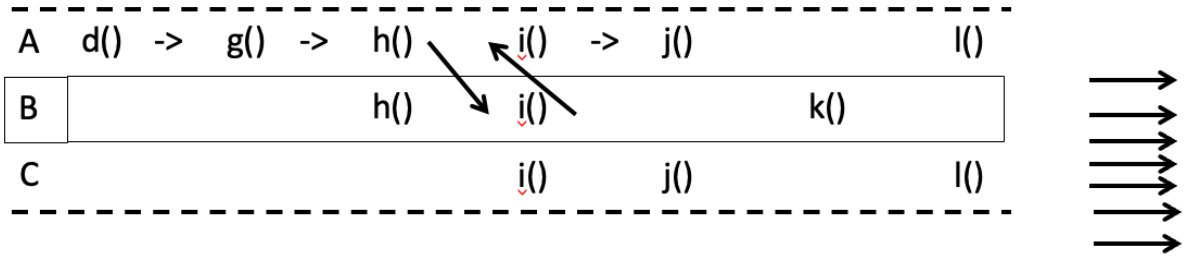
SYSC 4101 Lab 9
Juanita Rodelo
101141857

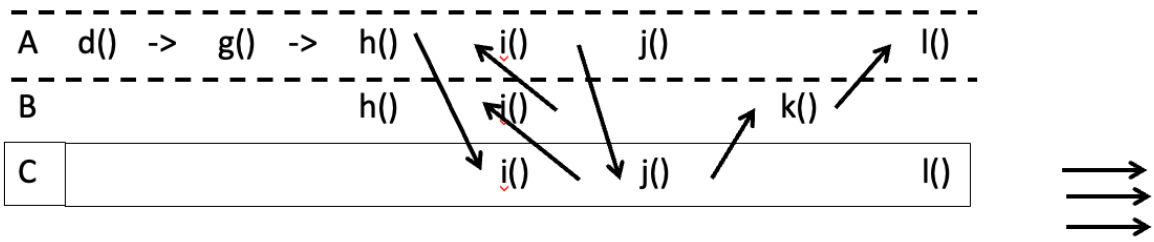
Exercise A

Call parent if class isn't called in child

1. Since class B does not call d() directly, it inherits the call in its parents class, which is class A so the following sequences are done when d() Is called on an instance of class B. After A calls h(), i() is called and class B defines i() so we use class B's instance of i() which then calls A.i() so we go back to class A's instance of i():



2. Since d() is not called in class C, class C inherits its parents behavior which is B. Class b does not call d() either so class B inherits its behavior from its parent class which is class A. That is why the sequence of events starts at class A. This continues until we see a class that is called within class C.



3. Yes these methods do need to be re-tested in the context of class B because class B now uses these methods in different ways and situations. These new ways may not be covered in the tests that were done in the context of class A so they need to be re-tested.

4. In this case, the test scaffolding done to test these methods in class A should be designed in a way that is applicable to instances in class B. This means that the instance of these methods in class B should not break the functionality in class A. However there should be new tests that directly test the instances defined in class B. Overall, the test scaffolding in class A should be able to be re-used to account for inheritance and

polymorphism and the introduction of class B's behavior should not break the functionality from class A.

5. The following methods need to be re-tested:
   h(): because class A calls i() after h() is called and i() is re-defined in class C.
   i(): because class C redefines this method
   j(): because class C redefines this method
   l(): because class C redefines this method
   k(): because class C calls k() after j() is called and k() is defined in class B and not class C so class B's behavior is inherited for this call. Class B calls l() after k() which is redefined in class C so k() needs to be re-tested.
   d() may need to be re-tested in the context of class C because it inherits class A's behavior which calls g() then calls h() then calls i() which is redefined in class C.

Exercise B

1. Test 1: An empty feed
   Input: A instance of FeedOfIntValues with no values
   Expected Output: No factorial should be computed

   Test 2: Feed with one value
   Input: An instance of FeedOfIntValues with one positive integer value (ex. 5)
   Expected Output: the factorial of that single value (ex. 120 for 5)

   Test 3: Feed with multiple values
   Input: An instance of FeedOfIntValues with multiple positive integer values
   Expected Output: Factorials of all values in given in the input

   Test 4: Feed with 0 value
   Input: 0
   Expected Output: 1

   Test 5: Feed with negative values
   Input: An instance of FeedOfIntValues with negative integer values (ex. -1, -5, -10)
   Expected Output: ? not defined

2. See attached java files