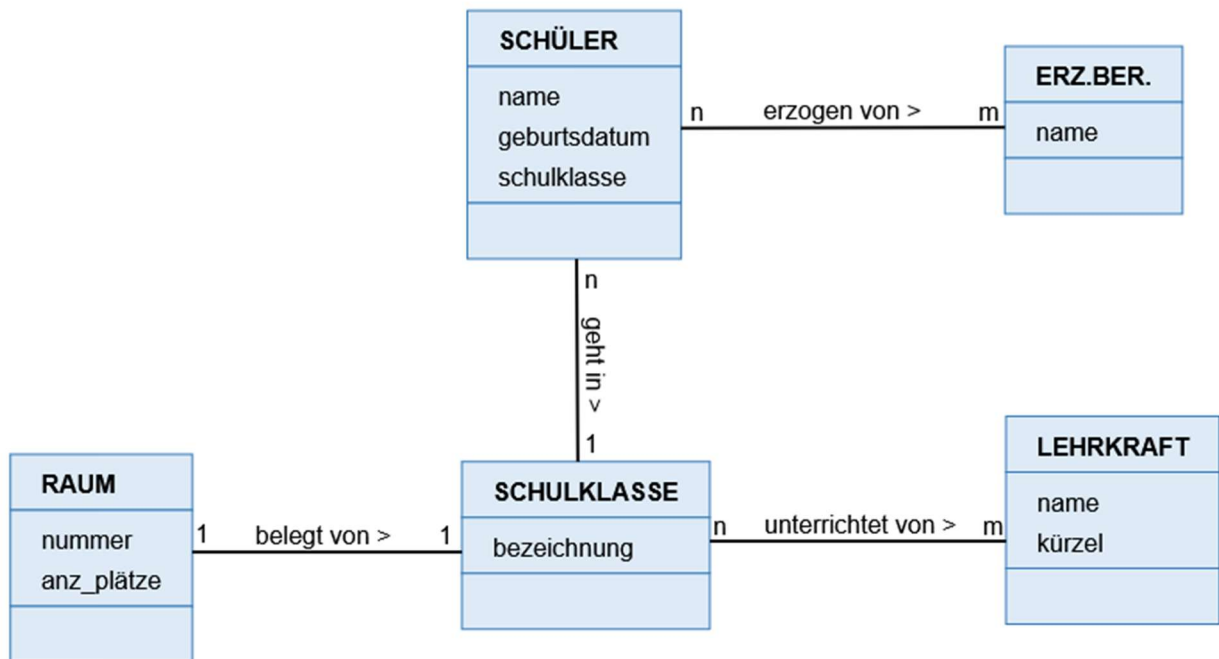
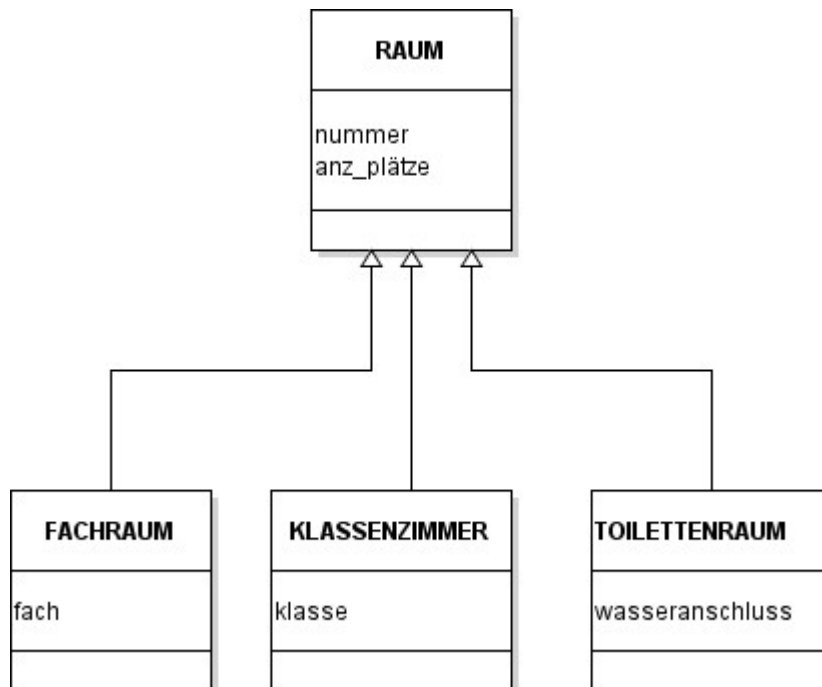


