Nachdenkzettel Clean Code

1. Klassenexplosion (Schwierig..)

class Formularfeld;
class Textfeld extends Formularfeld;
class Zahlfeld extends Formularfeld;
class TextUndZahlFeld extends Formularfeld;
class TextfeldOCR extends Textfeld;
class ZahlfeldOCR extends Zahlfeld;
class TextUndZahlFeldOCR extends TextUndZahlFeld;
class TextfeldSonderZ extends TextUndZahlFeld;
class TextfeldOCRSonderZ extends TextUndZahlFeldOCR;
class .....

Jede weitere Eigenschaft oder Spezialisierung führt zu vielen neuen Klassen durch Kombination. Die Folge ist explosives Anwachsen der Zahl der Klassen mit identischem Code. (Lösung?)

- To resolve the issue one could use interfaces. A „Formularfeld“ in this case does not seem to do anything but be the parent class of fields you can actually put something into – so, it could be an interface.
- If you don’t need fields for just number or text input you could even implement Textfeld and Zahlfeld as interfaces and implement them in TextUndZahlFeld as well as the OCR-Variants.
- If you do so, you could use a factory in order to make working with the classes easier.

2. Der verwirrte und der nicht-verwirrte Indexer
was genau unterscheidet die beiden Indexer? Wieso ist der eine „verwirrt“?

The indexer classes can be found here:

- Confused: https://gitlab.mi.hdm-stuttgart.de/kriha/se2_examples/-/blob/master/src/main/java/se2examples/uncleanExamples/constraintProblems/ConfusedIndexer.java

- Unconfused: https://kriha.pages.mi.hdm-stuttgart.de/se2_lecture/CleanCodeInOO.html#24

They differenciate in the way they implement the concept of languageID and isoCode. While the UnconfusedIndexer carefully checks the input to its constructors, the ConfusedIndexer doesn’t even have constructors. It has setter-methods for both the languageID and the isoCode – which can be changed independently. For example, it is possible to have the isoCode „de“ while having the langID for the US. Also, its states are mutable – so, when passed to another method, it could change its languageID while you still think it is set to German.
This is not the case with the UnconfusedIndexer. It maps the given isoCode to a languageID, so that the method index always uses the correct languageID.
That’s why the ConfusedIndexer is confused – its languageID can have a discrepancy from the isoCode.

3. Korrekte Initialisierung und Updates von Objekten

public class Address {

	private String City;
	private String Zipcode;
	private String Streetname;
	private String Number;

	public void setCity (String c) {
		City = c;
	}
	public void setZipcode (String z) {
		Zipcode = z;
	}
}

Wie initialisieren Sie Address richtig? Wie machen Sie einen korrekten Update der Werte?

```
public class Address {

    private String City;
    private String Zipcode;
    private String Streetname;
    private String Number;

    public void changeAddress (String City, String Zipcode, String Streetname, String Number) {
        this.City = City;
        this.Zipcode = Zipcode;
        this.Streetname = Streetname;
        this.Number = Number;
    }
}
```

4. Kapselung und Seiteneffekte

```
public class Person {

        public Wallet wallet = new Wallet();
        int balance = 0;

        public Wallet getWallet(void) {
                return wallet;
        }

        public addMoney(int money) {
                wallet.add(money);
                balance = wallet.size();

        public int getBalance() {
                return balance;
        }
}
```

Reparieren Sie die Klasse und sorgen Sie dafür, dass die Gültigkeit der Objekte erhalten bleibt und keine Seiteneffekte auftreten.

```java
public class Person {

    private final Wallet wallet = new Wallet();

    public Wallet getWallet() {
        Wallet tempWallet = new Wallet();
        tempWallet.add(wallet.size());
        return new Wallet();
    }

    public void addMoney(int money) {
        wallet.add(money);
    }

    public int getBalance() {
        return wallet.size();
    }
}
```