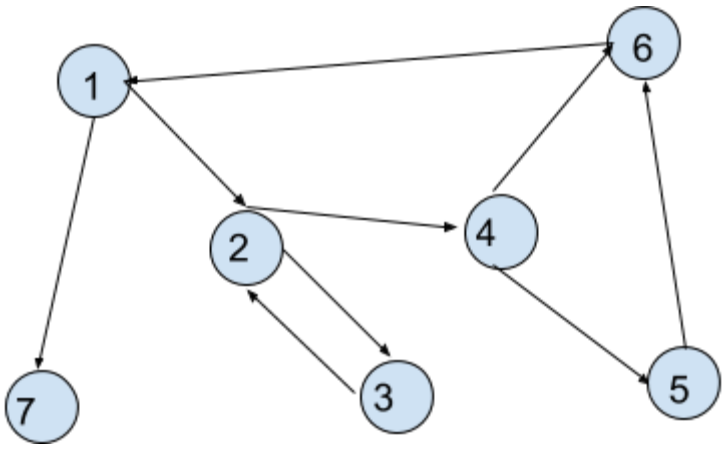
SE 419 HW #3
Shaun VanWeelden, Cole Cummings, Anne Ore

1. 2nd ed , Chap. 7.2.2., page 18 in 1/5/16 version, Problem #5: (a), (b), (e), (f) and (g) only. 25 pts.

a)



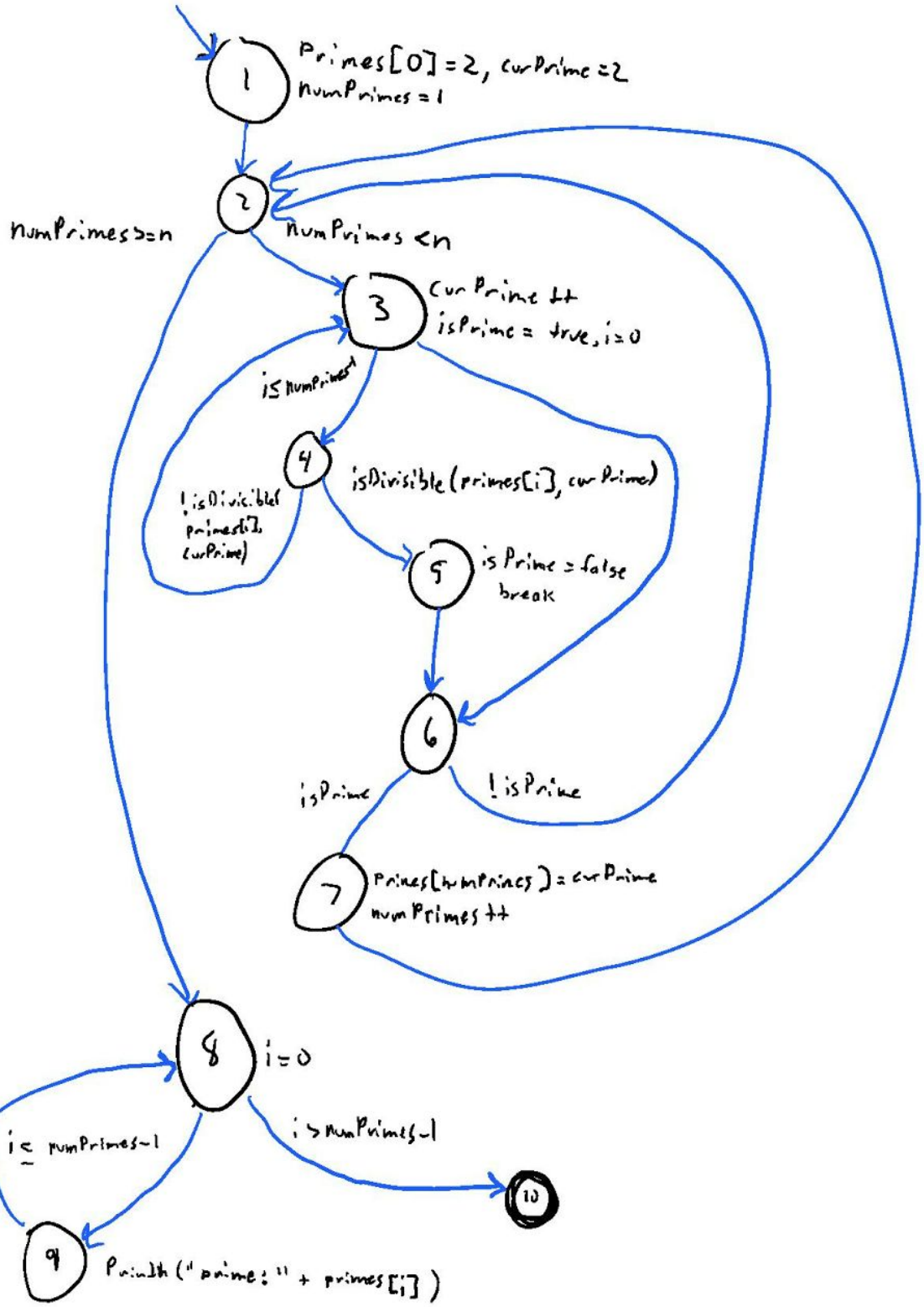
b) (1-2-3) (1-2-4) (2-3-2) (2-4-5) (2-4-6) (3-2-3) (3-2-4) (4-5-6) (4-6-1) (5-6-1) (6-1-2) (6-1-7)

