C--编译器测试报告

【样例测试结果】

```
使用指定的源文件和默认语法文件
==================== 编译器开始工作 ====================
源文件: d:\TinyCompiler\data\testsample\test-4.sy
语法文件: ../data/grammar.txt

[第一步] 开始词法分析...
词法分析完成! 结果已写入: ../data/lexical.txt

[第二步] 开始语法分析...
FIRST集已成功输出到 ../data/first.txt
FOLLOW集已成功输出到 ../data/follow.txt
ACTION分析表已保存到: ../data/action_table.csv
GOTO分析表已保存到: ../data/goto_table.csv
语法分析结果已写入: ../data/syntax.txt
语法分析成功!
==================== 词法语法分析结束 ====================
```

```
1   int <KW,1>
2   a   <IDN,a>
3   =   <OP,14>
4   10  <INT,10>
5   ;   <SE,28>
6   int <KW,1>
7   main    <KW,5>
8   (   <SE,24>
9   )   <SE,25>
10  {   <SE,26>
11  a   <IDN,a>
12  =   <OP,14>
13  10  <INT,10>
14  ;   <SE,28>
15  return  <KW,3>
16  0   <INT,0>
17  ;   <SE,28>
18  }   <SE,27>
```

```
73  lAndExpAtom # ;   reduction
74  lAndExp # ;   reduction
75  lOrExpAtom # ;   reduction
76  lOrExp # ;   reduction
77  cond # ;   reduction
78  exp # ;   reduction
79  argExp # ;   move
80  ; # }   reduction
81  matched_stmt # }   reduction
82  stmt # }   reduction
83  blockItem # }   reduction
84  blockItem # }   reduction
85  blockItem # }   move
86  } # #   reduction
87  block # #   reduction
88  funcDef # #   reduction
89  compUnit # #   reduction
90  compUnit # #   reduction
92  compUnit # #   accept
```

【功能集成测试代码】

```
const int MAX = 100;
const float PI = 3.14159;
int add(int a, int b) {
    return a + b;
}
float multiply(float x, float y) {
    return x * y;
}
int max(int a, int b) {
    if (a > b) {
        return a;
    } else {
        return b;
    }
}
int complex_function(int a, int b, int c) {
    int result;

    if (a > b) {
        if (a > c) {
            result = a;
        } else {
            result = c;
        }
    } else {
        if (b > c) {
            result = b;
        } else {
            result = c;
        }
    }
    return result;
}
int test_operators(int a, int b) {
    int result = 0;

    result = a + b;
    result = a - b;
    result = a * b;
    result = a / b;
    result = a % b;
    result = -a;
    if (a > b && a != 0) {
        result = a;
    }

    if (a < b || b == 0) {
        result = b;
    }
    if (!(a <= b) && !(b >= a)) {
        result = a + b;
    }
    return result;
}
int main() {
    int a;
    int b = 20;
    float c = 3.14;
    a = 10;
    int sum = add(a, b);
    float product = multiply(c, 2.0);
    int complex_result = (a + b) * (a - b) / (a * b) + (a % b);
    if (a < b) {
        int temp = a;
        a = b;
        b = temp;

    if (a > 0) {
        if (b > 0) {
            sum = sum + 1;
        } else {
            sum = sum - 1;
        }
    } else {
        sum = 0;
    }

    int max_value = max(a, b);
    int complex_value = complex_function(a, b, sum);
    int op_result = test_operators(a, b);

    return 0;
}
```

