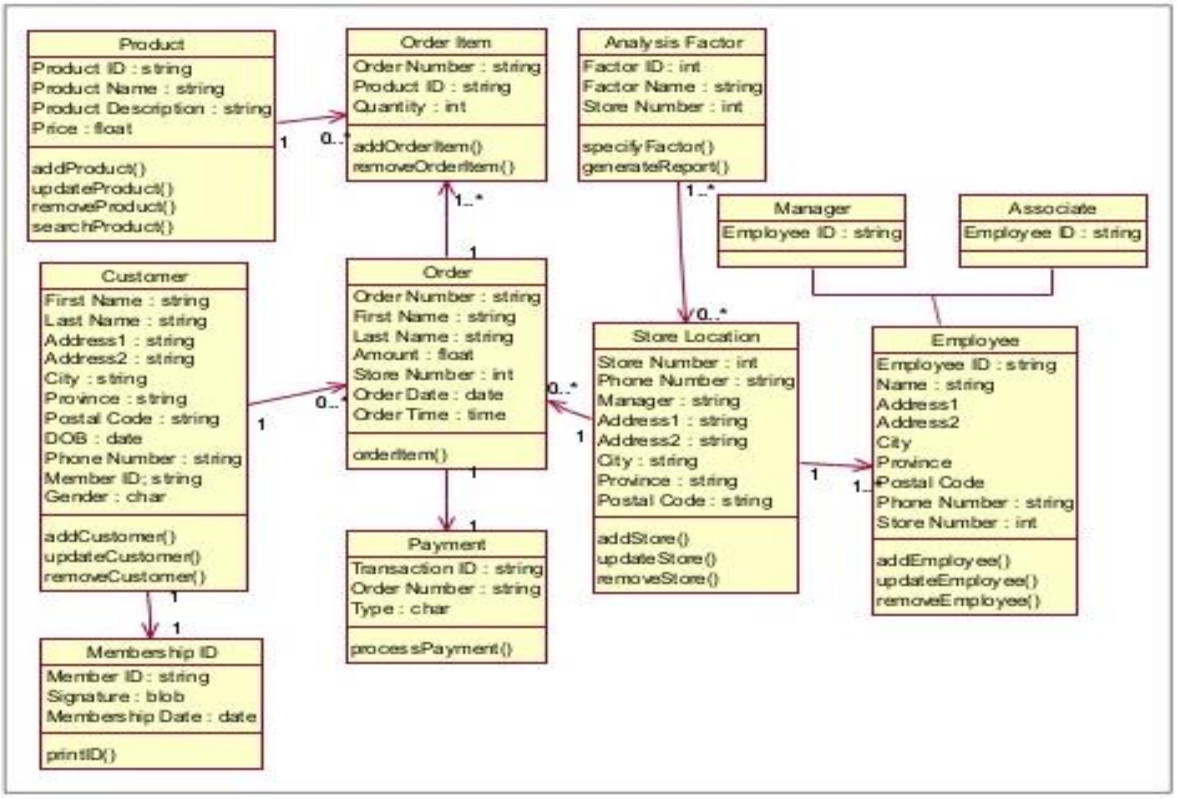
SOFTWARE QUALITY & METRICS
QUIZ 1

- ASHISH KUMAR
- 2K18/SE/041

Ques 1: Identify two types of design antipatterns (also called code smells) in the following class diagram and explain your detection process based quality metrics to justify your decision (symptoms).



Ans 1: Two types of design antipatterns that I found:

1. <u>Duplicate Code(Data Clump)</u>

   If we see classes Customer, Store location and Employee, we can see that the address information which consists of:

   address1, address2, city, province, postal code

   So, these parameters are sharing duplicate code because they are written redundantly in each of the classes.

2. Pull Up method(move attributes)

We can see that Classes Manager and Associate both have a common attribute i.e., employee id field which is inherited from the employee class. So, solution for this is to Make the methods identical and then move them to the relevant superclass i.e. employee class.