e)
Node coverage should contain (1,2,3,4,5,6,7)
Edge coverage should contain (1-2) (1-7) (2-3) (2-4) (3-2) (4-5) (4-6) (5-6) (6-1)
Prime path coverage would be (1-2-3) (1-2-4-5-6-1) (1-2-4-6-1) (1-7) (3-2-3)

f) p3, all nodes are covered but (4-6) is not covered. (P1 doesn't cover node 3, p2 doesn't cover node 5)

g) p3 achieves edge coverage but is not a prime path because there are internal loops. (p1 does not cover (2-3) and p2 does not cover (4-5) so they are not edge coverage)

2. 2nd ed , Chap. 7.3., page 42 in 1/5/16 version, Problem #7: (a) and (c) only. 20 pts.
The control flow graph for the printPrimes() method

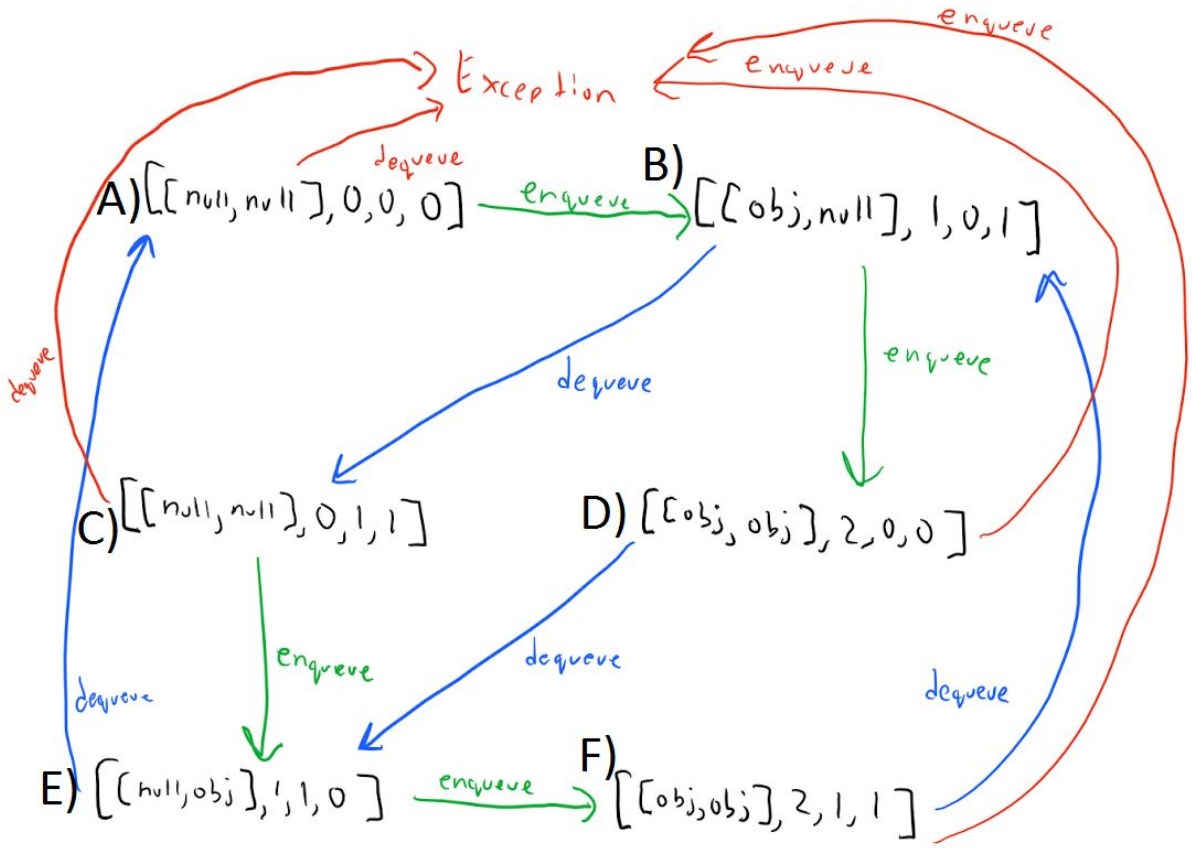


For printPrimes(), find a test case such that the corresponding test path visits the edge that connects the beginning of the while statement to the for statement without going through the body of the while loop.

If n is less than or equal to 1, the condition in the while loop will evaluate to false, skipping the body of the while loop and going immediately to the for loop.

3. 2nd ed , Chap. 7.5., page 65 in 1/5/16 version, Problem #1: (d), (e) and (f) only. On (f), just define, don't have to execute. 25 pts

I combined parts d and e, the text in black is each state, plus one state for an exception. Green edges are for enqueues and blue are dequeues. The start state is always A.



A set of test cases that would cover each edge would be:
To hit each unique edge that points to an exception

1) A -> Dequeue -> Exception
