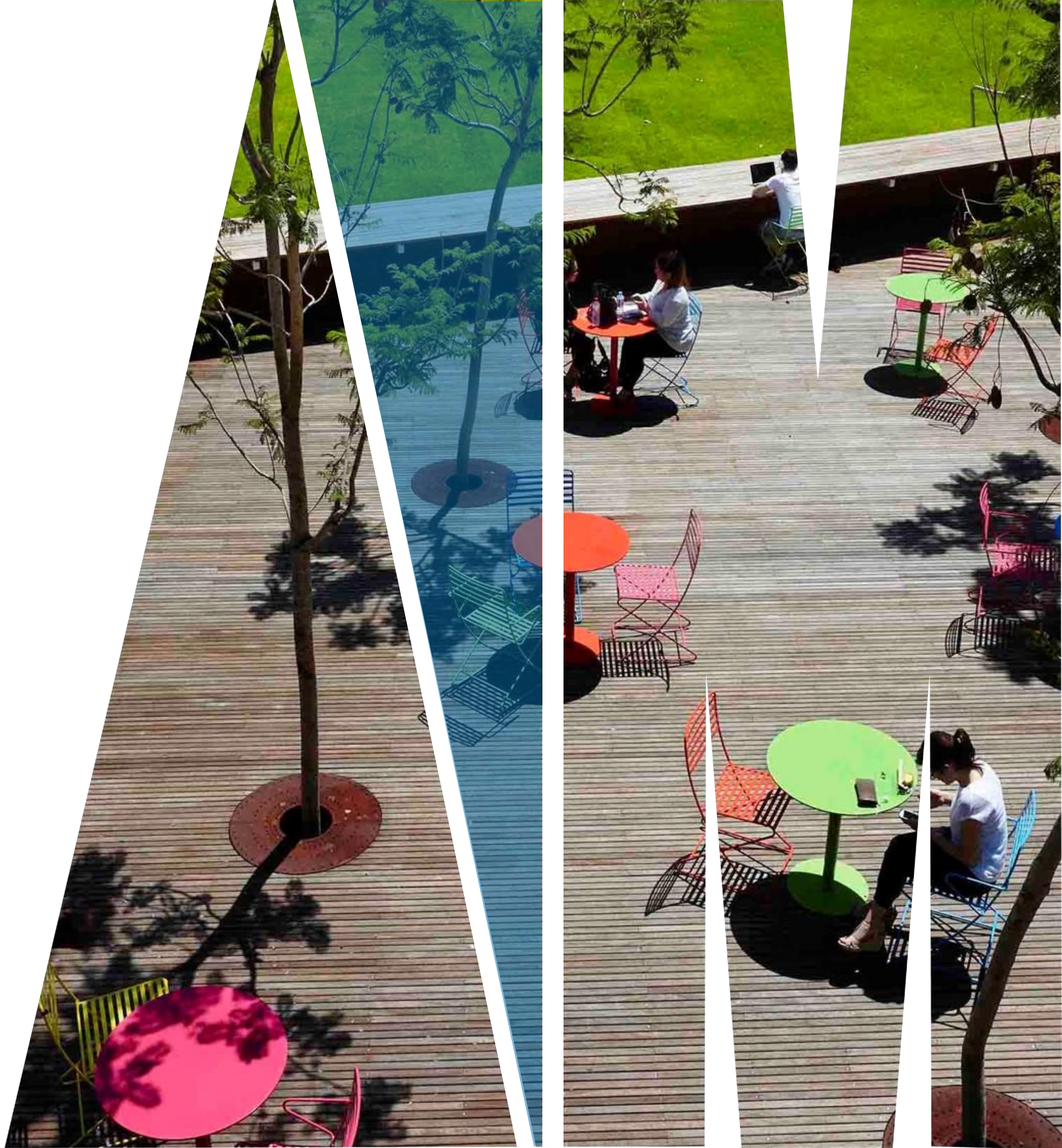
FIT2099 Object-Oriented Design and Implementation

Review of code smells
(by examples)



Outline

Code smells examples

A “BLOATER”

What may be wrong with this code?

What **code smell** can you identify?

