# CS361 Algorithm Lab 4

# What to do

1. Write a program to implement the following DFA.
   1. Q = {s, q1, q2, r1, r2}
   2. s is the start state
   3. A = {q1,r1} are the accept states
   4. Σ = {a,b}
   5. σ is defined by the following table:

|  |  |  |
| --- | --- | --- |
|  | a | b |
| s | q1 | r1 |
| q1 | q1 | q2 |
| q2 | q1 | q2 |
| r1 | r2 | r1 |
| r2 | r2 | r1 |

Show me the output (accepting the string or not) for the following strings:

**I keep getting a** java.lang.NullPointerException at DFA.transition(DFA.java:86) at DFA.main(DFA.java:125) **And I don’t know why, but I am getting a result for 1 3 and 5… Guess it only does the odds…**

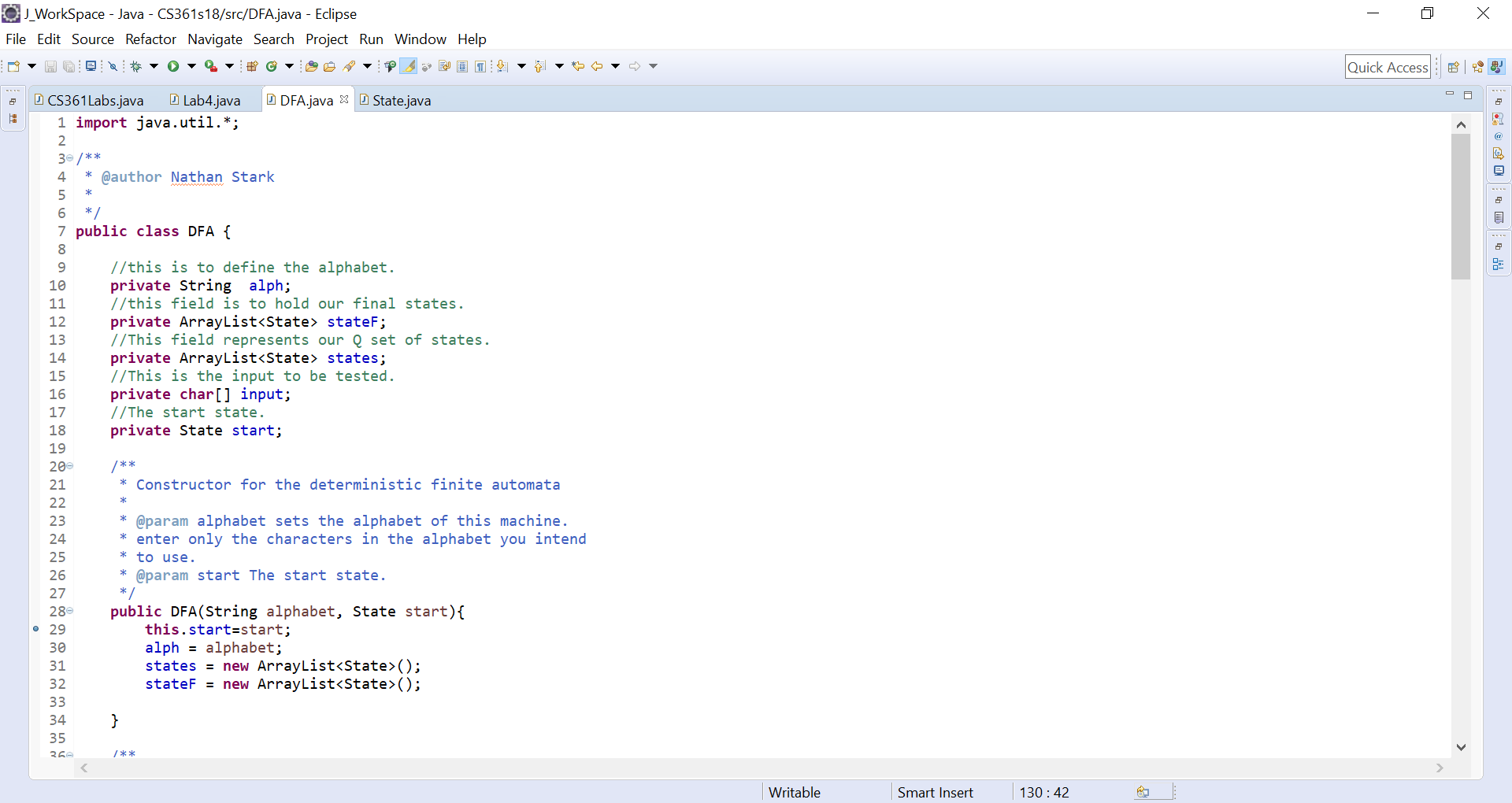
1. Ababa

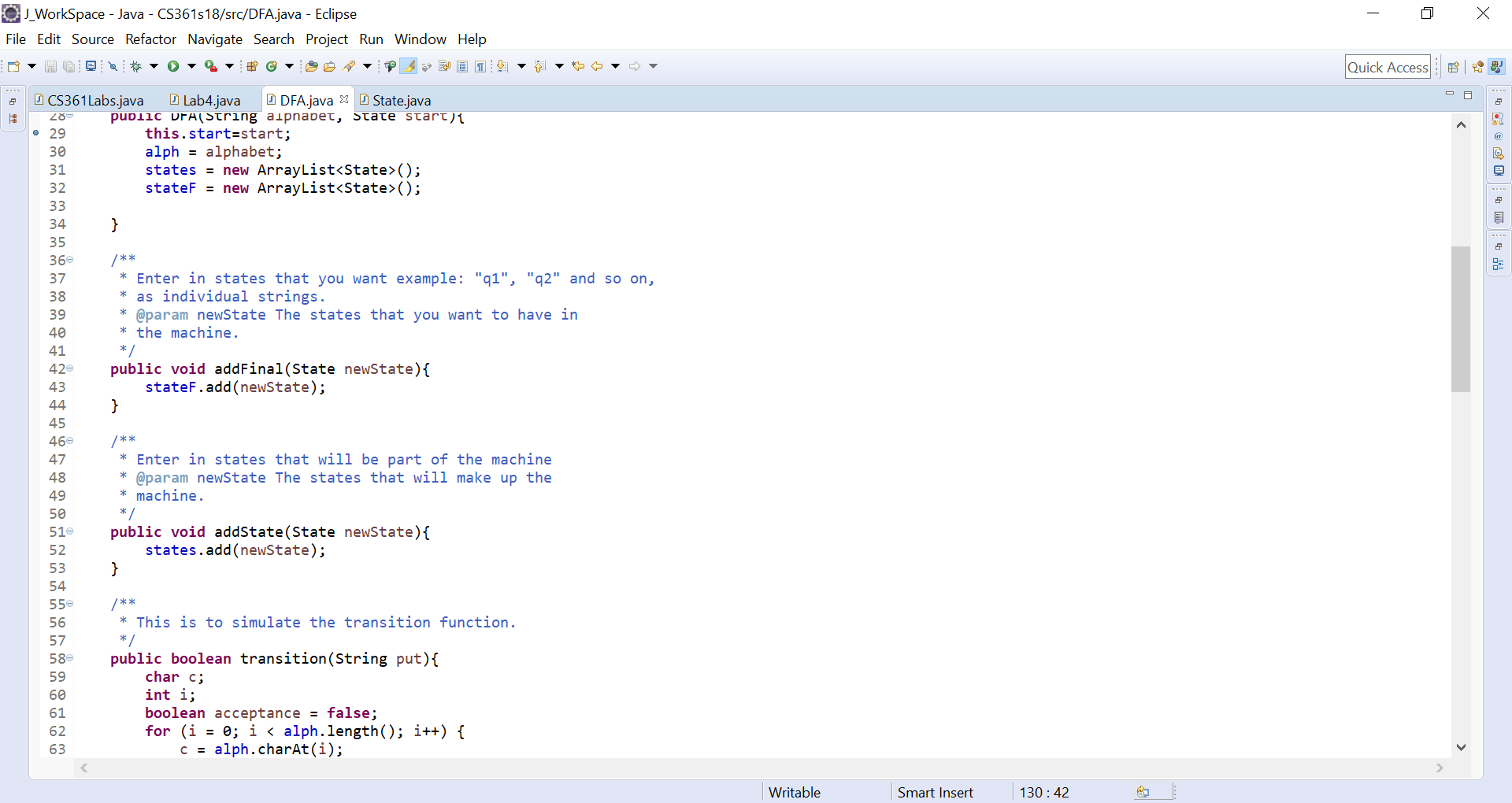
1. Accepted string: ababa

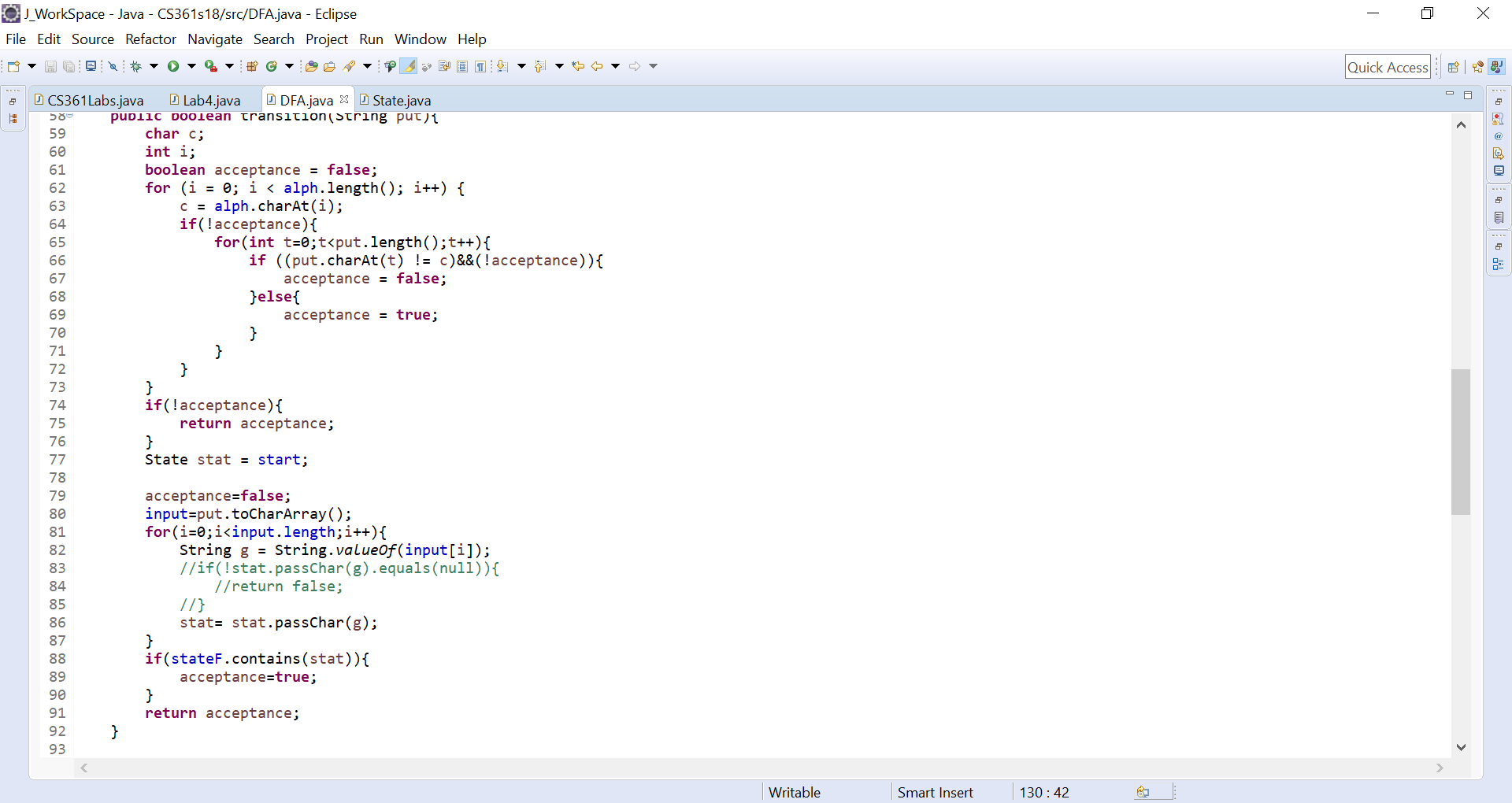
1. baba // This should be not accepted.
2. aababaab

