

Homework 5

Part 1:

In this we will extend the Strategy pattern code in Example5. I've included it under the `userid` directory. The `Main.java` file has been changed.

We will define a new kind of duck – a Pekin duck. Pekin ducks can sometimes fly, but poorly. We'll also introduce a `LaysEggs` behavior.

LaysEggs: Create an interface for this, with an abstract void `laysEgg()` function. Have `LaysEggsBroody`, `LaysEggsNotBroody`, and `DoesNotLayEggs` be concrete classes that implement the `laysEggs` function.

`LaysEggsBroody` will print "Lays eggs, but will fight you for them.". All of the living ducks in the example except for the Pekin duck will have this behavior.

`LaysEggsNotBroody` will print "Lays eggs and will give them up if fed.". The Pekin duck has this behavior.

`DoesNotLayEggs` will print "Not an egg layer." `DecoyDuck` has this behavior.

FlysPoorly: create a new concrete class that extends the `FlyBehavior` interface class. The `fly()` method will print "flies poorly". This will be the `FlyBehavior` for the Pekin duck.

Pekin duck: A new concrete class that inherits from `Duck`. It will have a `Quack` quacking behavior, `FlysPoorly` flying behavior, `LaysEggsNotBroody` egg laying behavior.

Notice that we had to change a lot of files to do this. This is because we didn't just add a new type of an existing behavior, e.g., a new kind of flying, but we added an entirely different kind of behavior.

To test your code follow the comments in the `main.cpp` file.

Part 2:

Add a new duck, the `ToyDuck`. It will not fly, will quack, and has a new egg laying behavior called `LaysToyEggs` that prints "Lays toy eggs." Its display function prints "I'm a toy duck."

Note that for part two the only changes needed are for the code related to the new functionality, i.e., a new `LaysToyEggs` class that inherits from `LaysEggs`, and a new `ToyDuck` class that inherits from `Duck` and changing `Main.java` to use the new duck class. All other code should be unchanged.

To test your code follow the comments in the `Main.java` file.

What to turn in:

Zip up your `userid` directory and turn it in. Executing `javac Main.java` in the `userid` directory should compile your code and run the complete program with part 2 implemented.