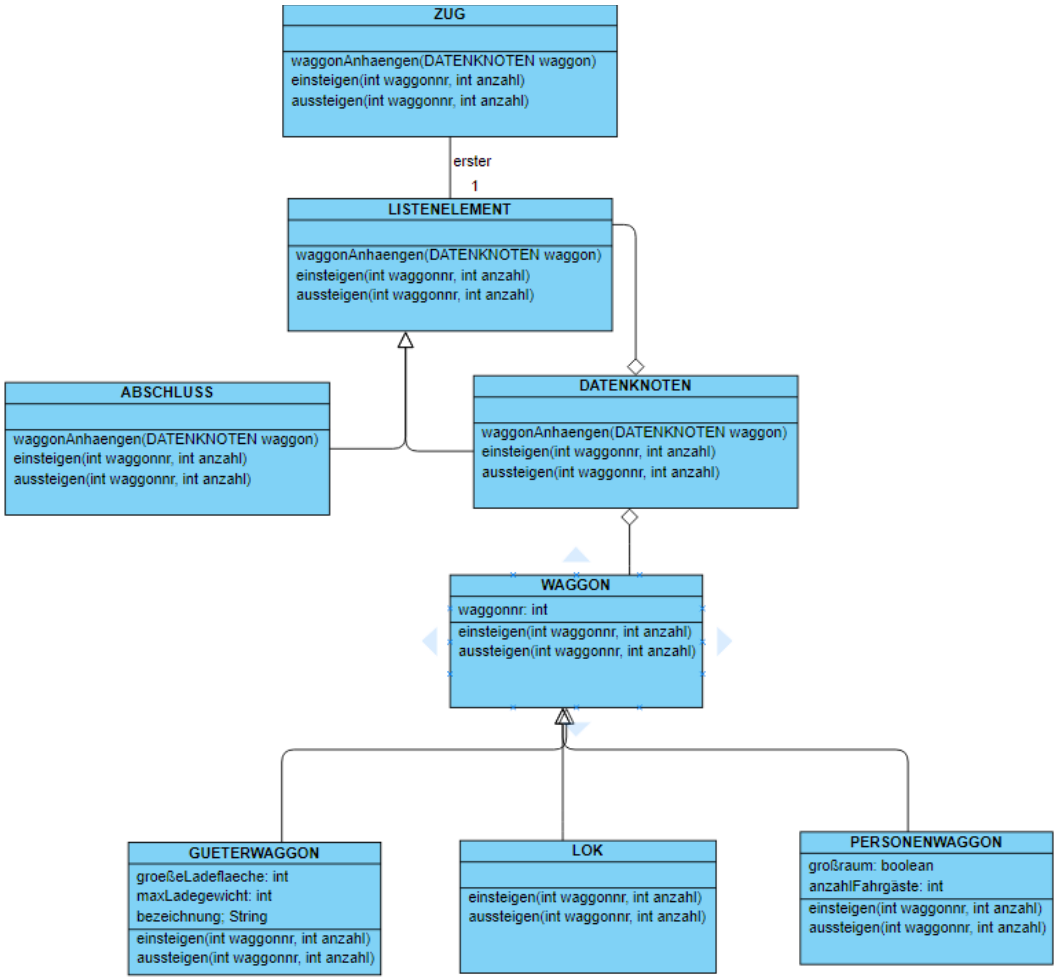
Aufgabe 1 a,


ZUG
waggonAnhaengen(DATENKNOTEN waggon)
einsteigen(int waggonnr, int anzahl)
aussteigen(int waggonnr, int anzahl)
erster
1
LISTENELEMENT
waggonAnhaengen(DATENKNOTEN waggon)
einsteigen(int waggonnr, int anzahl)
aussteigen(int waggonnr, int anzahl)
ABSCHLUSS
waggonAnhaengen(DATENKNOTEN waggon)
einsteigen(int waggonnr, int anzahl)
aussteigen(int waggonnr, int anzahl)
DATENKNOTEN
waggonAnhaengen(DATENKNOTEN waggon)
einsteigen(int waggonnr, int anzahl)
aussteigen(int waggonnr, int anzahl)
WAGGON
waggonnr: int
einsteigen(int waggonnr, int anzahl)
aussteigen(int waggonnr, int anzahl)
GUETERWAGGON
groeßeLadeflaeche: int
maxLadegewicht: int
bezeichnung; String
einsteigen(int waggonnr, int anzahl)
aussteigen(int waggonnr, int anzahl)
LOK
einsteigen(int waggonnr, int anzahl)
aussteigen(int waggonnr, int anzahl)
PERSONENWAGGON
großraum: boolean
anzahlFahrgäste: int
einsteigen(int waggonnr, int anzahl)
aussteigen(int waggonnr, int anzahl)