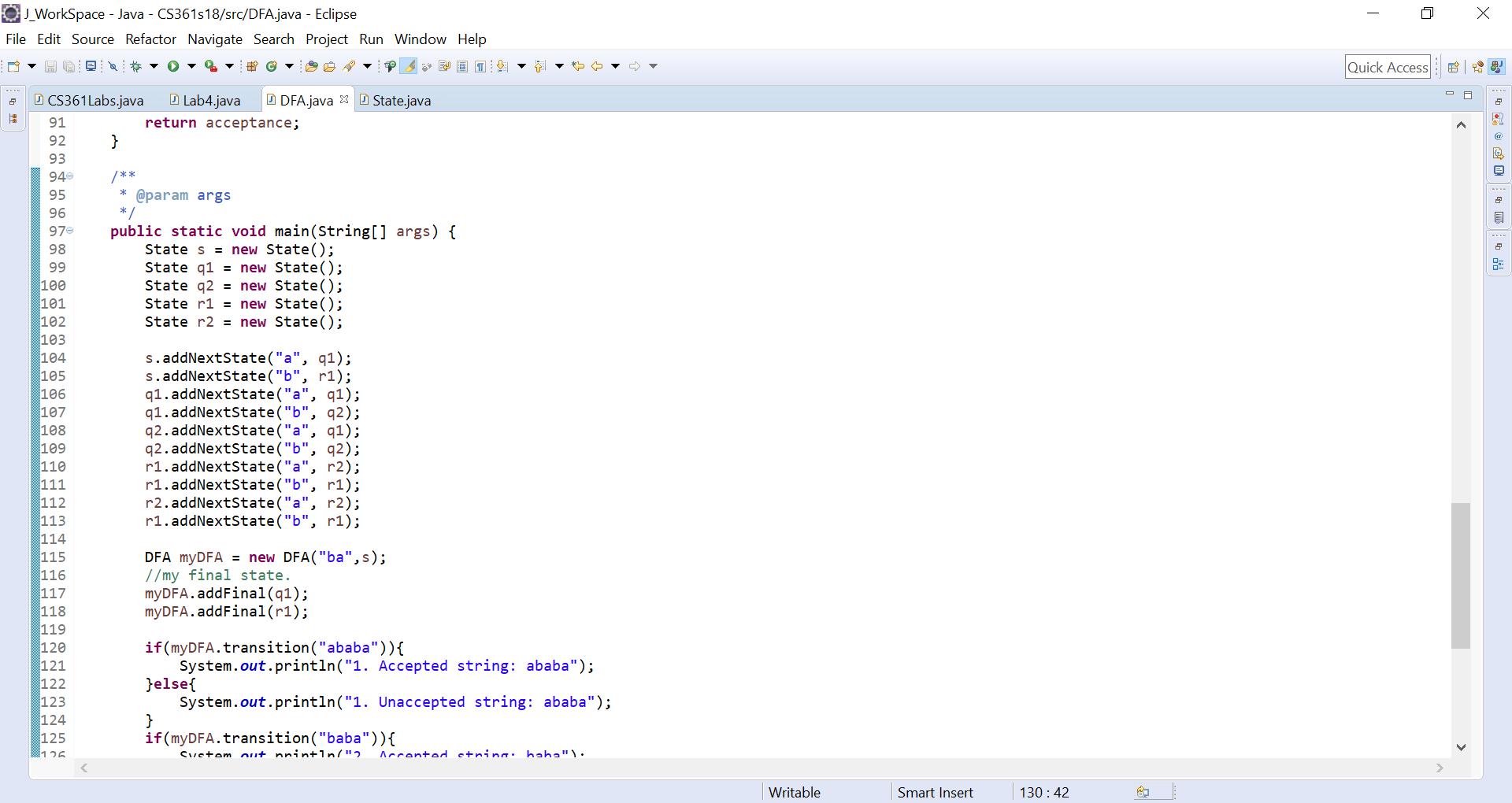
3. Unaccepted string: aababaab

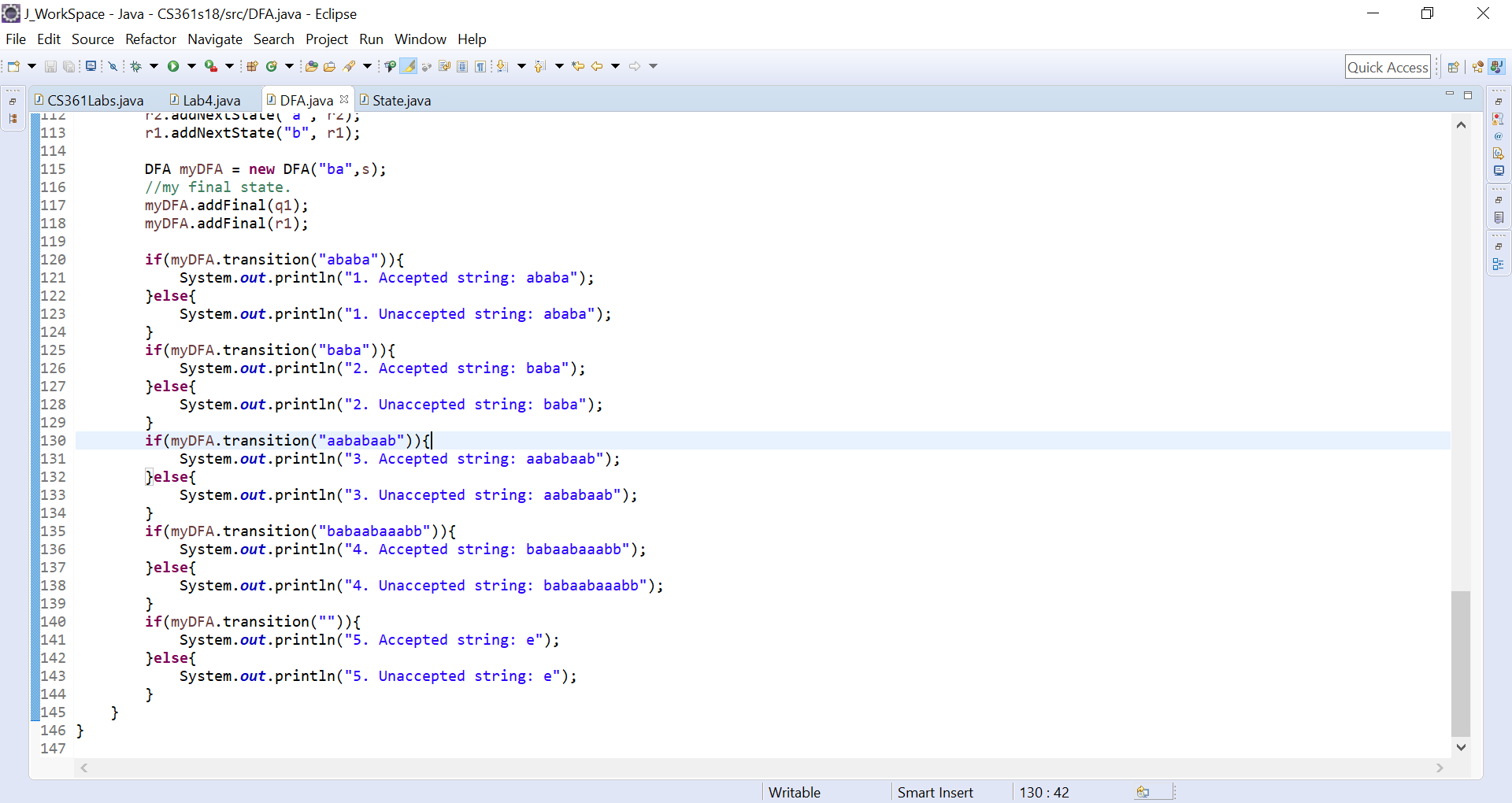
1. babaabaaabb // This should be accepted.
2. ε (the empty string)
3. Unaccepted string: e

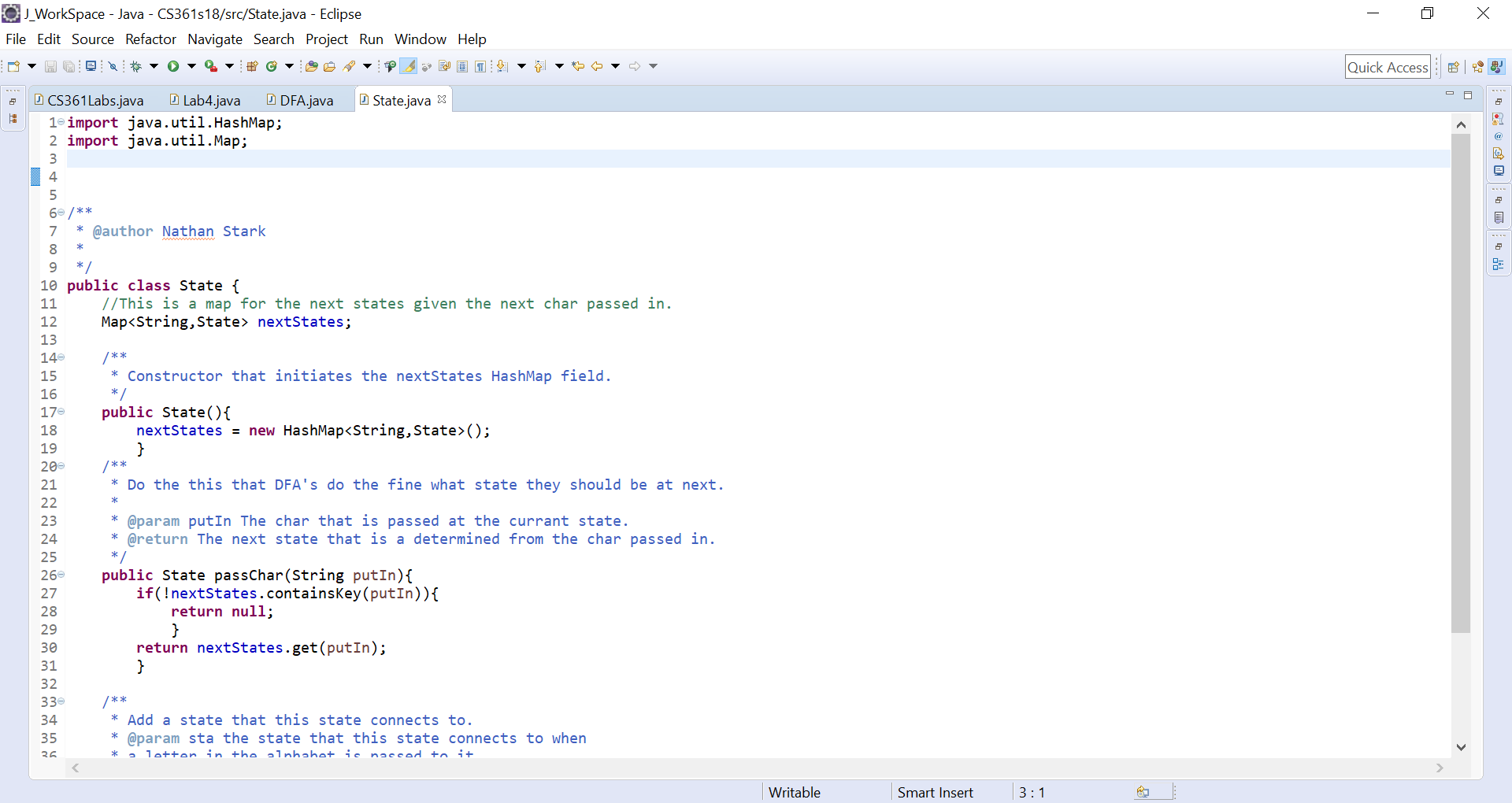


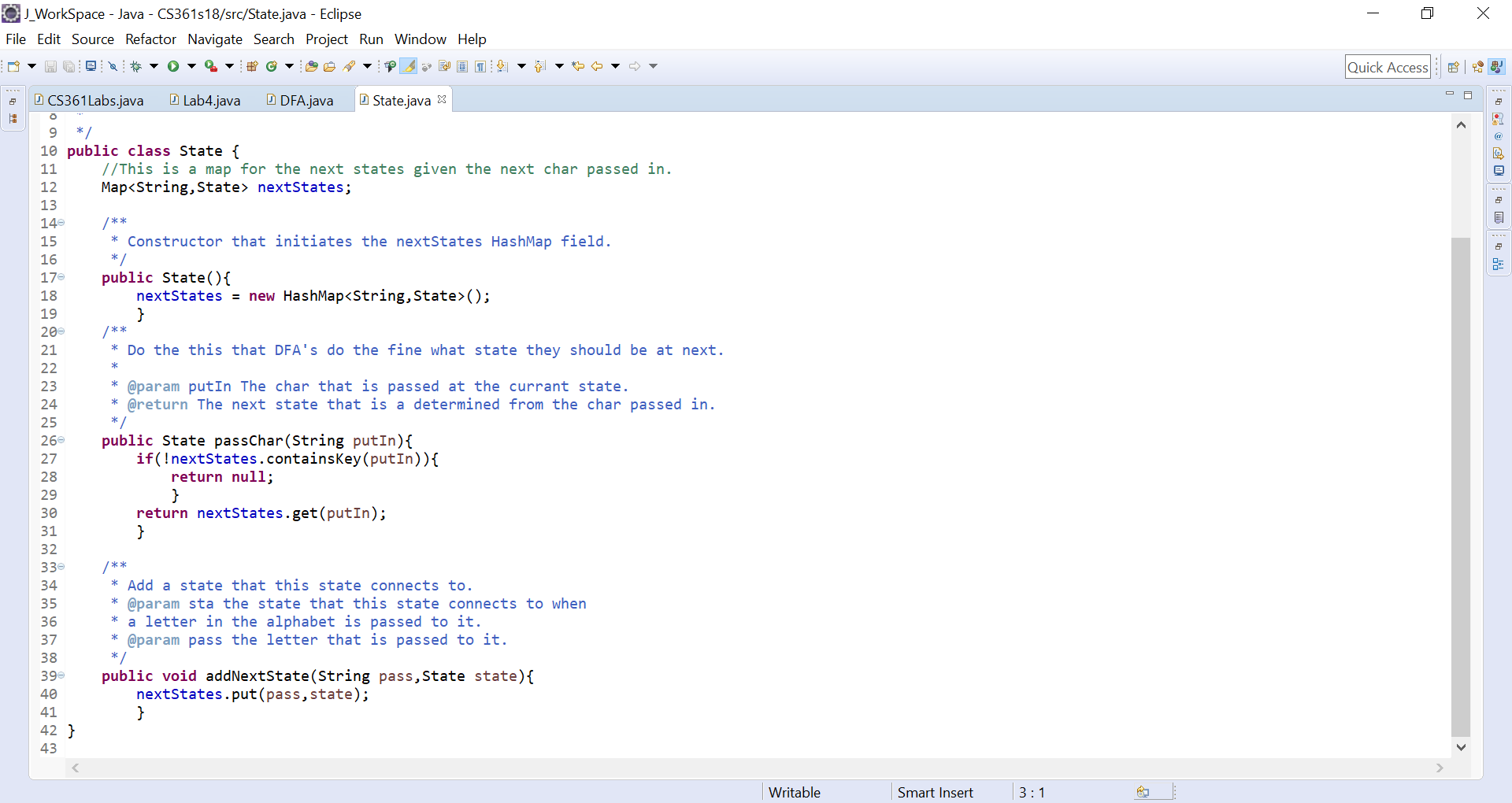




