BSc (Hons) in Information Technology
Year 3

Lab Exercise – Cyclomatic Complexity

SE3010 – SEPQM **Semester 1**

The objective of this lab is to learn how to calculate cyclomatic complexity (CC) by examining the source code and byte code.

Warning : Certain web pages do not compute CC correctly. If in doubt, ask the lecturer or tutor.

Question 1

Briefly explain what is CC and its usage?

Question 2

Draw control flow graphs and calculate the CC values of the following methods:

Method	Source File	Class File
public void recQuickSort(int left, int right)	quickSort1.java	ArrayIns
public void setCurrentValue(float val)	SpreadSheet.java	SpreadSheet
public void bubbleSort( )	bubbleSort.java	ArrayBub
public float evaluateFormula(Node n)	SpreadSheet.java	SpreadSheet

Question 3

The disassembled codes of the ***public void recQuickSort(int left, int right)***, ***public void setCurrentValue(float val), public void bubbleSort( )***, *and* ***public float evaluateFormula(Node n)*** methods are given below. Calculate the CC value of them and compare those with the ones derived in the previous question.

Note:

- To compile all the Java applications in a folder, type:

```
[user@comp]$ javac *.java
```

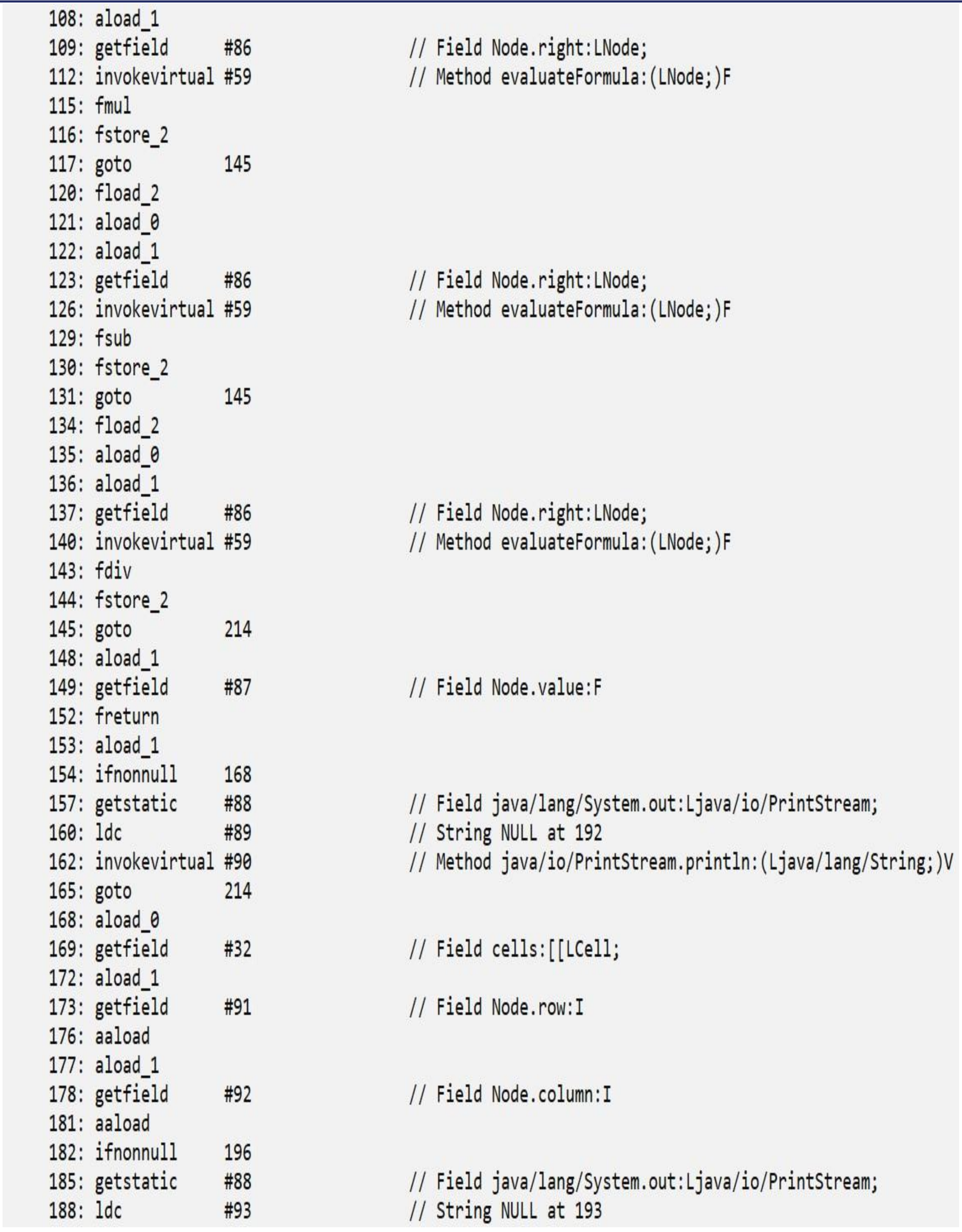
- To disassemble the bytecode, type:

```
[user@comp]$ javap –c ClassFileName
```