Aufgabe 1 b,

Implementierung:

Klasse ZUG:

```
public class ZUG
{
	public LISTENELEMENT erster;
	public ZUG() {
		erster = new ABSCHLUSS();
	}
	public void waggonAnhaengen(DATENKNOTEN wagen)

		erster = erster.waggonAnhaengen(wagen);
	}

	public void einsteigen(int waggonnr, int anzahl) {
		erster.einsteigen(waggonnr, anzahl);
	}

	public void aussteigen(int waggonnr, int anzahl) {
		erster.aussteigen(waggonnr, anzahl);
	}

	public void auskunft() {
		erster.auskunft();
	}
}
```

Klasse ZUGENDE

```
public class ABSCHLUSS extends LISTENELEMENT {
	public ABSCHLUSS() {
	}
	public DATENKNOTEN waggonAnhaengen(DATENKNOTEN wagen) {
		wagen.naechster=this;
		return wagen;
	}
	public void auskunft() {
	}

	public void einsteigen(int waggonnr, int anzahl) {
		System.out.println("Waggonnummer nicht gefunden!");
	}

	public void aussteigen(int waggonnr, int anzahl) {
		System.out.println("Waggonnummer nicht gefunden!");
	}
}
```

Klasse ZUGTEIL

```
public abstract class LISTENELEMENT {
	public abstract DATENKNOTEN waggonAnhaengen(DATENKNOTEN wagen);
	public abstract void auskunft();
	public abstract void einsteigen(int waggonnr, int anzahl);
	public abstract void aussteigen(int waggonnr, int anzahl);
}
```

Klasse ECHTERZUGTEIL

```
public class DATENKNOTEN extends LISTENELEMENT {
	public LISTENELEMENT naechster;
	public WAGGON inhalt;
	public DATENKNOTEN(LISTENELEMENT folgender, WAGGON wagen) {
		naechster = folgender;
		inhalt = wagen;
	}

	public DATENKNOTEN waggonAnhaengen(DATENKNOTEN wagen) {
		naechster = naechster.waggonAnhaengen(wagen);
		return this;
	}

	public void auskunft() {
		this.inhalt.auskunft();
		naechster.auskunft();
	}

	public void einsteigen(int waggonnr, int anzahl) {
		if (this.inhalt.waggonnummer == waggonnr) {
			this.inhalt.einsteigen(waggonnr, anzahl);
		} else {
			naechster.einsteigen(waggonnr, anzahl);
		}
	}

	public void aussteigen(int waggonnr, int anzahl) {
		if (this.inhalt.waggonnummer == waggonnr) {
			this.inhalt.aussteigen(waggonnr, anzahl);
		} else {
			naechster.aussteigen(waggonnr, anzahl);
		}
	}
}
```

Klasse WAGGON

```
public abstract class WAGGON {
        int waggonnummer;
        public abstract void auskunft();
        public abstract void einsteigen(int waggonnr, int anzahl);
        public abstract void aussteigen(int waggonnr, int anzahl);
}
```

Klasse GUETERWAGGON

```
public class GUETERWAGGON extends WAGGON {
        int groeßeLadeflaeche;
        int maxLadegewicht;
        int Ladegewicht;
        String bezeichnung;
        public GUETERWAGGON(String bezeichnung, int groeßeLadefl, int
        maxLadegew, int waggonnr ) {
                waggonnummer = waggonnr;
                groeßeLadeflaeche = groeßeLadefl;
                maxLadegewicht = maxLadegew;
                Ladegewicht = 0;
        }

        public void auskunft() {
                System.out.println("Ich bin ein Güterwaggon mit Waggonnummer " +
                        waggonnummer+ ,  " und beinhalte " + Ladegewicht+ " kg
                Ladung.");
        }

        public void einsteigen(int waggonnr, int anzahl) {
                System.out.println("Ich bin ein Güterwaggon, kein Personenwaggon");
        }

        public void aussteigen(int waggonnr, int anzahl) {
                System.out.println("Ich bin ein Güterwaggon, kein Personenwaggon");
        }
}
```

