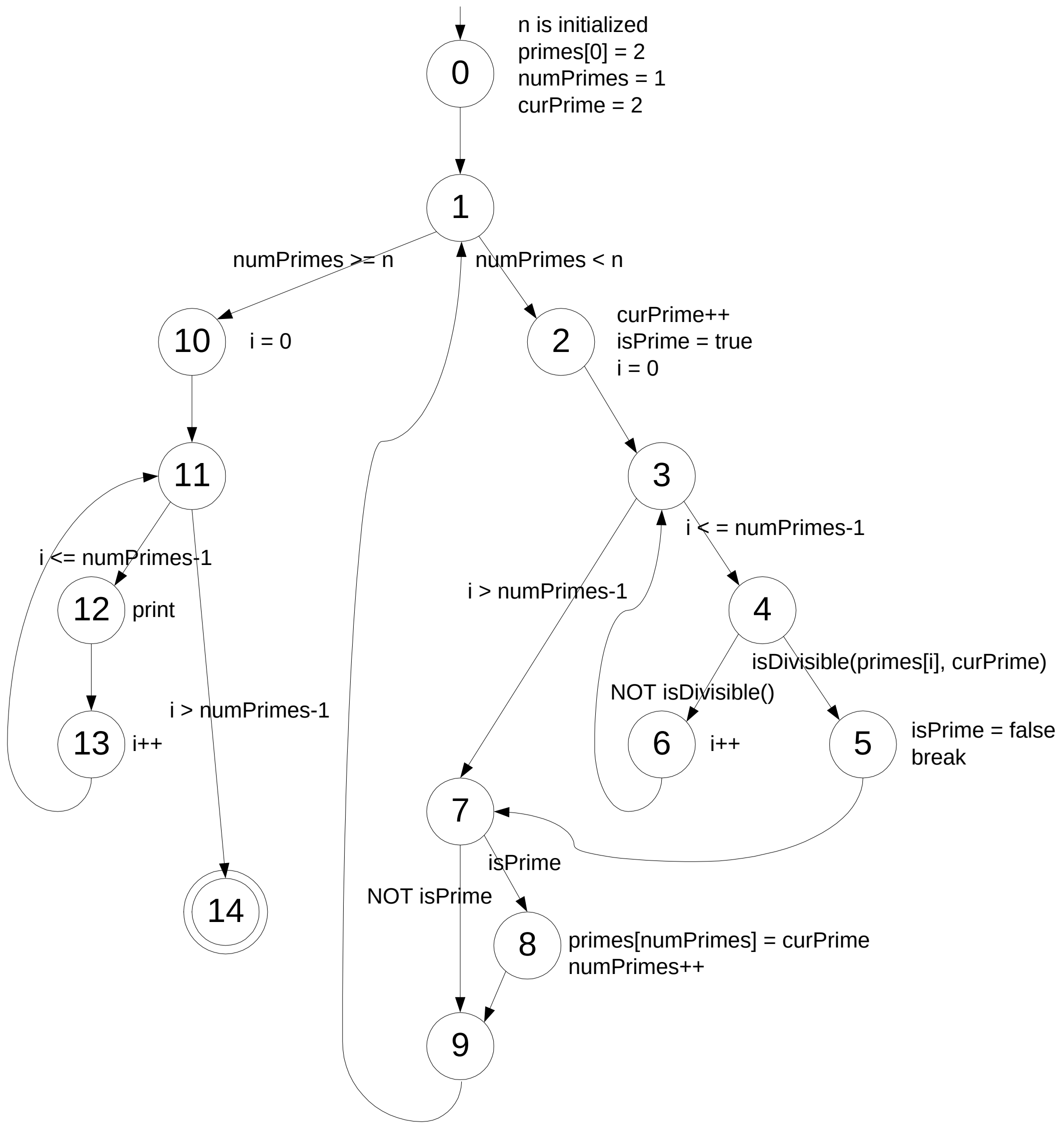

(b) Consider test cases $t_1 = (n = 3)$ and $t_2 = (n = 5)$. Although these tour the same prime paths in *printPrimes()*, they do not necessarily find the same faults. Design a simple fault that t_2 would be more likely to discover than t_1 would.

Solution (Instructor only):

An obvious and boring fault is if the while loop test is incorrect – for example, `while (numPrimes < 3)`*.*

A more interesting type of fault is to note that `n=3` *returns all the odd numbers between 2 and 5, whereas* `n=5` *does not. Thus a fault that caused the program to return odd numbers instead of prime numbers would be detected by* `n=5`*, not* `n=3`*. For example, if the if test was* `if isDivisible (primes[0], curPrime)`*, or if the* `isDivisible()` *method was implemented incorrectly. This is 'interesting'*