

JLhs.java

```
1  // Copyright 2013 Bill Campbell, Swami Iyer and Bahar Akbal-Delibas
2
3  package jminusminus;
4
5  /**
6   * The type of any expression that can appear on the lhs of an assignment
7   * statement, i.e., JVariable, JFieldSelection, JArrayExpression.
8   */
9
10 interface JLhs {
11
12     /**
13      * Analyze the lhs of an assignment. This is very much like analyze() but
14      * perhaps a little more selective here and there.
15      *
16      * @param context
17      *         context in which names are resolved.
18      * @return the analyzed (and possibly rewritten) AST subtree.
19      */
20
21     public JExpression analyzeLhs(Context context);
22
23     /**
24      * The up front code necessary for implementing an assignment; it generates
25      * code to load onto the stack any part of the lhs variable that must be
26      * there. For example, in a[i] = x, code must be generated to load the array
27      * a) and the index i.
28      *
29      * @param output
30      *         the code emitter (basically an abstraction for producing the
31      *         class file).
32      */
33
34     public void codegenLoadLhsLvalue(CLEmitter output);
35
36     /**
37      * Generate the code required for loading an Rvalue for this variable, as in
38      * a +=.
39      *
40      * @param output
41      *         the code emitter (basically an abstraction for producing the
42      *         .class file).
43      */
44
45     public void codegenLoadLhsRvalue(CLEmitter output);
46
47     /**
48      * Generate the code required for duplicating the Rvalue that is on the
49      * stack because it is to be used in a surrounding expression, as in a[i] =
50      * x = <expr> or x = y--.
51      *
52      * @param output
53      *         the code emitter (basically an abstraction for producing the
54      *         .class file).
55      */
56
57     public void codegenDuplicateRvalue(CLEmitter output);
58
59     /**
60      * Generate the code required for doing the actual assignment.
61      *
62      * @param output
63      *         the code emitter (basically an abstraction for producing the
64      *         .class file).
65      */
66 }
```

```
67     public void codegenStore(CLEmitter output);  
68  
69 }  
70
```

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder