## Replace Temp with Query

Hier is baseprice een tijdelijke variabele die vervangen kan worden door een methode. Hierdoor wordt er een regel uitgespaard ten koste van een toegevoegde methode.

double basePrice = \_quantity \* \_itemPrice;

if (basePrice > 1000)

return basePrice \* 0.95;

else

return basePrice \* 0.98;

Dit komt er zo uit te zien:

if (basePrice() > 1000)

return basePrice() \* 0.95;

else

return basePrice() \* 0.98;

...

double basePrice() {

return \_quantity \* \_itemPrice;

}

## Replace Conditional with polymorphism

**class** **Bird** {

*//...*

double getSpeed() {

**switch** (type) {

**case** EUROPEAN:

**return** getBaseSpeed();

**case** AFRICAN:

**return** getBaseSpeed() - getLoadFactor() \* numberOfCoconuts;

**case** NORWEGIAN\_BLUE:

**return** (isNailed) ? 0 : getBaseSpeed(voltage);

}

**throw** **new** RuntimeException("Should be unreachable");

}

}

Elke case in het switch statement wordt een klasse die van de abstracte classe Bird extend:

**abstract** **class** **Bird** {

*//...*

**abstract** double getSpeed();

}

**class** **European** **extends** Bird {

double getSpeed() {

**return** getBaseSpeed();

}

}

**class** **African** **extends** Bird {

double getSpeed() {

**return** getBaseSpeed() - getLoadFactor() \* numberOfCoconuts;

}

}

**class** **NorwegianBlue** **extends** Bird {

double getSpeed() {

**return** (isNailed) ? 0 : getBaseSpeed(voltage);

}

}

*// Somewhere in client code*

speed = bird.getSpeed();