How can we **improve the design**?

```
class DateUtil {
    boolean isAfter(int year1, int month1, int day1, int year2, int month2, int day2) {
        // implementation code here
        return true;
    }

    int differenceInDays(int year1, int month1, int day1, int year2, int month2, int day2) {
        // implementation code here
        return 0;
    }

    // other  methods
}
```

A

"BLOATER": Fixing Data Clumps

```
class Date {
    int year;
    int month;
    int day;
}

class DateUtil {
    boolean isAfter(Date date1, Date date2) {
        // implementation code here
    }

    int differenceInDays(Date date1, Date date2) {
        // implementation code here
    }
}
```

What do we call a class with **no methods**?

FIXING A
DATA CLASS

```
class Date {
    int year;
    int month;
    int day;

    boolean isAfter(Date date1, Date date2) {
        // implementation code here
    }

    int differenceInDays(Date date1, Date date2) {
        // implementation code here
    }
}
```

WHAT MAY BE WRONG WITH THIS CODE?

WHAT CODE SMELL CAN YOU IDENTIFY?

```java
public class Pet {
    private String type;

    public String makeSoundInSpanish() {
        switch (type) {
            case "cat":
                return "miau miau";
            case "dog":
                return "guau guau";
            default:
                throw new IllegalStateException();
        }
    }
}
```

COMMONLY, ADDRESSED VIA POLYMORPHISM

```
public abstract class Pet {
    abstract String makeSoundInSpanish();
}
```

```
public class Cat extends Pet {
    String makeSoundInSpanish() {
        return "miau miau";
    }
}
```

```
public class Dog extends Pet {
    String makeSoundInSpanish() {
        return "guau guau";
    }
}
```

WHY IS THIS CODE “SMELLY”?

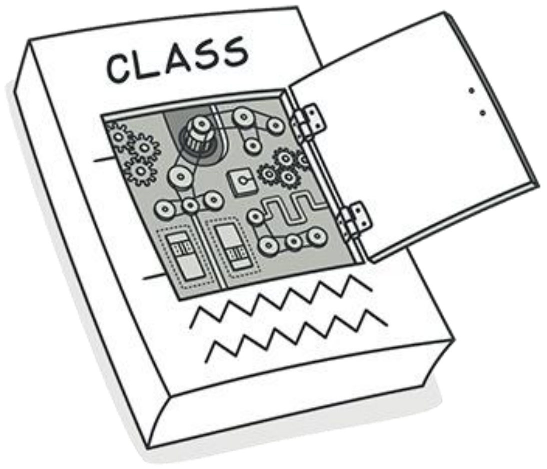
```
public class Hero {
    private Integer stamina;
    private Integer health;
    private Integer armourHealth;
    private Integer armourStatus;
    private String armourRarity;

    public void defense(){
        // implementation code here
    }

    public void attack(){
        // implementation code here
    }
}
```

The team notices that at a specific point they perform too many changes related to the armour functionality.

THIS CAN BE RELATED TO THE SMELL CALLED

DIVERGENT CHANGE

```
public class Hero {
    private Integer stamina;
    private Integer health;
    private Integer armourHealth;
    private Integer armourStatus;
    private String armourRarity;

    public void defense(){
        // implementation code here
    }

    public void attack(){
        // implementation code here
    }
}
```



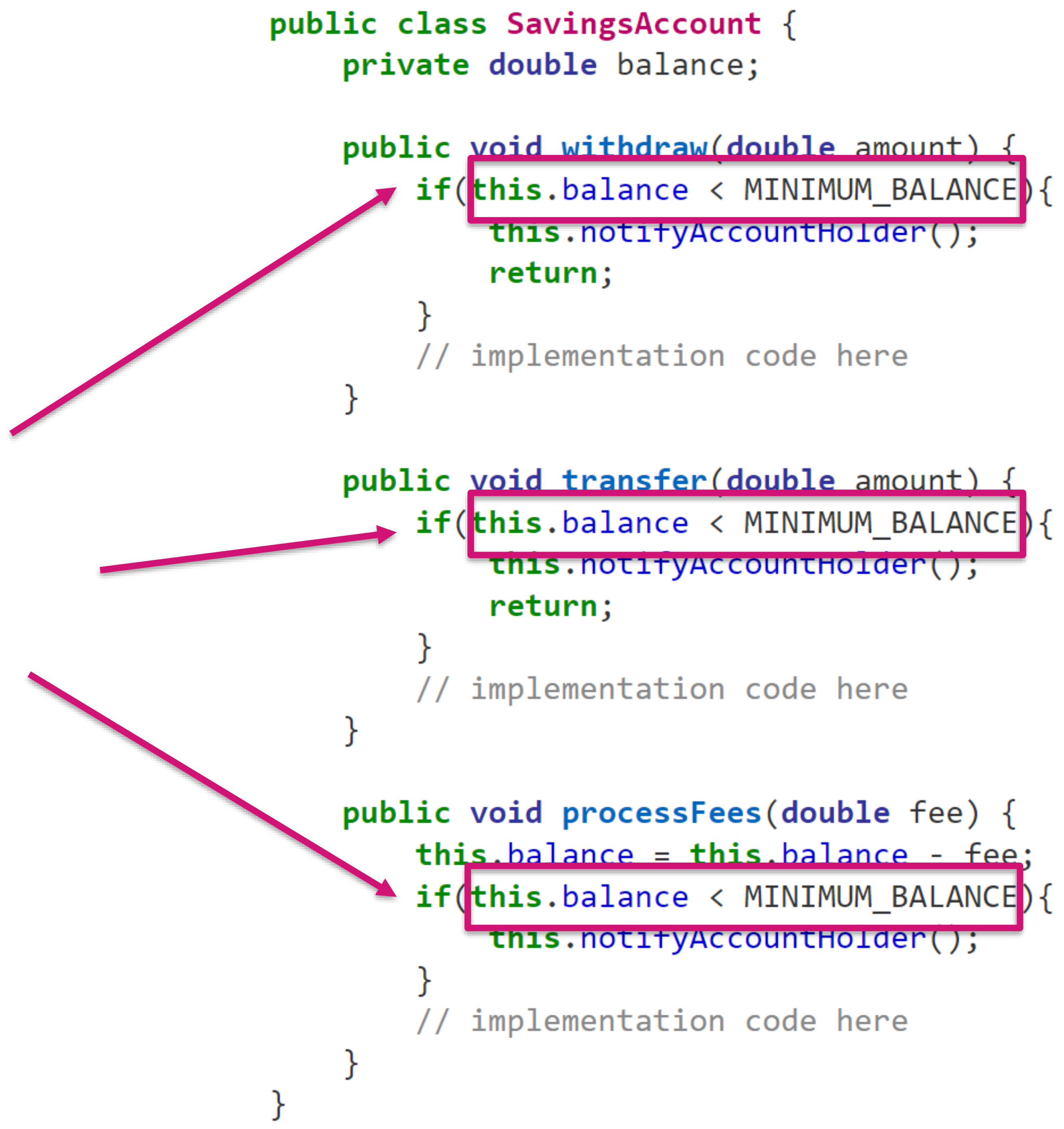
Source: refactoring.guru

HOW TO ADDRESS DIVERGENT CHANGE?

```
public class Hero {
    private Integer stamina;
    private Integer health;
    private Armour armour;

    public void defense(){
        // implementation code here
    }

    public void attack(){
        // implementation code here
    }
}

public class Armour {
    private Integer health;
    private Integer status;
    private String rarity;

    public Integer getHealth() {
        return health;
    }
}
```

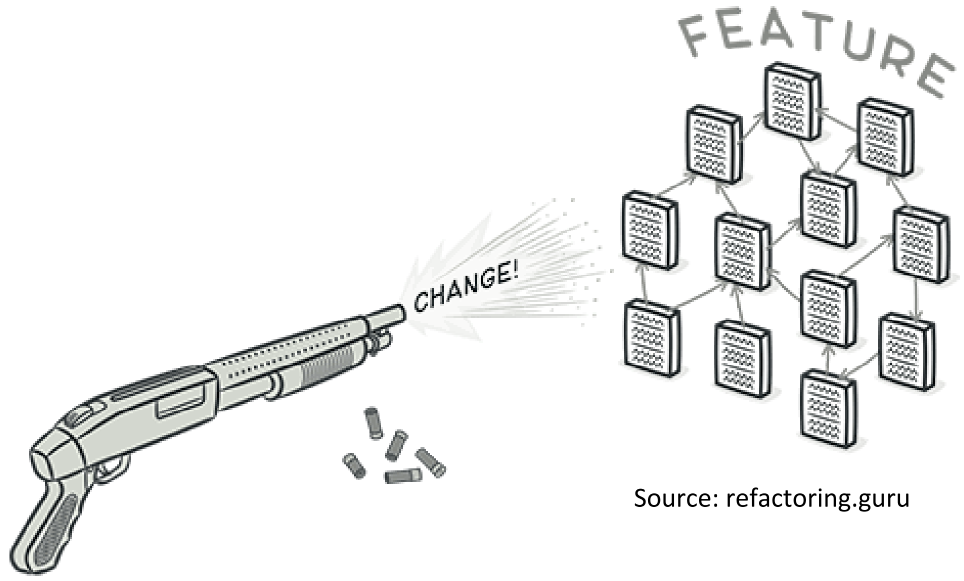
WHY IS THIS CODE “SMELLY”?