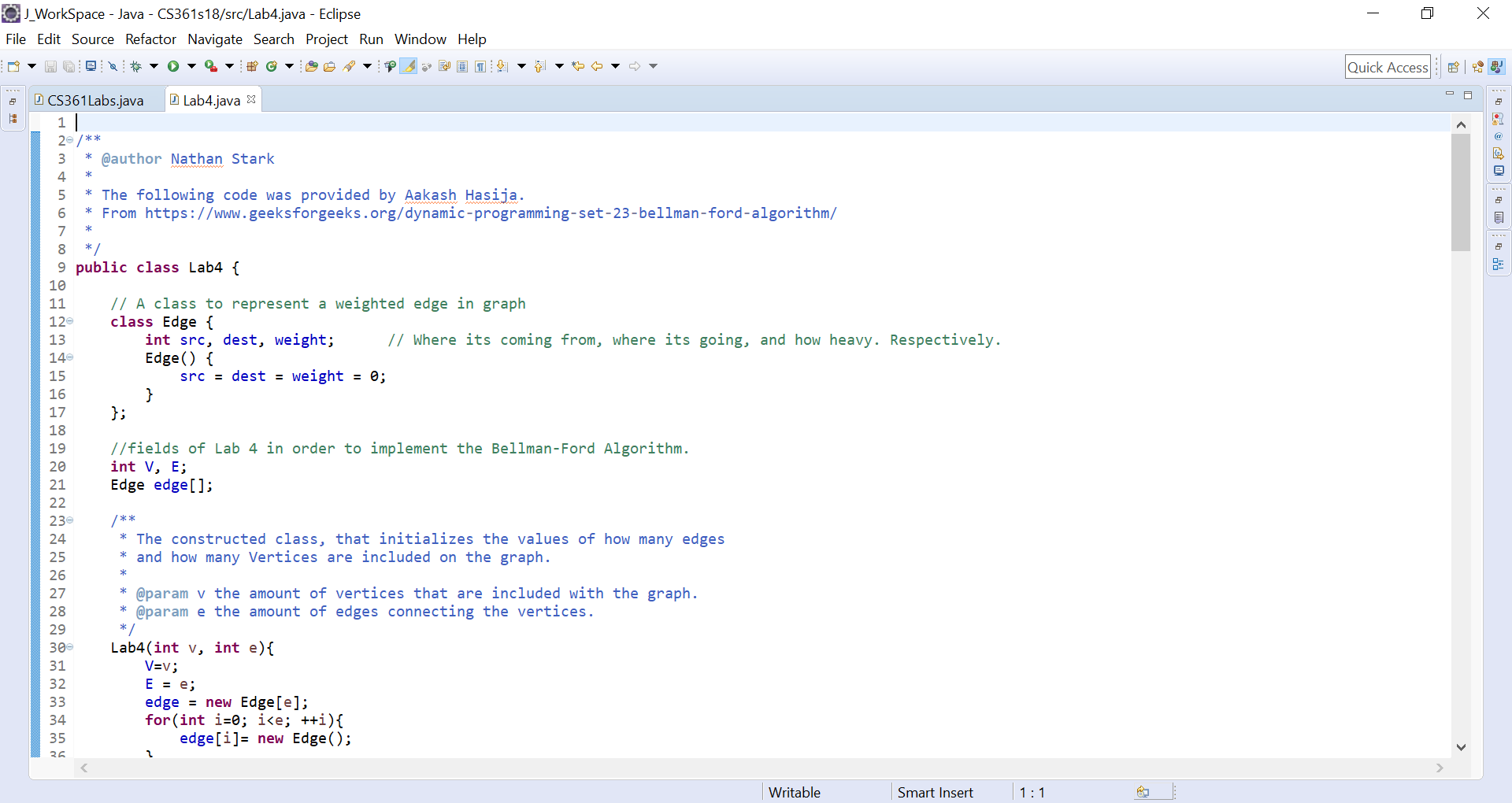


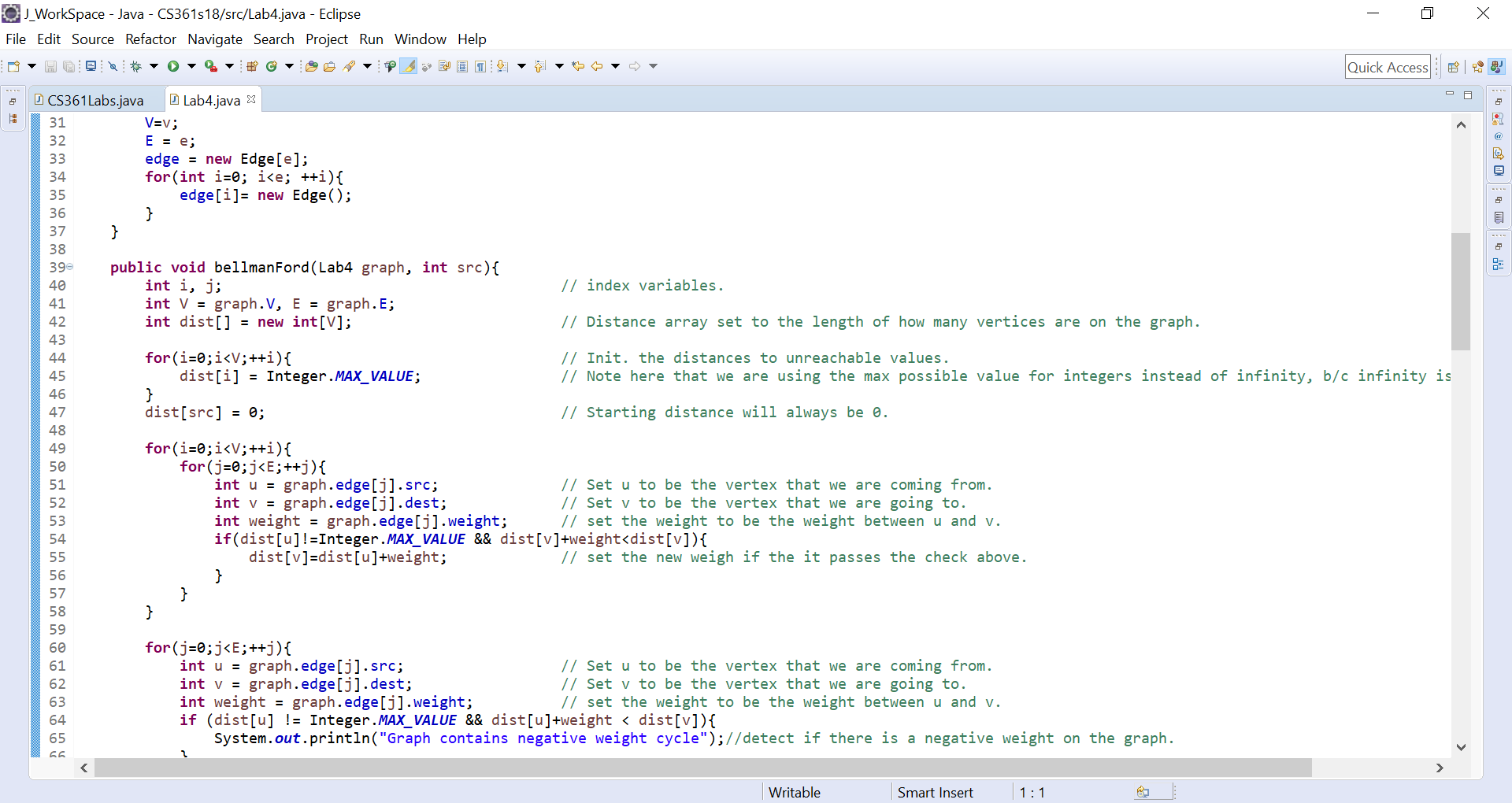