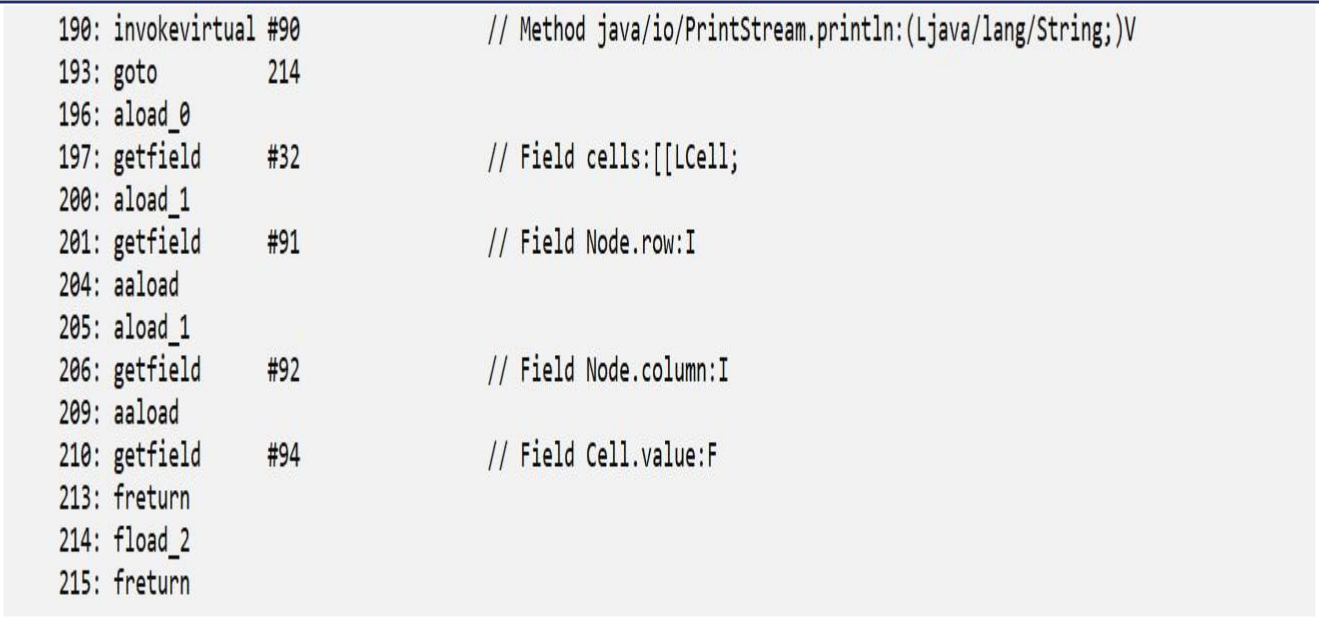
```
public void recQuickSort(int, int);
  Code:
     0: iload_2
     1: iload_1
     2: isub
     3: ifgt          7
     6: return
     7: aload_0
     8: getfield      #2                  // Field theArray:[D
    11: iload_2
    12: daload
    13: dstore_3
    14: aload_0
    15: iload_1
    16: iload_2
    17: dload_3
    18: invokevirtual #11                 // Method partitionIt:(IID)I
    21: istore        5
    23: aload_0
    24: iload_1
    25: iload         5
    27: iconst_1
    28: isub
    29: invokevirtual #10                 // Method recQuickSort:(II)V
    32: aload_0
    33: iload         5
    35: iconst_1
    36: iadd
    37: iload_2
    38: invokevirtual #10                 // Method recQuickSort:(II)V
    41: return
```

```
public void setCurrentValue(float);
  Code:
     0: aload_0
     1: getfield      #9                  // Field selectedRow:I
     4: iconst_m1
     5: if_icmpeq     16
     8: aload_0
     9: getfield      #10                 // Field selectedColumn:I
    12: iconst_m1
    13: if_icmpne     17
    16: return
    17: aload_0
    18: getfield      #32                 // Field cells:[[LCell;
    21: aload_0
    22: getfield      #9                  // Field selectedRow:I
    25: aaload
    26: aload_0
    27: getfield      #10                 // Field selectedColumn:I
    30: aaload
    31: fload_1
    32: invokevirtual #96                 // Method Cell.setValue:(F)V
    35: aload_0
    36: invokevirtual #53                 // Method repaint:()V
    39: return
  public void bubbleSort();
    Code:
       0: aload_0
       1: getfield      #3                  // Field nElems:I
       4: iconst_1
       5: isub
       6: istore_1
       7: iload_1
       8: iconst_1
       9: if_icmple     57
      12: iconst_0
      13: istore_2
      14: iload_2
      15: iload_1
      16: if_icmpge     51
      19: aload_0
      20: getfield      #2                  // Field a:[D
      23: iload_2
      24: daload
      25: aload_0
      26: getfield      #2                  // Field a:[D
      29: iload_2
      30: iconst_1
      31: iadd
      32: daload
      33: dcmpl
      34: ifle          45
      37: aload_0
      38: iload_2
      39: iload_2
      40: iconst_1
      41: iadd
      42: invokevirtual #9                  // Method swap:(II)V
      45: iinc          2, 1
      48: goto          14
      51: iinc          1, -1
      54: goto          7
      57: return
}
```

```
public float evaluateFormula(Node);
  Code:
     0: fconst_0
     1: fstore_2
     2: aload_1
     3: ifnonnull     8
     6: fload_2
     7: freturn
     8: aload_1
     9: getfield      #83                 // Field Node.type:I
    12: tableswitch   { // 0 to 2
                   0: 40
                   1: 148
                   2: 153
             default: 214
        }
    40: aload_0
    41: aload_1