There is some very similar code in several places



```java
public class SavingsAccount {
    private double balance;

    public void withdraw(double amount) {
        if(this.balance < MINIMUM_BALANCE){
            this.notifyAccountHolder();
            return;
        }
        // implementation code here
    }

    public void transfer(double amount) {
        if(this.balance < MINIMUM_BALANCE){
            this.notifyAccountHolder();
            return;
        }
        // implementation code here
    }

    public void processFees(double fee) {
        this.balance = this.balance - fee;
        if(this.balance < MINIMUM_BALANCE){
            this.notifyAccountHolder();
        }
        // implementation code here
    }
}
```

THIS CAN BE RELATED TO THE SMELL SHOTGUN SURGERY

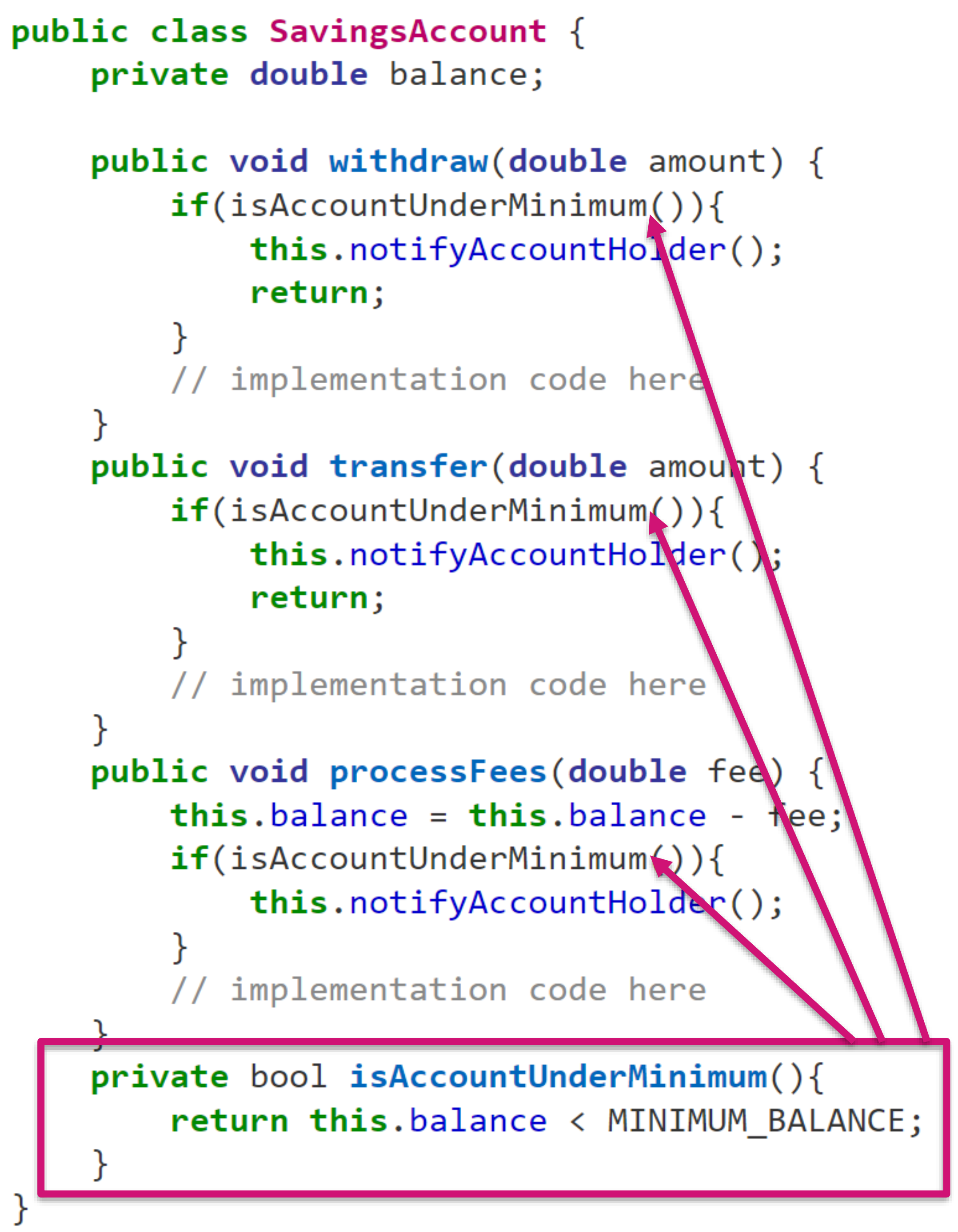


Source: refactoring.guru

```java
public class SavingsAccount {
    private double balance;

    public void withdraw(double amount) {
        if(this.balance < MINIMUM_BALANCE){
            this.notifyAccountHolder();
            return;
        }
        // implementation code here
    }

    public void transfer(double amount) {
        if(this.balance < MINIMUM_BALANCE){
            this.notifyAccountHolder();
            return;
        }
        // implementation code here
    }

    public void processFees(double fee) {
        this.balance = this.balance - fee;
        if(this.balance < MINIMUM_BALANCE){
            this.notifyAccountHolder();
        }
        // implementation code here
    }
}
```

HOW TO ADDRESS

SHOTGUN SURGERY?



```
public class SavingsAccount {
    private double balance;

    public void withdraw(double amount) {
        if(isAccountUnderMinimum()){
            this.notifyAccountHolder();
            return;
        }
        // implementation code here
    }
    public void transfer(double amount) {
        if(isAccountUnderMinimum()){
            this.notifyAccountHolder();
            return;
        }
        // implementation code here