【测试结果】

```
未指定参数，将使用默认文件...
==================== 编译器开始工作 ====================
源文件: ../data/testsample/test.sy
语法文件: ../data/grammar.txt

[第一步] 开始词法分析...
词法分析完成! 结果已写入: ../data/lexical.txt

[第二步] 开始语法分析...
FIRST集已成功输出到 ../data/first.txt
FOLLOW集已成功输出到 ../data/follow.txt
ACTION分析表已保存到: ../data/action_table.csv
GOTO分析表已保存到: ../data/goto_table.csv
语法分析结果已写入: ../data/syntax.txt
语法分析成功!
==================== 词法语法分析结束 ====================
```

```
const   <KW,4>              2059    relExpAtom # ;   reduction
int <KW,1>                  2060    relExp # ;   reduction
MAX <IDN,MAX>               2061    eqExpAtom # ;   reduction
=   <OP,14>                 2062    eqExp # ;   reduction
100 <INT,100>               2063    lAndExpAtom # ;   reduction
;   <SE,28>                 2064    lAndExp # ;   reduction
const   <KW,4>              2065    lOrExpAtom # ;   reduction
float   <KW,6>              2066    lOrExp # ;   reduction
PI  <IDN,PI>                2067    cond # ;   reduction
=   <OP,14>                 2068    exp # ;   reduction
3.14159 <FLOAT,3.14159>     2069    argExp # ;   move
;   <SE,28>                 2070    ; # }   reduction
int <KW,1>                  2071    matched_stmt # }   reduction
add <IDN,add>               2072    stmt # }   reduction
(   <SE,24>                 2073    blockItem # }   reduction
int <KW,1>                  2074    blockItem # }   reduction
a   <IDN,a>                 2075    blockItem # }   reduction
,   <SE,29>                 2076    blockItem # }   reduction
int <KW,1>                  2077    blockItem # }   reduction
b   <IDN,b>                 2078    blockItem # }   reduction
)   <SE,25>                 2079    blockItem # }   reduction
{   <SE,26>                 2080    blockItem # }   reduction
return  <KW,3>              2081    blockItem # }   reduction
a   <IDN,a>                 2082    blockItem # }   reduction
+   <OP,9>                  2083    blockItem # }   reduction
b   <IDN,b>                 2084    blockItem # }   reduction
;   <SE,28>                 2085    blockItem # }   reduction
}   <SE,27>                 2086    blockItem # }   move
float   <KW,6>              2087    } # #   reduction
multiply    <IDN,multiply>  2088    block # #   reduction
(   <SE,24>                 2089    funcDef # #   reduction
```

【错误程序测试】

```
使用指定的源文件和默认语法文件
================== 编译器开始工作 ==================
源文件: d:\TinyCompiler\data\testsample\test-1.sy
语法文件: ../data/grammar.txt

[第一步] 开始词法分析...
词法分析完成! 结果已写入: ../data/lexical.txt

[第二步] 开始语法分析...
FIRST集已成功输出到 ../data/first.txt
FOLLOW集已成功输出到 ../data/follow.txt
ACTION分析表已保存到: ../data/action_table.csv
GOTO分析表已保存到: ../data/goto_table.csv
语法分析结果已写入: ../data/syntax.txt
语法分析失败!
================== 词法语法分析结束 ==================
```

```
1   const int a = 10;
2
3   int main(){
4       int b;
5       b=a+5;
6       return 0
7   }
```

解释：缺少分号，语句不完整。

【测试结果】

```
const   <KW,4>      47    number # ;   reduction
int <KW,1>          48    primaryExp # ;   reduction
a   <IDN,a>         49    unaryExp # ;   reduction
=   <OP,14>         50    mulExpAtom # ;   reduction
10  <INT,10>        51    mulExp # ;   reduction
;   <SE,28>         52    addExpAtom # ;   reduction
int <KW,1>          53    addExpAtom # ;   reduction
main    <KW,5>      54    addExp # ;   reduction
(   <SE,24>         55    relExpAtom # ;   reduction
)   <SE,25>         56    relExp # ;   reduction
{   <SE,26>         57    eqExpAtom # ;   reduction
int <KW,1>          58    eqExp # ;   reduction
b   <IDN,b>         59    lAndExpAtom # ;   reduction
;   <SE,28>         60    lAndExp # ;   reduction
b   <IDN,b>         61    lOrExpAtom # ;   reduction
=   <OP,14>         62    lOrExp # ;   reduction
a   <IDN,a>         63    cond # ;   reduction
+   <OP,9>          64    exp # ;   move
5   <INT,5>         65    ; # return   reduction
;   <SE,28>         66    matched_stmt # return   reduction
return  <KW,3>      67    stmt # return   move
0   <INT,0>         68    return # IntConst   move
}   <SE,27>         70    IntConst # }   error
```

【FIRST 集】

```
constExp: IntConst floatConst - + Ident ! (
lAndExpAtom: $ &&
relExp: IntConst floatConst - + Ident ! (
addExp: + - floatConst IntConst ! ( Ident
mulExpAtom: $ / % *
mulExp: IntConst floatConst - + Ident ! (
argFunctionR: $ ,
relExpAtom: $ >= <= > <
unaryOp: ! - +
funcRParam: + - floatConst IntConst ! ( Ident
funcRParams: IntConst floatConst - + Ident ! ( $
eqExpAtom: $ != ==
initVal: + - floatConst IntConst ! ( Ident
number: floatConst IntConst
varDef: Ident
varDecl: int float
block: {
argVarDecl: $ ,
constInitVal: + - floatConst IntConst ! ( Ident
lOrExp: + - floatConst IntConst ! ( Ident
argConst: $ ,
lOrExpAtom: $ ||
blockItem: ! + $ float int const ; { if Ident return ( IntConst floatConst -
decl: float int const
unaryExp: + - floatConst IntConst ! ( Ident
argFunctionF: $ ,
constDecl: const
constDef: Ident
compUnit: const float int void $
eqExp: + - floatConst IntConst ! ( Ident
funcFParam: int float
bType: float int
callFunc: (
funcType: void
program: const void int float $
lAndExp: IntConst floatConst - + Ident ! (
funcFParams: float int $
addExpAtom: $ - +
stmt: IntConst floatConst - + ! ( ; Ident return if {
matched_stmt: IntConst floatConst - + ! ( ; Ident return { if
open_stmt: if
primaryExp: IntConst floatConst Ident (
funcDef: void int float
argExp: + - floatConst IntConst ! ( Ident $
exp: IntConst floatConst - + Ident ! (
cond: IntConst floatConst - + Ident ! (
lVal: Ident
```

【FOLLOW 集】

```
constExp: ; ,
lAndExpAtom: || ) ; ,
relExp: , ; ) || && == !=
addExp: || && != == < ) > >= <= , ;
mulExpAtom: || && != ) > >= - == < + <= , ;
mulExp: || && != >= > ) ; <= , == < + -
argFunctionR: )
relExpAtom: ) ; , || && != ==
unaryOp: Ident ( ! IntConst floatConst - +
funcRParam: ) ,
funcRParams: )
eqExpAtom: , ; ) || &&
initVal: , ;
number: || && == / % + ; * < , <= - >= > ) !=
varDef: ; ,
varDecl: return ( if { # Ident int float void const ; } - floatConst IntConst ! +
block: else floatConst IntConst return ( Ident int float void const ; } - ! + # { if
argVarDecl: ;
constInitVal: , ;
lOrExp: ) ; ,
argConst: ;
lOrExpAtom: , ; )
blockItem: }
decl: } - floatConst IntConst ; float int const void Ident ! + # { if ( return
unaryExp: || && == / % + ; * < , <= - >= > ) !=
argFunctionF: )
constDecl: return ( if { # Ident int float void const ; } - floatConst IntConst ! +
constDef: ; ,
compUnit: #
eqExp: ) ; , || &&
funcFParam: ) ,
bType: Ident
```

```
callFunc: || && != % + == / - * < , <= ; ) > >=
funcType: Ident
program: #
lAndExp: , ; ) ||
funcFParams: )
addExpAtom: || && != >= > ) == < ; <= ,
stmt: } - ! + float int const ; { if Ident ( return IntConst floatConst
matched_stmt: else floatConst IntConst } - ! + float int const ; { if Ident ( return
open_stmt: floatConst IntConst } - ! + float int const ; { if Ident ( return
primaryExp: || && != % + == / - * < , <= ; ) > >=
funcDef: # void int float const
argExp: ;
exp: ) ; ,
cond: , ; )
lVal: || && == != / % + ; * < , <= = - >= > )
```