Ques 2: Define software refactoring then recommend a set of useful refactoring to fix the two types of design antipatterns identified in the previous question (Question 1). It is not required to provide a source code implementing the refactorings. You can provide the name of the sequence of refactorings type along with their parameters such as movemethod (*m1*, Class A, Class B) as a solution to fix the design antipatterns.

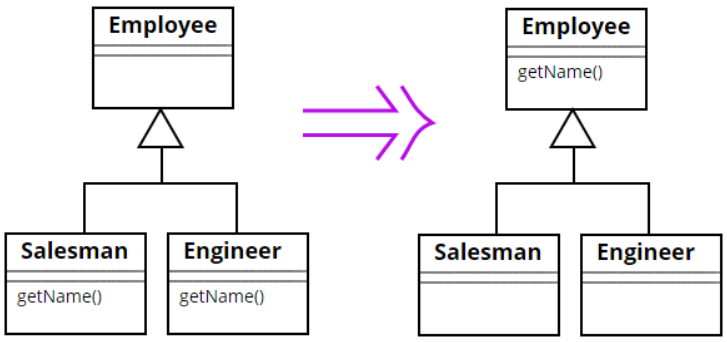
Ans 2: Software refactoring is modification of internal structure of software for readability, maintainability and extensibility without affecting the external functionality of the software.

We can use **extract class refactoring** to fix the design antipatterns. We can extract common features of customer, store location and employee class into super class (Address). This makes it easier to bind class to an abstraction, and removes duplicate code from the original classes.

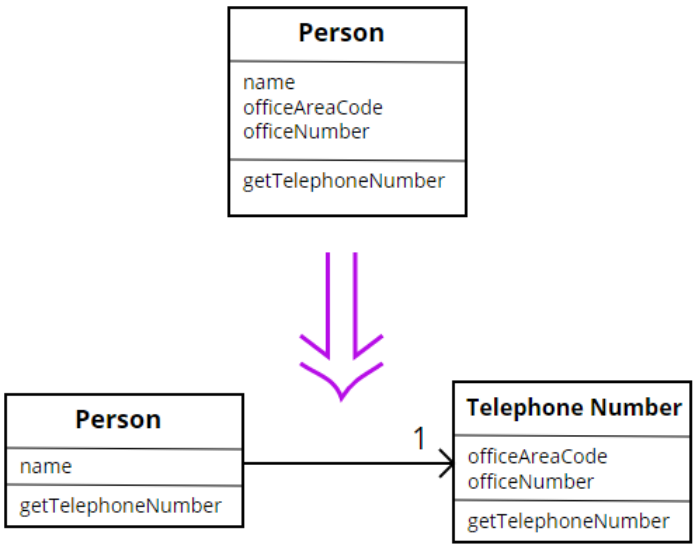
We can use **move method refactoring** to fix the design antipatterns. We can pull up the employee id from manager and associate class to employee class.

Ques 3: Identify the applied refactoring(s) for each of these diagrams.

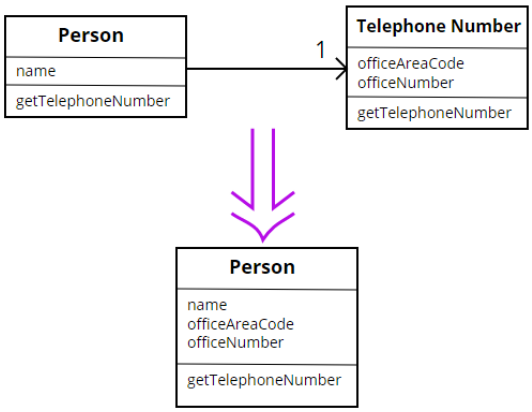
a.



b.



c.



Ans 3:

a) Pull up method Refactoring (Move method).
b) Extract class Refactoring (break one class into two classes).
c) Inline class Refactoring (combine two classes into one class).