**PRIVACY LEAK COMMENTS AND PROPOSALS TO AVOID THEM**

By

Salahddin Warid

40191626

Kevin Estrada Dominguez

40076515

A report submitted in partial fulfilment of the requirements of COMP-249

Concordia University

August 202

**Discussion**

Privacy leaks in a program happen when a client of a class can access the private attribute of the class given its reference. For instance, this can occur when a getter of an attribute returns a direct reference to an object. To avoid this, we made the CellNode Class private and being the inner class of CellList. That means we didn’t need to create any getters or setters for our attributes of the CellNode class because the outer class (CellList) can access the inner class freely. Having the inner class as private enforces encapsulation and give some form of protection against privacy leak. For the outer class CellList, we made its attributes private (to provide encapsulation) from which only the integer size of the array can be access by a getter. The CellNode object (head) cannot be accessed directly by the user. All the methods in the CellList are public, with the exception of a helping method which is private. The only method that risks returning a reference object is the public CellNode find(long serialNum) method. Note that we made sure to create a new object when returning the CellNode object found by the method to avoid a privacy leak.

The CellPhone class only contained primitive attributes (except a String variable), so we didn’t discuss it here.