

# Lab 3 Semantic Analysis

Sorin Macaluso

April 7, 2024

## 1 Test Cases

These are a few repeat test cases from the other examples as well as some of the ones from the lab itself. This will only show semantic analysis output since the lexer and the parser have already been established. All test cases were ran individually since there would be too much output to grab just the semantic stuff. Test number 9 is proof that I can work all together.

### 1.1 Test Case 1

---

```
1  /*1 valid*/
2  /*Long Test Case - Everything Except Boolean Declaration */
3  {
4      /* Int Declaration */
5      int a
6      int b
7      a = 0
8      b=0
9      /* While Loop */
10     while (a != 3) {
11         print(a)
12         while (b != 3) {
13             print(b)
14             b = 1 + b
15             if (b == 2) {
16                 /* Print Statement */
17                 print("there is no spoon" /* This will do nothing */ )
18             }
19         }
20         b = 0
21         a = 1+a
22     }
23 }
24 $
25
26 Semantic Analysis starting :)
27 AST for Program 1 Done
28 <block>
29
30 -<var_decl>
31
32 --[int]
33
```

```
34  --[a]
35
36  -<var_decl>
37
38  --[int]
39
40  --[b]
41
42  -<assignment_statement>
43
44  --[a]
45
46  --[0]
47
48  -<assignment_statement>
49
50  --[b]
51
52  --[0]
53
54  -<while_statement>
55
56  --[while]
57
58  --[(]
59
60  --[a]
61
62  --[!=]
63
64  --[3]
65
66  --[)]
67
68  --<block>
69
70  ---<print_statement>
71
72  ----[print]
73
74  ----[a]
75
76  ---<while_statement>
77
78  ----[while]
79
80  ----[(]
81
82  ----[b]
83
84  ----[!=]
85
86  ----[3]
87
```

```

88 ----[)]
89
90 ----<block>
91
92 -----<print_statement>
93
94 -----[print]
95
96 -----[b]
97
98 -----<assignment_statement>
99
100 -----[b]
101
102 -----<+>
103
104 -----[1]
105
106 -----[b]
107
108 -----<if_statement>
109
110 -----[if]
111
112 -----[(]
113
114 -----[b]
115
116 -----[==]
117
118 -----[2]
119
120 -----[)]
121
122 -----<block>
123
124 -----<print_statement>
125
126 -----[print]
127
128 -----["there is no spoon"]
129
130 -<assignment_statement>
131
132 --[b]
133
134 --[0]
135
136 -<assignment_statement>
137
138 --[a]
139
140 --<+>
141

```

```
142 ---[1]
143
144 ---[a]
145
146 -[$]
147
148 Scope 0:
149 Variable Name: a, Type: int, IsUsed: true, IsInitialized: true
150 Variable Name: b, Type: int, IsUsed: true, IsInitialized: true
151 End of program 1
```

---

## 1.2 Test Case 2

---

```
1  /*2 valid*/
2  {
3      int a
4      boolean b
5      {
6          string c
7          a = 5
8          b = true /* no comment */
9          c = "inta"
10         print(c)
11     }
12     print(b)
13     print(a)
14 }$
15
16 Semantic Analysis starting :)
17 AST for Program 1 Done
18 <block>
19
20 -<var_decl>
21
22 --[int]
23
24 --[a]
25
26 -<var_decl>
27
28 --[boolean]
29
30 --[b]
31
32 -<block>
33
34 --<var_decl>
35
36 ---[string]
37
38 ---[c]
39
40 --<assignment_statement>
```

```

41
42 ---[a]
43
44 ---[5]
45
46 --<assignment_statement>
47
48 ---[b]
49
50 ---[true]
51
52 --<assignment_statement>
53
54 ---[c]
55
56 ---["inta"]
57
58 --<print_statement>
59
60 ---[print]
61
62 ---[c]
63
64 -<print_statement>
65
66 --[print]
67
68 --[b]
69
70 -<print_statement>
71
72 --[print]
73
74 --[a]
75
76 -[$]
77
78 Scope 0:
79 Variable Name: a, Type: int, IsUsed: true, IsInitialized: true
80 Variable Name: b, Type: boolean, IsUsed: true, IsInitialized: true
81 Scope 1:
82 Variable Name: c, Type: string, IsUsed: true, IsInitialized: true
83 End of program 1

