Test Cases

Kai Wong

04/03/2018

1 Valid Test Cases

1.1 Simple 1

```
1        /* Simple test case */
2        {}$
```

Results: Valid

1.2 Simple 2

```
1        /* Test case for print statement */
2        {
3            print("i love compilers")
4        }$
```

Results: Valid

1.3 Regular

```
1        /* Test case for a 'regular' program*/
2        {
3            int a
4            a = 1
5            print(a)
6            boolean b
7            b = true
8            print(b)
9
10           {
11               int a
12               a = 2
13               print(a)
14           }
15
16           {
17               int a
18               a = 3
19               print(a)
20           }
21
22           string s
23           s = "stra"
24           print(s)
25
26           s = "strb"
27           print(s)
28
29           if (a != 5) {
```

```
30                  print("true")
31              }
32
33              if (a == 5) {
34                  print("false")
35              }
36          }$
```

Results: Valid

1.4 Multiple

```
1           /* Test case for multiple programs */
2           {
3               print("i love compilers")
4               int a
5               a = 2
6               string s
7               s = "ha"
8           }$
9
10          {
11              int b
12              b = 4
13              string s
14              s = "hey"
15          }$
```

Results: Valid

1.5 All Productions thx Tien

```
1           /* Test case for all productions - thx Tien */
2           {
3             /* Int Declaration */
4             int a
5             int b
6             string s
7             boolean z
8
9             z = true
10            s = "kai sucks"
11
12            a = 0
13            b = 0
14
15            /* While Loop */
16            while (a != 3) {
17                print(a)
18                while (b != 3) {
19                      print(b)
20                      b = 1 + b
21                      if (b == 2) {
22                        /* Print Statement */
23                          print("kai sucks"/* This will do nothing */)
24                      }
25                }
26
27                b = 0
28                a = 1 + a
29            }
30          }$
```

Results: Valid

1.6 Crazy One Liner (Lex Pass)

```
/* Test case for crazy one liner */
+${hellotruefalse!======trueprinta=3b=0print("false true")whi33leiftruefalsestring!=
     stringintbooleanaa truewhileif{hi+++==!==}}/*aaahaha*/hahahahaha/*awao*/$
```

Results: Valid (for Lex)

1.7 Crazy One Liner Pt. 2 Thx Tien

```
/*Test case for all productions - thx Tien*/{/*IntDeclaration*/
    intaintbstringsbooleanzz=trues="kai sucks"a=0b=0/*WhileLoop*/while(a!=3){print(a
    )while(b!=3){print(b)b=1+bif(b==2){/*PrintStatement*/print("kai sucks"/*
    Thiswilldonothing*/)}}b=0a=1+a}}$
```

Results: Valid

1.8 WhileStatement

```
/* Test case for WhileStatement */
{
    string s
    int a
    a = 1
    {
        s = "hey there sexy"
        int a
        a = 2
        print(a)
    }
    {
        while (a != 5) {
            a = 1 + a
            print(a)
        }
        print(3 + a)
        print(s)
    }
} $
```

1.9 IfStatement

```
/* Test case for IfStatement */
{
    int a
    a = 1
    if(1 == 1){
        print("nums")
    }
    if(a == a){
        print("ids")
    }
    if("hey" == "hey"){
        print("strings")
    }
    if(true == (a == a)){
        print("booleans")
    }
} $
```

2 Warning Test Cases

2.1 Missing EOP

```
/* Missing EOP */
{
    int b
    b = 4
    string s
    s = "hey"
}
```

Results: WARNING: No EOP [$] detected at end-of-file. Adding to end-of-file...

2.2 Semantic Warnings

```
/* has unused and undeclared variables */
{
    int a
    int b
    a = 3
    b = 4
    {
        string a
        a = "hey"
        print(a)
        print(b)
    }
    print(b)
    string s
    {
        boolean b
        b = false
    }
    string r
    r = "hey"
}$
```

Results:
WARNING: Variable [a] on line 2 col 4 has been initialized but is not used.
WARNING: Variable [s] on line 13 col 4 has been declared but is not initialized properly.
WARNING: Variable [r] on line 18 col 4 has been initialized but is not used.
WARNING: Variable [b] on line 15 col 8 has been initialized but is not used.

3 Lex Fail Programs

3.1 Alan

```
/*  Provided By
  - Compiler Tyrant
  - Alan G Labouseur
*/
{}$
{{{{{{{}}}}}}}$
{{{{{{{}}}}}}}$
{int  @}$
```

Results: ERROR: Unrecognized or Invalid Token [@] on line 8 col 5

3.2 Invalid String 1

```
/* Test case for placing $ in quotes */
{
    print("i love com$pilers")
    int a
    a = 2
    string s
    s = "ha"
    "
}$
```

Results: ERROR: Invalid character in String [$] on line 3 col 21

3.3 Invalid String 2

```
/* Test case for invalid characters in string */
{
    string s
    s = "cookies & cream"
}$
```