Klasse LOK

```
public class LOK extends WAGGON {
        public LOK(int waggonnr) {
                waggonnummer = waggonnr;
        }

        public void auskunft() {
                System.out.println("Ich bin eine Lok mit Waggonnummer" +
                waggonnummer +          ". “ );
        }

        public void einsteigen(int waggonnr, int anzahl) {
```

```
            System.out.println("Ich bin eine Lok und kein Personenwaggon");
        }

        public void aussteigen(int waggonnr, int anzahl) {
            System.out.println("Ich bin eine Lok und kein Personenwaggon");
        }
}
```

Klasse PERSONENWAGGON

```
public class PERSONENWAGGON extends WAGGON {
        boolean großraum;
        int anzahlFahrgaeste;
        public PERSONENWAGGON(int waggonnr, boolean großra) {
            waggonnummer = waggonnr;
            großraum = großra;
            anzahlFahrgaeste = 0;
        }

        public void auskunft() {
            System.out.println("Ich bin ein Personenwaggon mit Waggonnummer"
            + waggonnummer + " und beinhalte " + anzahlFahrgaeste +
            "Personen.");
        }

        public void einsteigen(int waggonnr, int anzahl) {
            anzahlFahrgaeste = anzahlFahrgaeste + anzahl;
            System.out.println("Jetzt im Waggonnr:"+waggonnr + “, “
                + anzahlFahrgaeste “ Fahrgaeste“);
        }

        public void aussteigen(int waggonnr, int anzahl) {
            if (anzahlFahrgaeste >= anzahl) {
                anzahlFahrgaeste = anzahlFahrgaeste - anzahl;
                System.out.println("Jetzt im Waggonnr:"+waggonnr + “, “
                    + anzahlFahrgaeste “ Fahrgaeste“);
            } else {
                System.out.println("Es können maximal " + anzahlFahrgaeste + "
                    aussteigen!");
            }
        }
}
```

2. Didaktische Reflexion a,

Zweck:
Verständnis und Veranschaulichung einer rekursiven Datenstruktur

Vorkenntnisse:
Die SuS haben bereits unterschiedliche abstrakte Klassen kennengelernt. Zusätzlich wissen sie wie man eine abstrakte Klasse erstellt und welche Eigenschaften sie hat.

Die SuS können eine Vererbung einer Klasse in Java-Syntax übersetzen. Ebenso wissen sie welche Einfluss die Polymorphie auf die gegebenen Klassen hat.

Die SuS können durch eine Ausgabekonsole, sinnvolle Fehlermeldungen an den Nutzer weitergeben.

Lernziele:
- Die SuS können ein geeignetes Klassenmodel einer rekursiven Datenstruktur zu einem gegebenen Kontext entwickeln.
- Die SuS können mit Hilfe eines Klassenmodels eine Implementierung einer rekursiven Datenstruktur ausarbeiten.
- Die SuS können eine Fehlerbehandlung mit für den Benutzer hilfreichen Ausgaben im Kontext einer rekursiven Datenstruktur erstellen.

Aufgabenteil b,

Genereller Zeitbedarf sind hierbei 2 Unterrichtsstunden. Im besten Fall wäre dieses Thema in einer Doppelstunde zu bearbeiten.

Zeitbedarf	**Arbeitsphase**	**Sozialform**	**Ergebnissicherung**
Kognitive Aktivierung 5 Minuten	Vorstellung des Themas mit Hilfe einer echten Modelleisenbahn	Klassengespräch	
Klärung der Aufgabestellung 5 Minuten	Die Aufgabestellung wird der Klasse präsentiert. (Nach Möglichkeit Ideen der Klasse eingebaut)	Klassengespräch	
Modellierung 20 Minuten	Klassendiagramm erstellen	Partnerarbeit	Abschließend an der Dokumentenkamera von einer Gruppe. Jeder SuS hat nach dieser Phase ein gültiges Klassendiagramm im Heft
Wiederholung 10 Minuten	Hilfsmittel, welche zum Implementieren benötigt werden, werden wiederholt	Klassengespräch	
Implementierung 45 Minuten	SuS implementieren das Programm	Partnerarbeit (Klassen können aufgeteilt werden)	Lehrer unterstützt die SuS
Abschluss 5 Minuten	Reflexion	Klassengespräch	Das vorhandene Programm wird hochgeladen und je nach Möglichkeit auch das Klassendiagramm