

JAVA Assignment 6

Sameer Khatwani
AIML-B1
PRN: 2070126099

- 1) Write an application to show the behavior of a Duck.
- 2) Create classes as mentioned in <https://www.oreilly.com/api/v2/epubs/0596007124/files/figs/web/022fig01.png.jpg>
- 3) Also create a new Behaviour :
 - a) <<interface>> SwimBehavior
 - b) Three different classes Swim, Float, and Drown which implement SwimBehavior containing swim() method.
- 4) Print which duck will fly, float or swim.
- 5) Create a class diagram for entire program (including swim interface).

CODE:

```
//SAMEER KHATWANI
//AIML-B1
//PRN: 22070126099

//Main.java

public class Main {
    public static void main(String[] args) {
        System.out.println("*****");
        RedHeadDuck redHeadDuck = new RedHeadDuck();
        redHeadDuck.display();
        redHeadDuck.performFly();
        redHeadDuck.performQuack();
        redHeadDuck.performSwim();
        System.out.println("*****");
        RubberDuck rubberDuck = new RubberDuck();
        rubberDuck.display();
        rubberDuck.performFly();
        rubberDuck.performQuack();
        rubberDuck.performSwim();
        System.out.println("*****");
        MallardDuck mallardDuck = new MallardDuck();
        mallardDuck.display();
        mallardDuck.performFly();
        mallardDuck.performQuack();
        mallardDuck.performSwim();

    }
}
```

```
//DecoyDuck.java
```

```
public class DecoyDuck extends Duck {

    public DecoyDuck() {
        flyBehaviour = new FlyNoWay();
        quackBehaviour = new Squeak();
        swimBehaviour = new SwimNoWay();
    }

    @Override
    public void display() {
        System.out.println("Decoy mf");
    }

}
```

```
//Duck.java
```

```
abstract public class Duck {

    FlyBehaviour flyBehaviour;
    QuackBehaviour quackBehaviour;
    SwimBehaviour swimBehaviour;

    public void setFlyBehaviour(FlyBehaviour fb){

        flyBehaviour = fb;

    }
    public void setQuackBehaviour(QuackBehaviour qb){

        quackBehaviour = qb;
    }
    abstract void display();

    public void performFly(){

        flyBehaviour.fly();
    }
    public void performQuack(){

        quackBehaviour.quack();
    }

    public void performSwim(){
        swimBehaviour.swim();
    }

}
```

```
//Float.java
```

```
public class Float implements SwimBehaviour{  
    public void swim(){  
        System.out.println("Floating mf");  
    }  
}
```

```
//FlyBehaviour.java
```

```
public interface FlyBehaviour {  
    public void fly();  
}
```

```
//FlyNoWay.java
```

```
public class FlyNoWay implements FlyBehaviour{  
    @Override  
    public void fly(){  
        System.out.println("Can't fly mf");  
    }  
}
```

```
//FlyWithWings.java
```

```
public class FlyWithWings implements FlyBehaviour{  
    @Override  
    public void fly(){  
        System.out.println("Flying with wings mf");  
    }  
}
```

```
//MallardDuck.java
```

```
public class MallardDuck extends Duck {  
  
    public MallardDuck(){  
  
        flyBehaviour = new FlyWithWings();  
        quackBehaviour = new Quack();  
        swimBehaviour = new Swim();  
    }  
    @Override  
    void display(){  
        System.out.println("MallardDuck mf");  
    }  
}
```

```
//NoSwim.java
```

```
public class NoSwim implements SwimBehaviour{  
    public void swim(){  
        System.out.println("Can't swim mf");  
    }  
}
```

```
//Quack.java
```

```
public class Quack implements QuackBehaviour {  
    @Override  
    public void quack(){  
        System.out.println("Quack mf");  
    }  
}
```

```
//QuackBehaviour.java
```

```
public interface QuackBehaviour{  
    public void quack();  
}
```

```
//RedHeadDuck.java
```

```
public class RedHeadDuck extends Duck {  
  
    public RedHeadDuck(){  
  
        flyBehaviour = new FlyWithWings();  
        quackBehaviour = new Quack();  
        swimBehaviour = new Swim();  
    }  
    @Override  
    void display(){  
        System.out.println("RedHeadDuck");  
    }  
}
```

```
//RubberDuck.java
```

```
public class RubberDuck extends Duck {  
    public RubberDuck(){  
        flyBehaviour = new FlyNoWay();  
        quackBehaviour = new Squeak();  
        swimBehaviour = new Float();  
    }  
  
    @Override  
    void display(){  
        System.out.println("RubberDuck");  
    }  
}
```

```
//Squeak.java
```

```
public class Squeak implements QuackBehaviour{  
    @Override  
    public void quack(){  
        System.out.println("Squeak mf");  
    }  
}
```

```
//SwimBehaviour.java
```

```
public class Swim implements SwimBehaviour{  
    public void swim(){  
        System.out.println("Swimming mf");  
    }  
}
```

```
//SwimBehaviour.java
```

```
public interface SwimBehaviour {  
    public void swim();  
}
```

```
//SwimNoWay.java
```

```
public class SwimNoWay implements SwimBehaviour {  
    public void swim() {  
        System.out.println("Can't swim mf");  
    }  
}
```

OUTPUT:

```
*****
RedHeadDuck
Flying with wings mf
Quack mf
Swimming mf
*****
RubberDuck
Can't fly mf
Squeak mf
Floating mf
*****
MallardDuck mf
Flying with wings mf
Quack mf
Swimming mf
*****
```

Github Repo:

<https://github.com/samv28/PIJ>