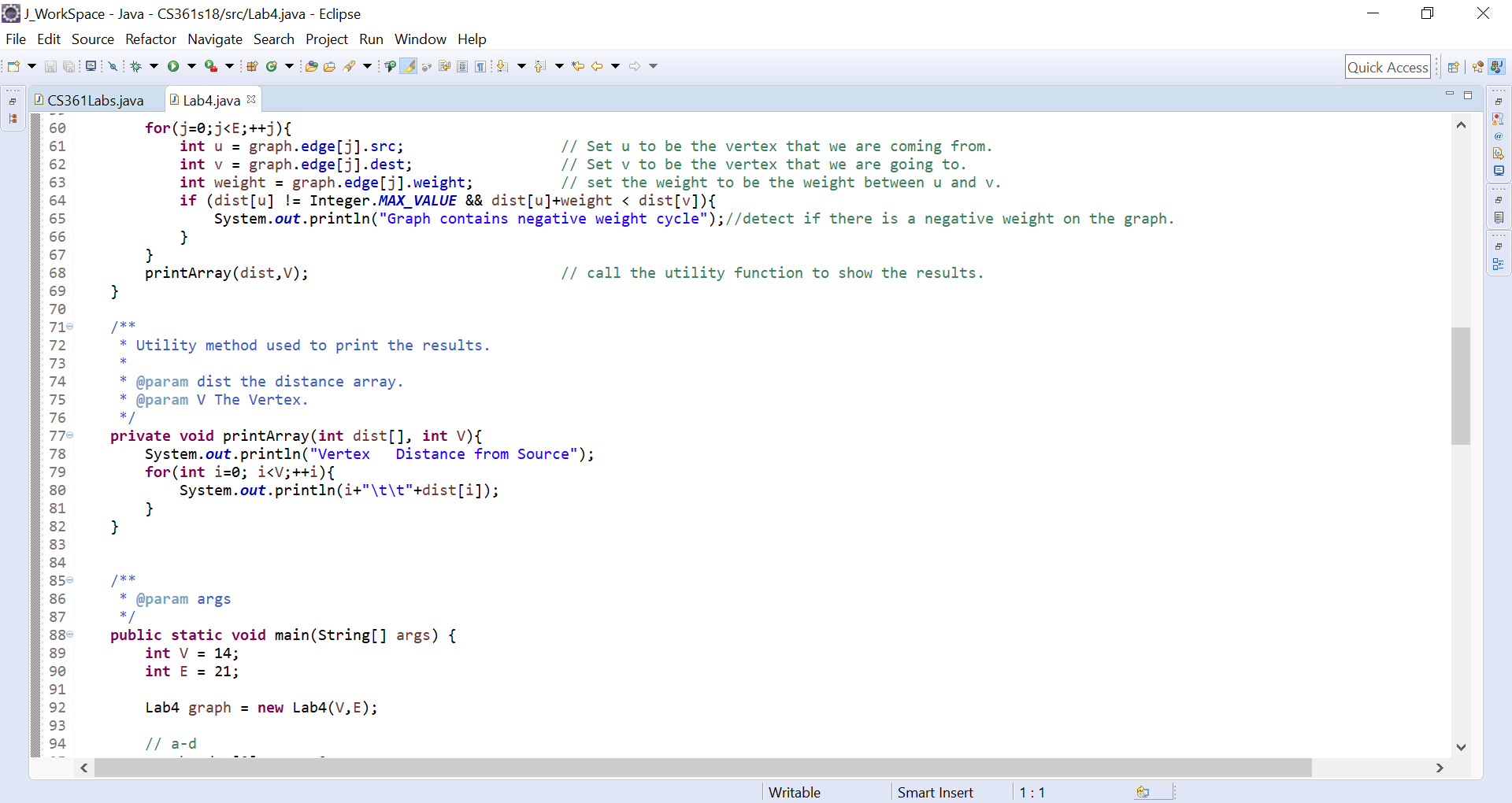




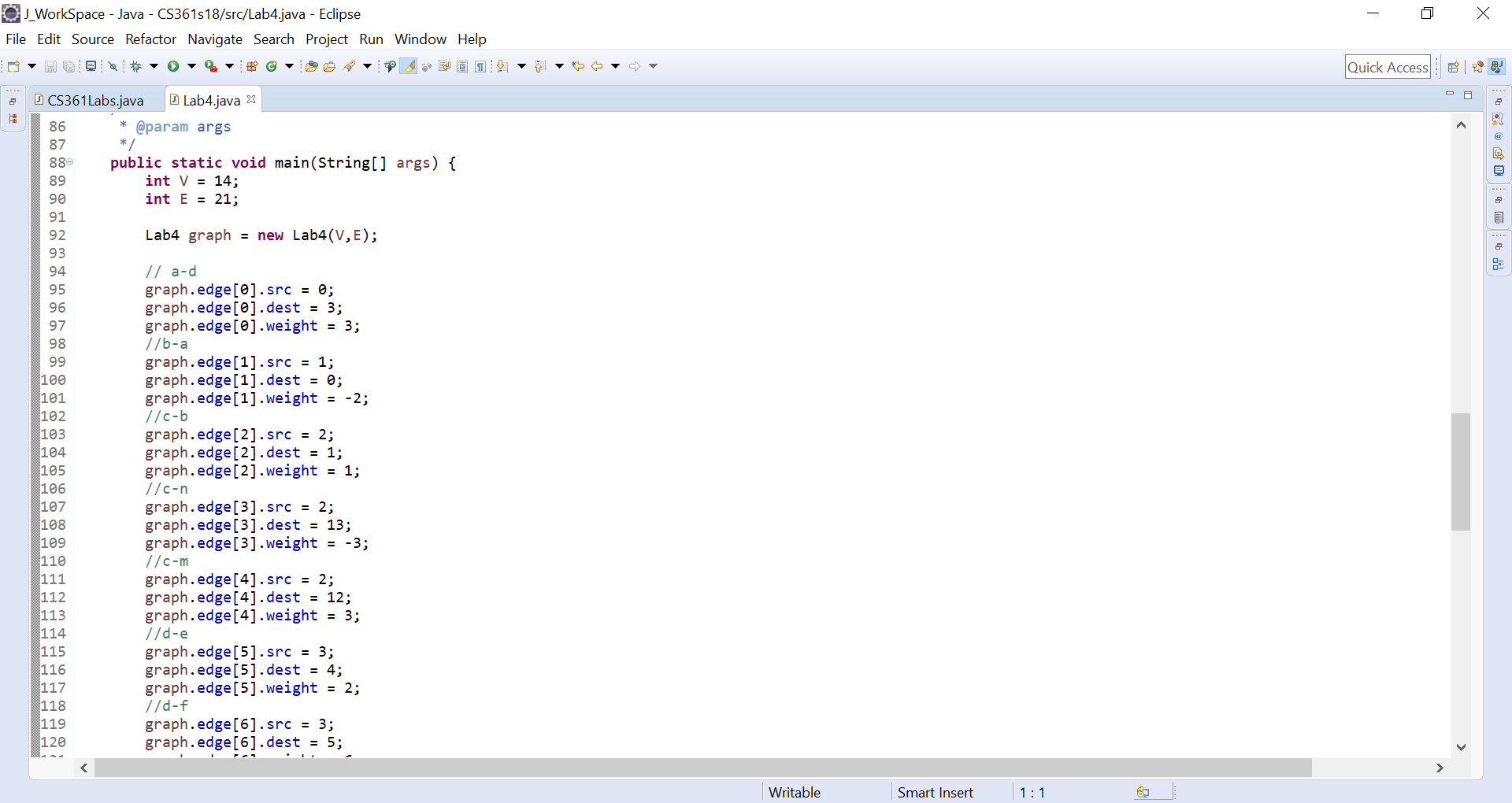
1. Implement the Bellman-Ford algorithm. Show commented code.