    }
    public void processFees(double fee) {
        this.balance = this.balance - fee;
        if(isAccountUnderMinimum()){
            this.notifyAccountHolder();
        }
        // implementation code here
    }
    private bool isAccountUnderMinimum(){
        return this.balance < MINIMUM_BALANCE;
    }
}
```

WHY IS THIS CODE “SMELLY”?

```java
public class SavingsAccount {
    private double balance;
    private int accountNumber;
    private String accountName;
    private String streetName;
    private String streetNumber;
    private int zipCode;
    private String city;
    private String state;
    private String country;
    private int medicareNumber;
    private Date medicareValidTo;
    private int individualReferenceNumber;

    //getters, setters and other methods
}
```

LET'S FOCUS ON THE PRIMITIVE OBSESSION FIRST

```
public class SavingsAccount {
    private double balance;
    private int accountNumber;
    private String accountName;
    private String streetName;
    private String streetNumber;
    private int zipCode;
    private String city;
    private String state;
    private String country;
    private int medicareNumber;
    private Date medicareValidTo;
    private int individualReferenceNumber;

    //getters, setters and other methods
}
```

These data is related to the construct **"Address"**

These data is related to the construct **"Medicare"**

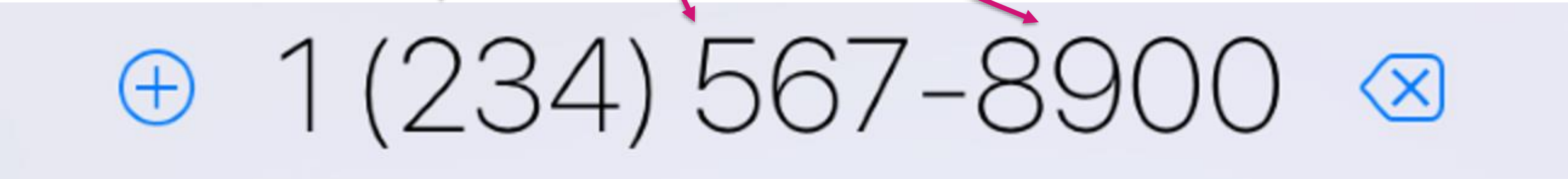
ADDRESING
PRIMITIVE OBSESSION

```java
public class SavingsAccount {
    private double balance;
    private int accountNumber;
    private String accountName;
    private Address address;
    private MedicareInfo medicare;
}
```

```java
public class Address {
    private String streetName;
    private String streetNumber;
    private int zipCode;
    private String city;
    private String state;
    private String country;
}
public class MedicareInfo {
    private int medicareNumber;
    private Date medicareValidTo;
    private int individualReferenceNumber;
}
```

