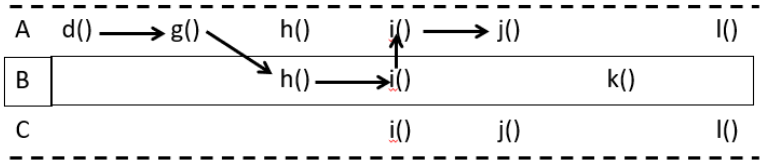
# SYSC 4101A
## Lab 9
**Nathan MacDiarmid**
**101098993**

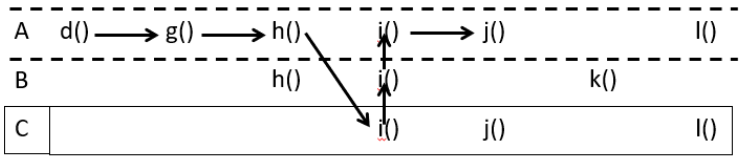
## Exercise A
### Question 1




| | Definitions | Uses |
|---|---|---|
| **1** | B.x | |
| **2** | | B.x |
| **3** | | A.u |
| **4** | A.v | A.w, A.v |

Yes, there are problems with this. A.i() calls the attribute definition A.u and A.w but they are defined in A.h(). Since we called B.h() instead, attributes A.u and A.w were never defined.

### Question 2




| | Definitions | Uses |
|---|---|---|
| **1** | A.u, A.w | |
| **2** | C.y | |
| **3** | | A.u |
| **4** | A.v | A.w, A.v |

There are no problems with this as all attributes are defined before potential use.

**Question 3**

No, they do not need to be re-tested in the context of class B. This is because the Hierarchical Incremental Testing principle states that “any inherited method which interacts with any re-defined method should be re-tested in the context of the derived class”. This means that if a method from the parent class is re-defined in the child class, that method should be re-tested in the context of the child class. In our scenario, class B is a child of class A and all methods of class A have been properly tested. Since class B does not provide its own definition of d(), g(), j(), and l(), there is no need to re-test them in the context of class B.

**Question 4**

The test scaffolding needs to be re-defined. Based on the Hierarchical Incremental Testing principle, h() and i() have been tested thoroughly in the context of class A, but need to be re-tested in the context of class B because of their re-definition. This is because the methods do not behave the same way in class B as they do in class A. In class A, h() calls i() and defines A.u and A.w, whereas in class B, h() calls i() and defines B.x. Furthermore, in class A, i() calls j() and uses A.u, whereas in class B, i() calls A.i() and used B.x. Based on this example, we can prove that the test scaffolding is not sufficient anymore as the oracle would be checking for the wrong outputs.

**Question 5**

According to the Hierarchical Incremental Testing principle with classes A and B being properly tested, methods d(), g(), h(), l(), and k() do not need to be re-tested as they have been properly tested already in classes A and B. Since methods i(), j(), and l() have been re-defined in class C, they need to be re-tested in its context. However, since k() was defined in class B and called l() from class A and class C also has a method l() now, I am going to assume that k() in class B calls l() in the parent automatically.

## Exercise B

**Question 1**

**(input, output)**
(-1, 1), (0, 1), (1, 1), (2, 2), (3, 6)

Although redundant as (0, 1) and (1, 1) will satisfy all edge criterion, I added extra test cases to see output when input value was higher.

**Question 2**

See three attached .java files. FactorialTest.java is my stub class that implements the FeedOfIntValues.java interface and passed to Factorial.java.