ESOF 322: Software Engineering I **DUE Date: September 19, 2019**
30 pts

Instructions:

- Do all exercises with your partner.
- Clearly print the names of all participants in the first page of your assignment.
- No hand-written answers allowed.
- Absolutely no late assignments.
- Your assignment should be turned in to D2L by all students

Exercise

The **Strategy pattern** is a design pattern used to encapsulate different behaviors and/or algorithms. The idea is to allow you to swap those strategies at will during the execution of a program. The architecture of the program remains the same. According to Gamma et al., the *"Strategy pattern is intended to define a family of algorithms, encapsulate each one, and make them interchangeable. Strategy lets the algorithm vary independently from clients that use it."*

In this assignment, you are to design, then implement a program in Java (or your OO language of choice) that uses the Strategy pattern to solve the following problem.

You would like to provide a system for your customer that allows them to choose any one of three types of database software (Relational, NoSQL, Graph). Each of these database tools provides a function (*Store()*) to store information. The **Relational** db tool uses table store, **NoSQL** uses document store, and **Graph** uses node store as their default algorithms respectively. You would like to allow your customer to select any one of these software tools and allow them to change the default algorithm to store the information with any of the three storage algorithms.

A) Design a UML **class diagram** to solve this problem. Use the Strategy Pattern.
B) Write code in your favorite OO language to implement your design. The various store methods can be implemented as dummy methods. That is, just store the data to a file. The emphasis of this homework is not on DB storage techniques, rather on the Strategy pattern. The data can be anything you want.
C) Create a UML sequence diagram that clearly shows the creation/usage of any database tool, its storage execution, the runtime switching of algorithms, and then a *Store( )* call again.

You will test your code by creating a client program (i.e. a main function) where you simulate the interaction between a client and the selection of database software. The main function will request your selection of software (Relational, NoSQL, or Graph) then call *Store()* to execute the default strategy to store information. Then call *setStoreStrategy()* on your database software object to dynamically switch strategies to one of the other storing alternatives and execute it.

Hand in to D2L:

- A UML class diagram of your design (part A),
- A file of your code in text format (part B). For example, hand in a .java or a .cpp file that I could compile and run.
- The output of running your program that clearly shows the main function executing the default strategy and the dynamic switching to another strategy. Include print statements on every method to help instrument your code. Your code should be well documented (but not excessively).
- A UML sequence diagram (part C)
- Bring each of these deliverables (a printout) to class.