Results: ERROR: Invalid character in String [&] on line 4 col 17

3.4 Invalid String 3

```
/* Test case for placing \n in quotes */
{
    "hey
    there"
}$
```

Results: ERROR: Invalid character in String [\n] on line 3 col 8

3.5 Invalid String 4

```
/* Test case for missing ending quote */
int a
a = 4
string s
s = "hey there
```

Results: ERROR: Missing ending quote for String literal starting on line 5 col 4

3.6 Invalid Print

```
/* Test case for invalid print */
{
    print("my name is 11")
}$
```

Results: ERROR: Invalid character in String [1] on line 3 col 22

3.7 Missing End Comment Brace

```
/* Test case for missing end comment brace */
{
    print("my name is eleven")
    /* hey i love compilers
}$
```

Results: ERROR: Missing ending comment brace (*/) for comment starting on line 4 col 4

4 Parse Fail Programs

4.1 Invalid StatementList

```
/* Test case for invalid StatementList */
{
    4 + 2
}$
```

Results: ERROR - Expecting [TRbrace], found [TDigit] on line 3

4.2 Invalid Expr

```
/* Test case for invalid Expr */
{
    int a
    a = a + 2
}$
```

Results: ERROR - Expecting [TRbrace], found [TDigit] on line 3

4.3 Invalid VarDecl

```
/* Test case for invalid VarDecl */
{
    int 4
}$
```

Results: ERROR - Expecting [Id], found [TDigit] on line 3

4.4 Invalid Print Pt. 2

```
/* Test case for invalid Print pt. 2 */
{
    print("$)
}$
```

Results: ERROR - Expecting [Expr], found [TEop] on line 3
ERROR - Expecting [Block], found [TRparen] on line 3

4.5 Incomplete BooleanExpr

```
/* Test case for incomplete BooleanExpr */
{
    s = "strb"
    print(s)

    if (a != ) {
        print("true")
    }
}$
```

Results: ERROR - Expecting [Expr], found [TRparen] on line 6

4.6 Incomplete IntExpr

```
/* Test case for incomplete IntExpr */
{
    int a
    a = 1 +
    print(a)
}$
```

Results: ERROR - Expecting [Expr], found [TPrint] on line 5

5 Semantic Analysis Fail Programs

5.1 Undeclared Variable

```
/* Variables being used but not declared first */
{
    int a
    b = 4
}$
```

Results: ERROR: Variable [b] on line 4 col 12 has not been previously declared.

5.2 Duplicate Variable

```
/* Variables being declared again in same scope*/
{
    int a
    {
        string a
        a = "this is fine"
    }
    boolean a /* this is not fine" */
}$
```

Results: ERROR: Variable [a] on line 8 col 20 has already been declared in current scope at line 3 col 12

5.3 Type Mismatch

```
/* A variable's type is not compatible with its assignment*/
{
    string s
    s = 4 + 3
}$
```

Results: ERROR: The variable [s] declared on line 4 col 12 is of type string and does not match the assignment type of int

5.4 Incorrect Type Comparisons

```
/* Types do not match in Boolean comparison*/
{
    if(4 == false){
        print("this no good")
    }
    if(4 == "hey"){
        print("int to string")
    }
    if(false != "hey"){
        print("bool to string")
```

```
11              }
12              if(4 != 3){
13                  print("int to int")
14              }
15          }$
```

Results: ERROR: The [Expression] on line 3 col 15 is of type int and is incompatibly compared to a type of boolean

5.5 Incorrect Integer Expression

```
1           /* A digit is added to something other than a digit */
2           {
3               int a
4               a = 4 + false
5           }$
```

Results: ERROR: The [Expression] on line 4 col 20 is of type boolean which cannot be added to digits of type int

5.6 Tien Test

```
1           /* Thx Tien. */
2           {
3               int a
4               a = 0
5               string z
6               z = "bond"
7               while (a != 9) {
8                  if (a != 5) {
9                      print("bond")
10                 }
11                 {
12                     a = 1 + a
13                     string b
14                     b = "james bond"
15                     print(b)
16                 }
17              }
18              {/*Holy Hell This is Disgusting*/}
19              boolean c
20              c = true
21              boolean d
22              d = (true == (true == false))
23              d = (a == b)
24              d = (1 == a)
25              d = (1 != 1)
26              d = ("string" == 1)
27              d = (a != "string")
28              d = ("string" != "string")
29              if (d == true) {
30                  int c
31                  c = 1 + d
32                  if (c == 1) {
33                      print("ugh")
34                  }
35              }
36              while ("string" == a) {
37                  while (1 == true) {
38                      a = 1 + "string"
39                  }
40              }
41          }$
```

Results: ERROR - Variable [b] on line 40 col 22 has not been previously declared.

5.7 Tien Boolean Hell

```
1         /* Thanks Tien. Assuming you get past Boolean Hell
2          - there is a boolean being compared to
3          - a string which will cause a type error */
4         {
5             int a
6             a = 4
7             boolean b
8             b = true
9             boolean c
10            string d
11            d = "there is no spoon"
12            c = (d != "there is a spoon")
13            if(c == (false != (b == (true == (a == 3+1))))) {
14                print((b != d))
15            }
16        }$
```

Results: ERROR - The [Expression] on line 14 col 23 is of type boolean and is incompatibly compared to a type of string