SE2811 Half-exam 2 Name:

This is a 25-minute, 100-point half-exam. The exam is printed double sided. If you use a note-sheet (prepared by yourself), please turn it in with your exam. You may use no other aids.

1. Decorator pattern

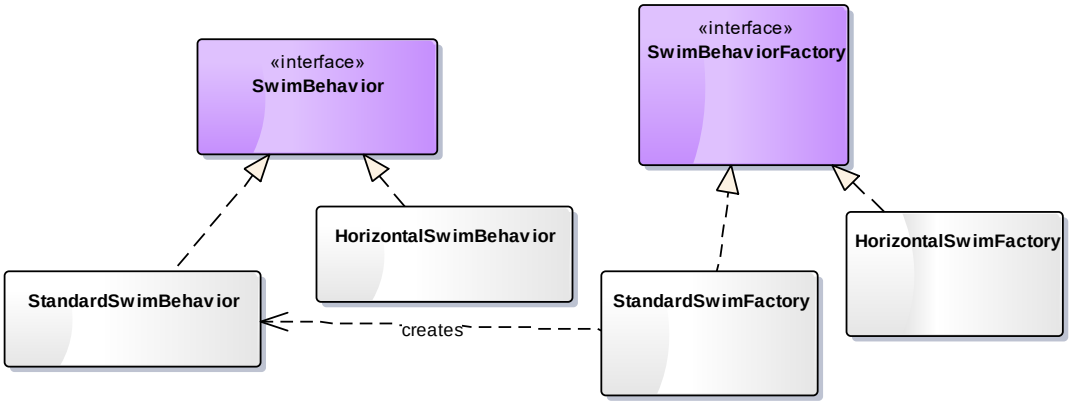
    a. (20 points) ***Write*** code to **create** a Writer that takes whatever is written to it, replaces spaces with minus signs, and encrypts it using a Caesar cipher, and then writes it to the file "secret.txt", in that order. Assume the constructor `FileWriter(String filename)` creates a new FileWrite to write the given file.

```
public static void main(String[] ignored) {




}
```

2. (10 points) ***Circle*** the classes/interfaces in the Factory Method Pattern which are open for extension, but closed for modification.



3. (10 points) A route-finding application finds the best path between two points by summing the cost along each segment of the path. ***Circle one*** pattern and ***explain*** how it allows the client to supply custom code that will compute the cost for a single segment according to different metrics (time, fuel consumption, scenic beauty, probability of crash, etc).

    a. Strategy
    b. Factory Method
    c. Singleton
    d. Observer
    e. Decorator
    f. Composite

4. (20 points) ***Draw*** a UML diagram illustrating how Swing implements the **observer** pattern to decouple an event source from the client code that responds to the event. For each class, write both the **name in Swing** and the **role** the class plays. Write the roles in angle brackets (e.g. <<abstract subject>>). It is OK if you do not remember the name of every

Swing class correctly, but write the pattern roles correctly. If one class has two roles, write both roles. Include relationship arrows with names if appropriate, and label any multiplicities greater than one. You do not need to include methods or fields in the diagram.

5. (20 points) ***Write*** a UML class diagram for a **composite** pattern for a drawing application. Your application should allow multiple shapes to be grouped into a single object on the screen. You should also be able to include an existing group in a new group. The basic shapes are an ellipse and a rectangle. Include the same elements (e.g. name, role, relationships, …) as on the previous problem.