## Lösung:

### 1. Klassendiagramm



2. z.B. FACHRAUM, TOILETTENRAUM, KLASSENZIMMER als Unterklassen von RAUM. (PERSON als Oberklasse von SCHÜLER, LEHRKRAFT, ERZ.BER. wäre eher Generalisierung)



### 3.e

Polymorphismus bezeichnet die Eigenschaft, dass ein Referenzattribut nicht nur Objekte der als Datentyp angegebenen Klasse referenzieren kann, sondern auch Objekte aller zu dieser Klasse existierenden Unterklassen. Das Feld `räume` kann somit nicht nur Objekte der Klasse `RAUM`, sondern auch Objekte der Klasse `KLASSENRAUM` und `FACHRAUM` verwalten. Werden Methoden in Unterklassen überschrieben, so werden bei Aufruf die Methoden dieser Unterklasse ausgeführt.

```
class RAUM{
    protected String nummer;        //was ist
    protected int anzahlPlätze; //sinnvoll?

    RAUM(String nummer, int anzahlPlätzeNeu){
        this.nummer = nummer;
        this.anzahlPlätze = anzahlPlätzeNeu;
    }

    void setNummer(String nummer){
        this.nummer = nummer;
    }

    String getNummer(){
        return nummer;
    }

    void setAnzahlPlätze(int anzahlNeu){
        this.anzahlPlätze = anzahlNeu;
    }

    int getAnzahlPlätze(){
        return anzahlPlätze;
    }

    void infosAusgeben(){
        System.out.print("Raumnummer: "+ nummer);
        System.out.println(" Anzahl Plätze: "+ anzahlPlätze);
    }
}
```

```

class KLASSENZIMMER extends RAUM{
    private String klasse;

    KLASSENZIMMER(String nummer, int anzahl, String klasseNeu){
        super(nummerNeu, anzahlNeu);
        this.klasse = klasseNeu;
    }

    void setKlasse(String klasseNeu){
        this.klasse = klasseNeu;
    }

    String getKlasse(){
        return this.klasse;
    }

    @Override void infosAusgeben(){
        System.out.print("Raumnummer: "+ nummer);
        System.out.print(" Anzahl Plätze: "+ anzahlPlätze);
        System.out.println("Klasse: " + klasse);
        System.out.println();
    }
}

```

```

class FACHRAUM extends RAUM{
    private String fach;

    FACHRAUM(String nummerNeu, int anzahlNeu, String fach){
        super(nummerNeu, anzahlNeu);
        this.fach = fach;
    }

    void setFach(String fach){
        this.fach = fach;
    }

    String getFach(){
        return fach;
    }

    @Override void infosAusgeben(){
        super.infosAusgeben();
        System.out.println("Fach: " + fach);
    }
}

```

```

import java.util.ArrayList;

class SCHULVERWALTUNG{
    private ArrayList<RAUM> räume;

    SCHULVERWALTUNG(){
        räume = new ArrayList<RAUM>();
    }

    void raumHinzufügen(RAUM r){
        räume.add(r);
    }

    void infosAusgeben(){
        for(RAUM r: räume){
            r.infosAusgeben();
        }
    }
}

public class TESTKLASSE{
    private KLASSENZIMMER k1a1;
    private FACHRAUM fr1;
    private SCHULVERWALTUNG sv1;

    public TESTKLASSE(){
        k1a1 = new KLASSENZIMMER("A104", 30, "11b");
        fr1 = new FACHRAUM("B007", 32, "Biologie");
        sv1 = new SCHULVERWALTUNG();
    }

    public void testmethode(){
        sv1.raumHinzufügen(k1a1);
        sv1.raumHinzufügen(fr1);
        sv1.infosAusgeben();
    }
}

```