WHY IS THIS CODE “SMELLY”?

```java
public class Phone {
    private final String thePhoneNumber;

    public String getAreaCode() {
        return thePhoneNumber.substring(0, 3);
    }

    public String getPrefix() {
        return thePhoneNumber.substring(3, 6);
    }

    public String getNumber() {
        return thePhoneNumber.substring(6, 10);
    }
}
```



WHY IS THIS CODE “SMELLY”?

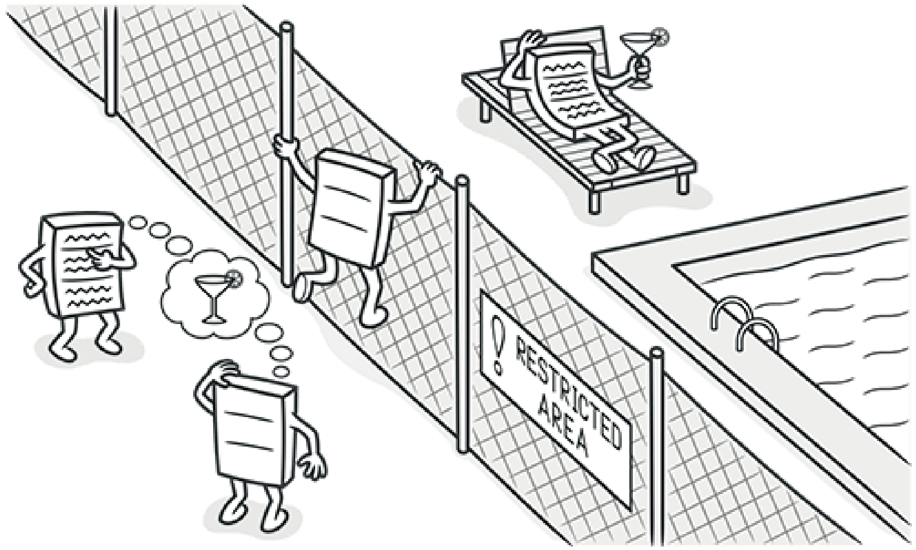
```java
public class Phone {
    private final String thePhoneNumber;

    public String getAreaCode() {
        return thePhoneNumber.substring(0, 3);
    }

    public String getPrefix() {
        return thePhoneNumber.substring(3, 6);
    }

    public String getNumber() {
        return thePhoneNumber.substring(6, 10);
    }
}
```

```java
public class Customer {
    private Phone mobilePhone;

    public String getMobilePhoneNumber() {
        return "(" +
            mobilePhone.getAreaCode() + ") " +
            mobilePhone.getPrefix() + "-" +
            mobilePhone.getNumber();
    }
}
```

THIS CAN BE RELATED TO THE SMELL
FEATURE ENVY!

```java
public class Phone {
    private final String thePhoneNumber;

    public String getAreaCode() {
        return thePhoneNumber.substring(0, 3);
    }

    public String getPrefix() {
        return thePhoneNumber.substring(3, 6);
    }

    public String getNumber() {
        return thePhoneNumber.substring(6, 10);
    }
}
```

```java
public class Customer {
    private Phone mobilePhone;

    public String getMobilePhoneNumber() {
        return "(" +
            mobilePhone.getAreaCode() + ") " +
            mobilePhone.getPrefix() + "-" +
            mobilePhone.getNumber();
    }
}
```



