊字符
cout 关键字
<< 运算符
"x is not less than y." 字符串
<< 运算符
endl 标识符
; 特殊字符
} 特殊字符
int 关键字
i 标识符
= 运算符
0 数字
; 特殊字符
while 关键字
(特殊字符
i 标识符
< 运算符
5 数字
) 特殊字符
{ 特殊字符
cout 关键字
<< 运算符
"Iteration " 字符串
<< 运算符
i 标识符
+ 运算符
1 数字
<< 运算符
endl 标识符
; 特殊字符
i 标识符
++ 运算符
; 特殊字符

```
} 特殊字符
for 关键字
( 特殊字符
int 关键字
j 标识符
= 运算符
0 数字
; 特殊字符
j 标识符
< 运算符
5 数字
; 特殊字符
j 标识符
++ 运算符
) 特殊字符
{ 特殊字符
if 关键字
( 特殊字符
j 标识符
== 运算符
3 数字
) 特殊字符
{ 特殊字符
break 关键字
; 特殊字符
} 特殊字符
if 关键字
( 特殊字符
j 标识符
== 运算符
4 数字
) 特殊字符
{ 特殊字符
continue 关键字
; 特殊字符
} 特殊字符
cout 关键字
<< 运算符
"Inside the for loop, j: " 字符串
<< 运算符
j 标识符
<< 运算符
endl 标识符
; 特殊字符
} 特殊字符
do 关键字
{ 特殊字符
cout 关键字
<< 运算符
"This will be executed at least once." 字符串
<< 运算符
endl 标识符
; 特殊字符
} 特殊字符
while 关键字
```

```
( 特殊字符
false 关键字
) 特殊字符
; 特殊字符
switch 关键字
( 特殊字符
x 标识符
) 特殊字符
{ 特殊字符
case 关键字
5 数字
: 特殊字符
cout 关键字
<< 运算符
"x is 5." 字符串
<< 运算符
endl 标识符
; 特殊字符
break 关键字
; 特殊字符
case 关键字
10 数字
: 特殊字符
cout 关键字
<< 运算符
"x is 10." 字符串
<< 运算符
endl 标识符
; 特殊字符
break 关键字
; 特殊字符
default 关键字
: 特殊字符
cout 关键字
<< 运算符
"x is neither 5 nor 10." 字符串
<< 运算符
endl 标识符
; 特殊字符
break 关键字
; 特殊字符
} 特殊字符
return 关键字
0 数字
; 特殊字符
} 特殊字符
```

测试截图：

测试结论：

针对以下标识符和保留字，均测试通过：

```
//标识符
 x  y  abc123   xy    _xy    xy_xy23

//保留字
 if  else do while for switch case break   continue default
```