    42: getfield      #84                 // Field Node.left:LNode;
    45: invokevirtual #59                 // Method evaluateFormula:(LNode;)F
    48: fstore_2
    49: aload_1
    50: getfield      #85                 // Field Node.op:C
    53: tableswitch   { // 42 to 47
                  42: 106
                  43: 92
                  44: 145
                  45: 120
                  46: 145
                  47: 134
             default: 145
        }
    92: fload_2
    93: aload_0
    94: aload_1
    95: getfield      #86                 // Field Node.right:LNode;
    98: invokevirtual #59                 // Method evaluateFormula:(LNode;)F
   101: fadd
   102: fstore_2
   103: goto          145
   106: fload_2
   107: aload_0
```



```
108: aload_1
109: getfield      #86                 // Field Node.right:LNode;
112: invokevirtual #59                 // Method evaluateFormula:(LNode;)F
115: fmul
116: fstore_2
117: goto          145
120: fload_2
121: aload_0
122: aload_1
123: getfield      #86                 // Field Node.right:LNode;
126: invokevirtual #59                 // Method evaluateFormula:(LNode;)F
129: fsub
130: fstore_2
131: goto          145
134: fload_2
135: aload_0
136: aload_1
137: getfield      #86                 // Field Node.right:LNode;
140: invokevirtual #59                 // Method evaluateFormula:(LNode;)F
143: fdiv
144: fstore_2
145: goto          214
148: aload_1
149: getfield      #87                 // Field Node.value:F
152: freturn
153: aload_1
154: ifnonnull     168
157: getstatic     #88                 // Field java/lang/System.out:Ljava/io/PrintStream;
160: ldc           #89                 // String NULL at 192
162: invokevirtual #90                 // Method java/io/PrintStream.println:(Ljava/lang/String;)V
165: goto          214
168: aload_0
169: getfield      #32                 // Field cells:[[LCell;
172: aload_1
173: getfield      #91                 // Field Node.row:I
176: aaload
177: aload_1
178: getfield      #92                 // Field Node.column:I
181: aaload
182: ifnonnull     196
185: getstatic     #88                 // Field java/lang/System.out:Ljava/io/PrintStream;
188: ldc           #93                 // String NULL at 193
```



```
190: invokevirtual #90                 // Method java/io/PrintStream.println:(Ljava/lang/String;)V
193: goto          214
196: aload_0
197: getfield      #32                 // Field cells:[[LCell;
200: aload_1
201: getfield      #91                 // Field Node.row:I
204: aaload
205: aload_1
206: getfield      #92                 // Field Node.column:I
209: aaload
210: getfield      #94                 // Field Cell.value:F
213: freturn
214: fload_2
215: freturn
```

Question 4

Explain why ***public void setCurrentValue(float val)*** and ***public float evaluateFormula(Node n)*** methods are reporting different values for source and byte codes.