```

---

### 1.3 Test Case 3

```

1 /*3 invalid (b not in scope)*/
2 {
3     int a
4     {
5         boolean b
6         a = 1
7     }

```

```

8      print(b)
9  }$
10
11  Semantic Analysis starting :)
12  AST for Program 1 Done
13  <block>
14
15  -<var_decl>
16
17  --[int]
18
19  --[a]
20
21  -<block>
22
23  --<var_decl>
24
25  ---[boolean]
26
27  ---[b]
28
29  --<assignment_statement>
30
31  ---[a]
32
33  ---[1]
34
35  -<print_statement>
36
37  --[print]
38
39  --[b]
40
41  -[$]
42
43  Variable b line num 8 used in print statement not found.
44  Scope 0:
45  Variable Name: a, Type: int, IsUsed: true, IsInitialized: true
46  End of program 1

```

---

## 1.4 Test Case 4

---

```

1  /*4 valid proof of scope maneuvering*/
2  {
3  int a
4  {
5      boolean b
6      {
7          string c
8          {
9              a = 5
10             b = false
11             c = "inta"

```

```

12         }
13         print(c)
14     }
15     print(b)
16 }
17 print(a)
18 }$
19
20 Semantic Analysis starting :)
21 AST for Program 1 Done
22 <block>
23
24 -<var_decl>
25
26 --[int]
27
28 --[a]
29
30 -<block>
31
32 --<var_decl>
33
34 ---[boolean]
35
36 ---[b]
37
38 --<block>
39
40 ---<var_decl>
41
42 ----[string]
43
44 ----[c]
45
46 ---<block>
47
48 ----<assignment_statement>
49
50 -----[a]
51
52 -----[5]
53
54 ----<assignment_statement>
55
56 -----[b]
57
58 -----[false]
59
60 ----<assignment_statement>
61
62 -----[c]
63
64 -----["inta"]
65

```

```

66 -<print_statement>
67
68 --[print]
69
70 --[c]
71
72 -<print_statement>
73
74 --[print]
75
76 --[b]
77
78 -<print_statement>
79
80 --[print]
81
82 --[a]
83
84 -[$]
85
86 Scope 0:
87 Variable Name: a, Type: int, IsUsed: true, IsInitialized: true
88 Scope 1:
89 Variable Name: b, Type: boolean, IsUsed: true, IsInitialized: true
90 Scope 2:
91 Variable Name: c, Type: string, IsUsed: true, IsInitialized: true
92 End of program 1

```

---

## 1.5 Test Case 5

---

```

1  /*5 valid*/
2  {
3      boolean a
4      a = (1 == (2 == 3))
5  }$
6
7  Semantic Analysis starting :)
8  AST for Program 1 Done
9  <block>
10
11 -<var_decl>
12
13 --[boolean]
14
15 --[a]
16
17 -<assignment_statement>
18
19 --[a]
20
21 --[(]
22
23 --[1]

```



```

24
25  --[==]
26
27  --[(]
28
29  --[2]
30
31  --[==]
32
33  --[3]
34
35  --[]]
36
37  --[]]
38
39  -[$]
40
41  Scope 0:
42  Variable Name: a, Type: boolean, IsUsed: true, IsInitialized: true
43  End of program 1

```

---

## 1.6 Test Case 6

---

```

1  /*6 invalid type mismatch error*/
2  {
3      string a
4      a = 2
5  }$
6
7  Semantic Analysis starting :)
8  AST for Program 1 Done
9  <block>
10
11  -<var_decl>
12
13  --[string]
14
15  --[a]
16
17  -<assignment_statement>
18
19  --[a]
20
21  --[2]
22
23  -[$]
24
25  Type mis-match error at 4 at 9 declared string but comparing int.
26  Scope 0:
27  Variable Name: a, Type: string, IsUsed: true, IsInitialized: false
28  End of program 1

```

---

## 1.7 Test Case 7

---

```
1  /*7 valid (checking for boolop stuff inside while)*/
2  {
3      while((1 == (2 == 3)) != 1){
4          print("hello")
5      }
6  }$
7
8  Semantic Analysis starting :)
9  AST for Program 1 Done
10 <block>
11
12 -<while_statment>
13
14 --[while]
15
16 --[(]
17
18 --[(]
19
20 --[1]
21
22 --[==]
23
24 --[(]
25
26 --[2]
27
28 --[==]
29
30 --[3]
31
32 --[)]
33
34 --[)]
35
36 --[!=]
37
38 --[1]
39
40 --[)]
41
42 --<block>
43
44 ---<print_statment>
45
46 ----[print]
47
48 ----["hello"]
49
50 -[$]
51
52 End of program 1
```

```
53
54 /*no print out since there are no varabils in this program*/
```

---

## 1.8 Test Case 8

---

```
1 /*8 valid (checking for boolop stuff inside if)*/
2 {
3     if((1 == (2 == 3)) != 1){
4         print("hello")
5     }
6 }$
7
8 Semantic Analysis starting :)
9 AST for Program 1 Done
10 <block>
11
12 -<if_statment>
13
14 --[if]
15
16 --[(]
17
18 --[(]
19
20 --[1]
21
22 --[==]
23
24 --[(]
25
26 --[2]
27
28 --[==]
29
30 --[3]
31
32 --[)]
33
34 --[)]
35
36 --[!=]
37
38 --[1]
39
40 --[)]
41
42 --<block>
43
44 ---<print_statment>
45
46 ----[print]
47
48 ----["hello"]
```

```
49
50 -[$]
51
52 End of program 1
53
54 /*no print out since there are no varabils in this program*/
```

---

## 1.9 Test Case 9

---

```
1 {
2     int a
3     a = 3
4 }$
5
6 {
7     int a
8     a = 4
9 }$
10
11 {
12     int a
13     a = 5
14 }$
15
16 Semantic Analysis starting :)
17 AST for Program 1 Done
18 <block>
19
20 -<var_decl>
21
22 --[int]
23
24 --[a]
25
26 -<assignment_statment>
27
28 --[a]
29
30 --[3]
31
32 -[$]
33
34 Scope 0:
35 Variable Name: a, Type: int, IsUsed: true, IsInitialized: true
36 End of program 1
37
38 Semantic Analysis starting :)
39 AST for Program 2 Done
40 <block>
41
42 -<var_decl>
43
44 --[int]
```

```

45
46 --[a]
47
48 -<assignment_statment>
49
50 --[a]
51
52 --[4]
53
54 -[$]
55
56 Scope 0:
57 Variable Name: a, Type: int, IsUsed: true, IsInitialized: true
58 End of program 2
59
60 Semantic Analysis starting :)
61 AST for Program 3 Done
62 <block>
63
64 -<var_decl>
65
66 --[int]
67
68 --[a]
69
70 -<assignment_statment>
71
72 --[a]
73
74 --[5]
75
76 -[$]
77
78 Scope 0:
79 Variable Name: a, Type: int, IsUsed: true, IsInitialized: true
80 End of program 3

```

---

## 2 References

[Java Regular Expression](#)  
[Enum](#)  
[Switch cases \(Was more of an idea\)](#)  
[Double Quote For Regular Expressions](#)  
[Groups in Regular Expressions](#)  
[More Group stuff](#)  
[Index for Regular Expressions](#)  
[Check for end of line](#)  
[Full Syntax Java Regx](#)  
[Quote In Regx](#)

Links that were already used/shown in previous lab reports. For semantic analysis I was either using the stuff that was mentioned in the slides or having to build my own thing completely. Especially the semantic

analysis, traversing through that is something that I completely made on my own.