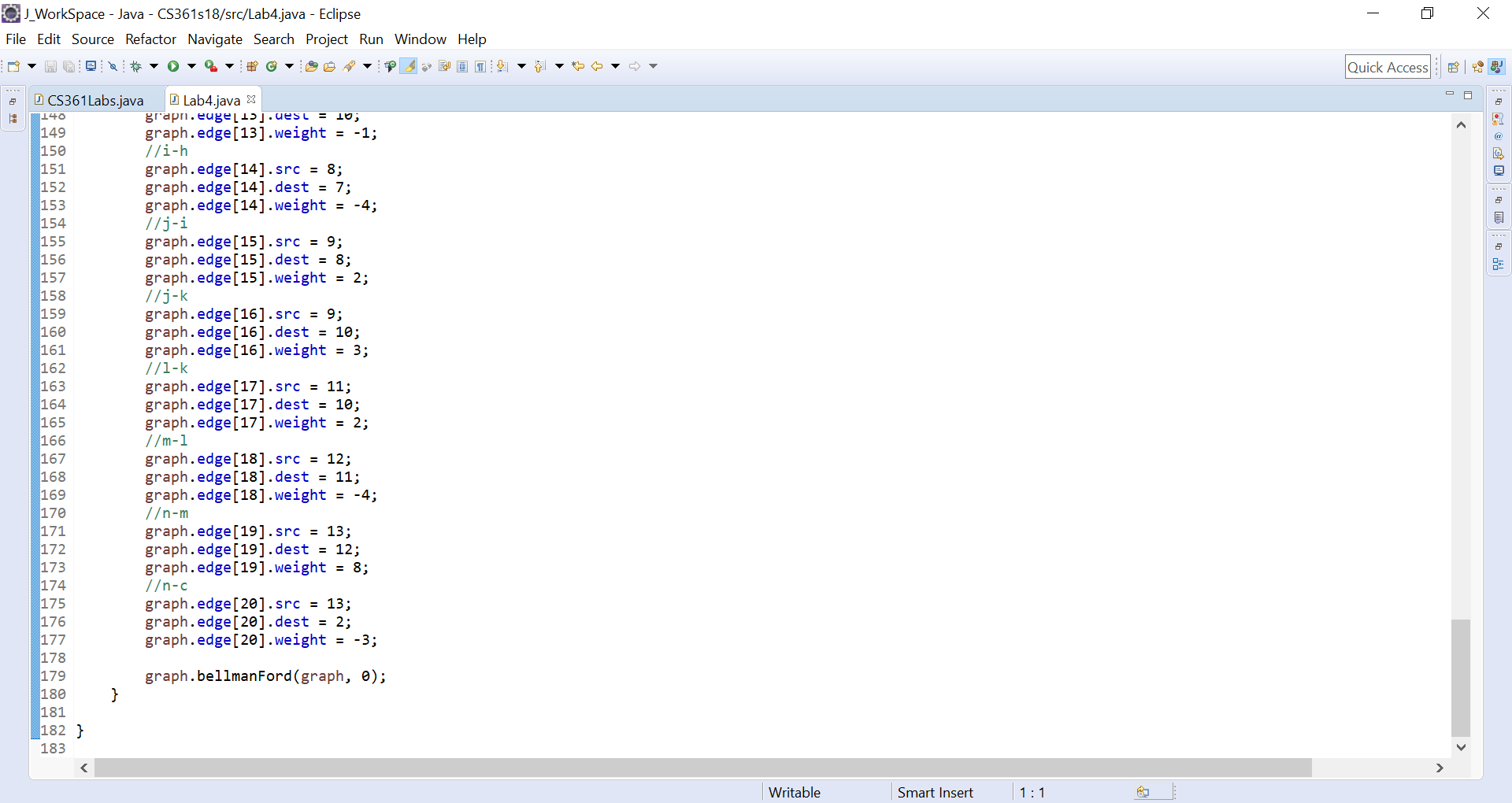






**I’m going to omit most of the main method but there should be enough to understand the basic jest of what is happening.**





1. Show the output (including all of the distances and predecessors) for the Bellman-Ford algorithm on the graph below.

../labgraph2.pdf

**Note that I have started with the letter a and replaced it with the number 0 and have do so with all of the respective letters as well (1 for b 2 for c and so on up to n).**