2) A -> Enqueue -> Dequeue -> Dequeue -> Exception
3) A -> Enqueue -> Enqueue -> Enqueue -> Enqueue -> Exception
4) A -> Enqueue -> Enqueue -> Dequeue -> Enqueue -> Enqueue -> Exception

After following the failure paths, only the edges C->E, E->A, and F->B remain, the following test method would cover each:
A -> Enqueue -> Dequeue -> Enqueue (covers C->E) -> Dequeue (covers E->A) -> Enqueue -> Enqueue -> Dequeue -> Enqueue -> Dequeue -> B (covers F->B)

4. 2nd ed , Chap. 8.1.6., page 22 in 1/5/16 version, Problem #5. Also answer: does your solution also satisfy predicate coverage? 20 pts.

a = true	x = 0, y = 1
a = false	x = 1, y = 0
b = true	done = true
b = false	done = false
c = true	list = [“one”], str = “one”
c = false	list = [“one”], str = “two”

To satisfy predicate coverage, use ((x = 0, y = 1), done = true, list = [“one”], str = “one”) for true and ((x = 0, y = 1), done = true, list = [“one”], str = “two”) for false.

5. Very briefly describe your project topic (1-2 sentences) and identify the team members. More info to follow. 10 pts.

Franklin Nelson, Cole Cummings, and Shaun VanWeelden are in a group together.

We'll be doing an analysis of a software called Atlas.Atlas makes it easy to explore and test code by visually augmenting code components along with other features to see where and how they are used in the codebase which makes increasing code coverage and writing more robust tests easier than ever.

http://www.ensoftcorp.com/atlas/developers/

Anne sent an email separately for her group