Aufgabe 5

(a) Nennen Sie vier Programmierparadigmen.

- Imperative Programmierung
- Prozedurale Programmierung
- Funktionale Programmierung
- Objektorientiere Programmierung

[a]

[a] https://de.wikipedia.org/wiki/Programmierparadigma#Strukturierte_Programmierung

(b) Erlautern Sie die Begriffe Overloading und Overriding, sowie deren Unterschiede.

- Beim Überladen muss die Methode eine andere Signatur haben, beim Überschreiben dieselbe Signatur.
- Die Intention beim Überladen ist, Methode zu erweitern, beim Überschreiben die Methode vollständig zu ersetzen.
- Überladen der Methode wird verwendet, um den Polymorphismus der Kompilierzeit zu erreichen. Das Überschreiben der Methode wird verwendet, um einen Laufzeit-Polymorphismus zu erreichen.

  In Methoden- / Funktionsüberladung weiß der Compiler, welches Objekt welcher Klasse zum Zeitpunkt der Kompilierung zugewiesen wurde. In der Methodenüberschreibung sind diese Informationen jedoch erst zur Laufzeit bekannt.
- Das Überladen von Methoden findet in derselben Klasse statt, während das Überschreiben in einer von einer Basisklasse abgeleiteten Klasse stattfindet.

[a]

[a] https://gadget-info.com/difference-between-method-overloading

(c) Erlautern Sie, wie sich zentrale und dezentrale Versionsverwaltung unterscheiden.

Beim der dezentralen Versionsverwaltung hat jeder EntwicklerIn die komplette Repository mit seiner kompletten History lokal gespeichert und kann diese dann mit anderen Repositories abgleichen.

Bei der zentralen Versionsverwaltung gibt es einen zentralen Server, der die komplette History vorhält.

(d) Erstellen Sie ein Sequenzdiagramm zur Methode `main` der Klasse Webshop.

Hinweise:

- Arithmetische Operationen müssen nicht weiter aufgelöst werden.
- Listenoperationen müssen nicht explizit dargestellt werden.
- Auf das Zeichnen einer passiven Lebenslinie muss nicht geachtet werden.
- Übertragen Sie das untenstehende Diagramm als Ausgangspunkt in Ihren Bearbeitungsbogen.

```
3  public class Webshop {
4    public static void main(String[] args) {
5      Bestellung b1 = new Bestellung();
6
7      // ab hier soll modelliert werden
8      Artikel a1 = new Artikel();
9      a1.setName("Taschenrechner");
10     a1.setPrice(10);
11
12     b1.addArticle(a1);
13
14     Bestellung b2 = new Bestellung();
15     Artikel a2 = new Artikel();
16     a2.setName("Lineal");
17     a2.setPrice(2.5);
18
19     Artikel a3 = new Artikel();
20     a3.setName("Bleistift");
21     a3.setPrice(0.7);
22
23     b2.addArticle(a3);
24     b1.addArticle(a2);
25
26     b1.getSize();
27
28     b2.getPrice();
29   }
30 }
```

```
3  @SuppressWarnings({"unused"})
4  public class Artikel {
5    private String name;
6
7    private double price;
8
9    public void setName(String name) {
10     this.name = name;
11   }
12
13   public void setPrice(double price) {
14     this.price = price;
15   }
16
17   public double getPrice() {
18     return price;
19   }
20 }
```

```
3  import java.util.ArrayList;
4  import java.util.List;
5
6  public class Bestellung {
7    private List<Artikel> articles;
```

```
8     // Anzahl an Artikeln
9     private int size = 0;
10    // Gesamtpreis der Bestellung
11    private double price = 0;
12
13    public Bestellung() {
14      articles = new ArrayList<>();
15    }
16
17    public void addArticle(Artikel article) {
18      // muss nicht weiter aufgelöst werden, siehe Hinweise
19      articles.add(article);
20      size++;
21      // muss nicht weiter aufgelöst werden, siehe Hinweise
22      price = article.getPrice() + price;
23    }
24
25    public int getSize() {
26      return size;
27    }
28
29    public double getPrice() {
30      return price;
31    }
32  }
```