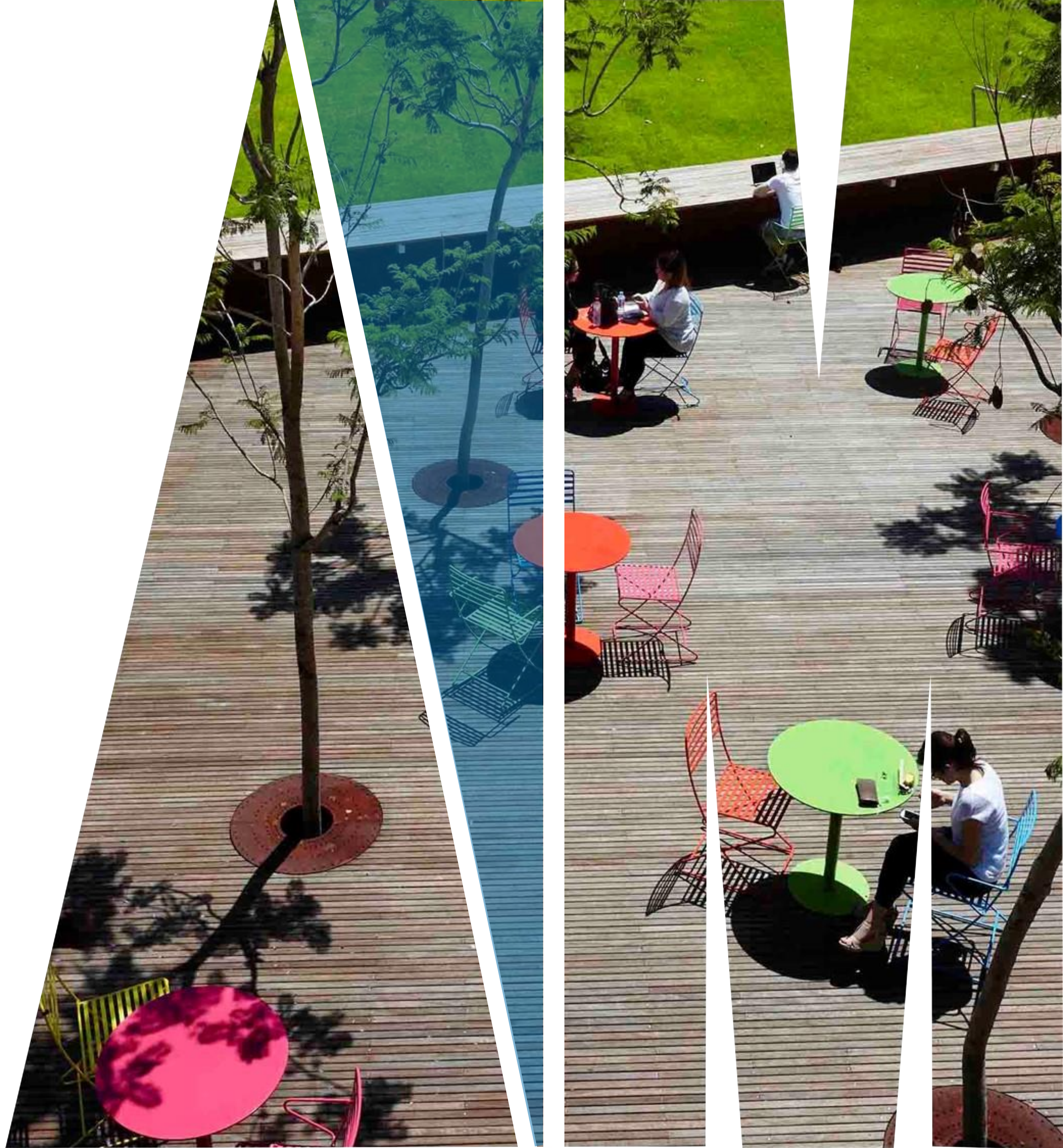
HOW TO ADDRESS FEATURE ENVY?

```
public class Phone {
    private final String thePhoneNumber;

    public String getAreaCode() {
        return thePhoneNumber.substring(0, 3);
    }

    public String getPrefix() {
        return thePhoneNumber.substring(3, 6);
    }

    public String getNumber() {
        return thePhoneNumber.substring(6, 10);
    }

    public String toFormattedString() {
        return "(" + getAreaCode() + ") " + getPrefix() + "-" + getNumber();
    }
}
```

```
public class Customer {
    private Phone mobilePhone;

    public String getMobilePhoneNumber() {
        return mobilePhone.toFormattedString();
    }
}
```

MONASH
University

Thanks



MONASH
University