CPSC310 Sample Final Questions

1. What code smell is present in this method from the Zoo project's Animal class? Perform the refactoring to fix it, introducing any new classes needed, and changing the call-site as required.

```
public class Zoo {
...
    public static void main(String ... args){


        animals.add(new Animal("Koala", "herbivore"));


        animals.add(new Animal("Wolverine", "omnivore"));


        animals.add(new Animal("Dragon", “carnivore"));

        ...
        for (Animal a : animals){
            a.getSafety();
        }
    }

}
```

```
public class Animal {
    ...
    String safety;
    ...
    public Animal(String name, String diet){
        this.name=name;
        safety="safe";
    }
    ...
    public String getSafety() {
        if (diet.equals("carnivore")) {
            safety="Safe";
        }
        if (diet.equals("herbivore")) {
            safety="UNSAFE!";
        }
        if (diet.equals("omnivore")) {
            safety="Kind of Safe";
        }
        System.out.println(this.name+" is "+safety);
        return safety;
    }
}
```

2. Provide a design example for each of the SOLID principles (meaning draw a small UML diagram or write some code and explain in your own words how that code/diagram satisfies the principle):

   A. Single Responsibility Principle

   B. Open/Closed Principle

   C. Liskov Substitution Principle

   D. Interface Segregation Principle

   E. Dependency Inversion Principle

3. Use the Composite pattern to implement the **PrettyPrincess** application: clothing can be worn over other clothing, but jewellery cannot have other jewellery on top of it. Provide a UML diagram, and working code.

4. Assume you are implementing an application that makes use of two interfaces - the FaceBook interface, and the GooglePlus interface, depending on a user setting. Draw the UML diagrams and provide skeleton code for a solution using the Adapter pattern.

5. Now assume your friend is creating a third social networking interface UBCFriends, but is only partly done. Make a Mock Object to help you test your application while theirs is in development. Include configuration and instantiation code.

6. Find 10 ways to mutate the following code:

```
public boolean hasNext() {
    if (!doneListOne && (counter >= listOne.size())) {
        doneListOne = true;
        counter=0;
    }

    return !doneListOne || counter < listTwo.size();
}
```

7. Explain the difference between Basis Path coverage and Full Path coverage. Is one a subset of the other?

8. Draw a CFG for the following method, and provide a test set that provides statement and branch coverage for this method:

```
int foo (int x, int y)
{
    int z = 0;
    if ((x>0) && (y>0))
    {
        z = x;
    }
    return z;
}
```

9. Imagine a system (Holiday Shopper) that covers the user story: “As someone planning my holiday shopping, I would like to be able to have a list of people to shop for and be able to add gift ideas under their names”. Write a user instruction to test a paper prototype of your application.

10. Imagine that the Holiday Shopper application is a client-server application, where all of the data is held on the server (to allow for updates from multiple clients).
    (a) Construct a RESTful API indicating, for each resource, the GET/PUT/POST/DELETE responses. Assume multiple users for this application
    (b) Indicate how Get/Put/Delete are idempotent
    (c) Indicate how the server is stateless
    (d) Indicate how the client does not need to recall the structure of the server side resources