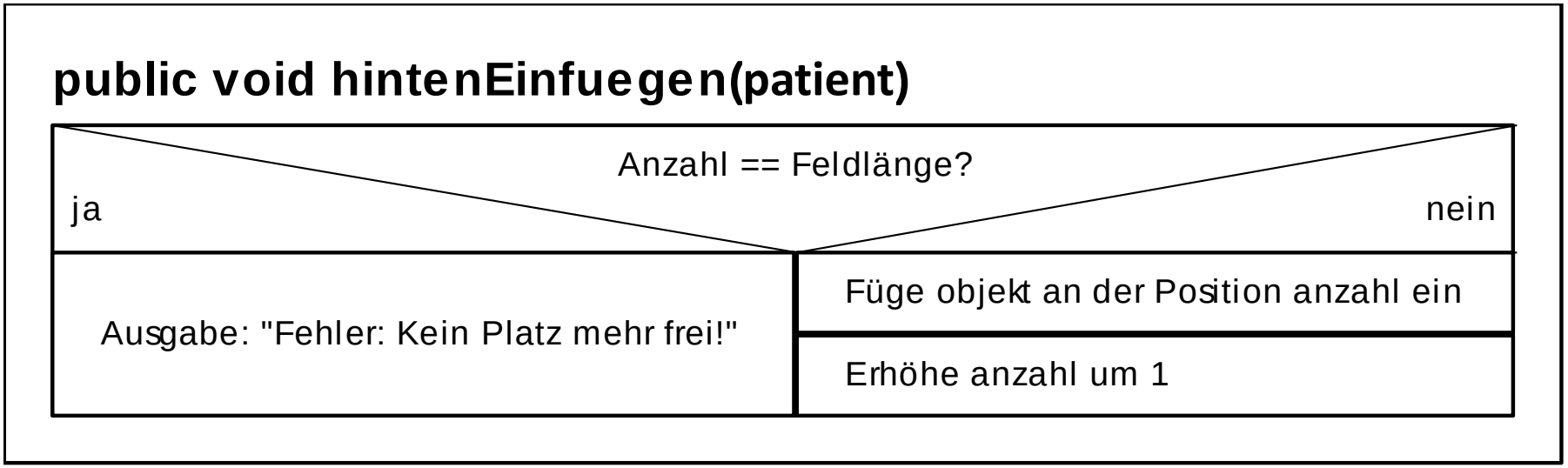
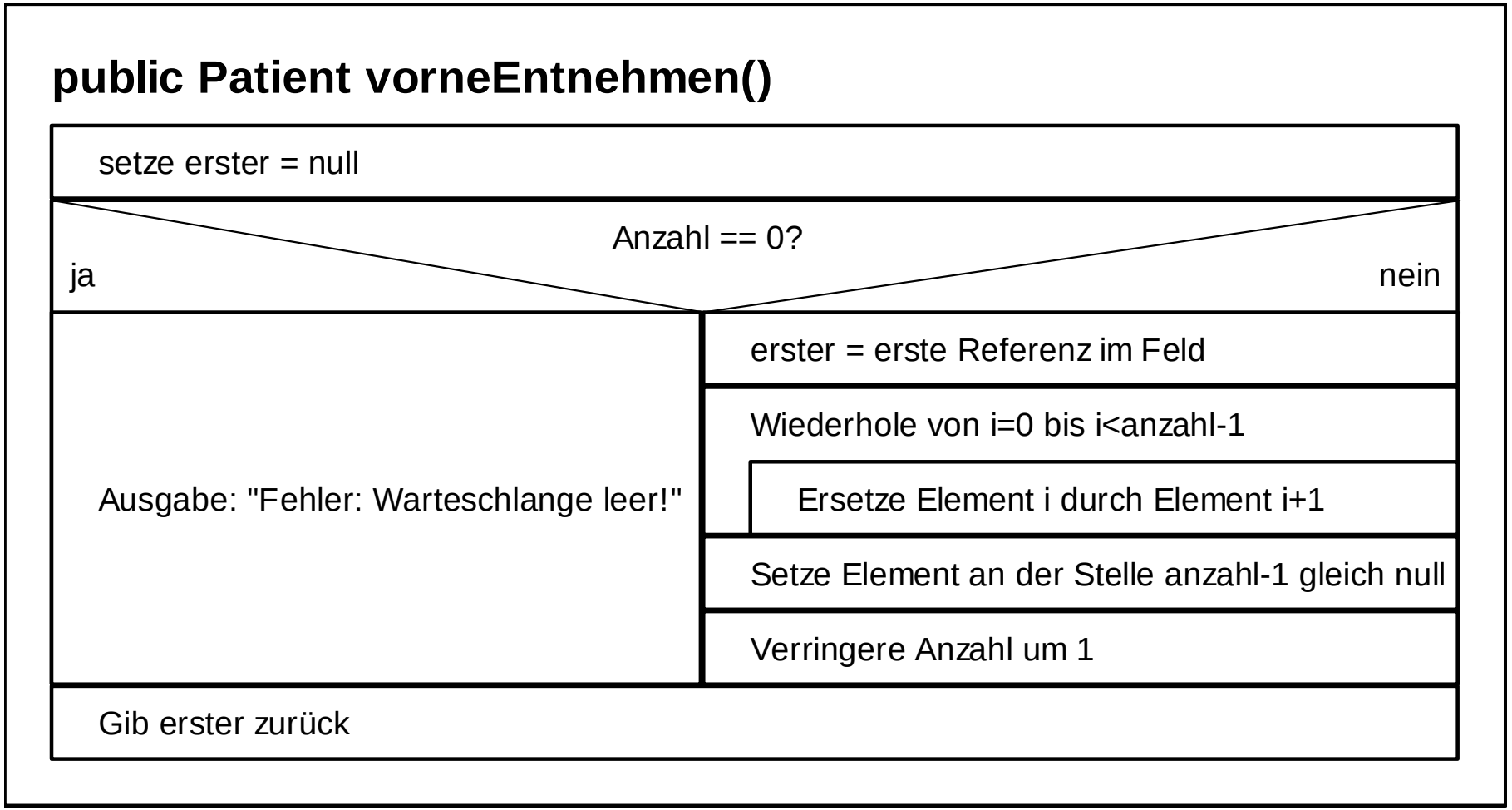
Struktogramme:

public void hintenEinfuegen(patient)



Struktogramme:

public Patient vorneEntnehmen()

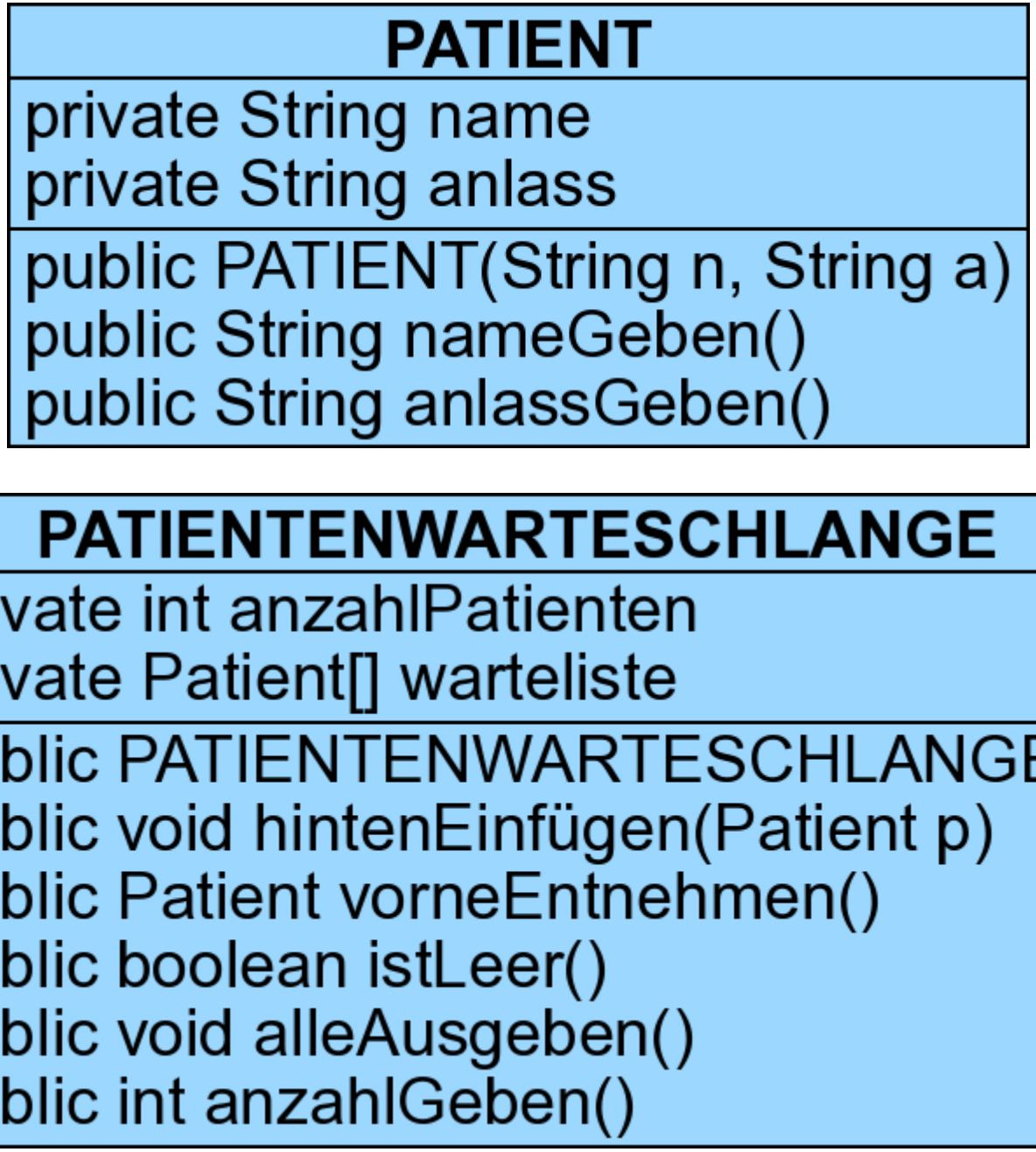


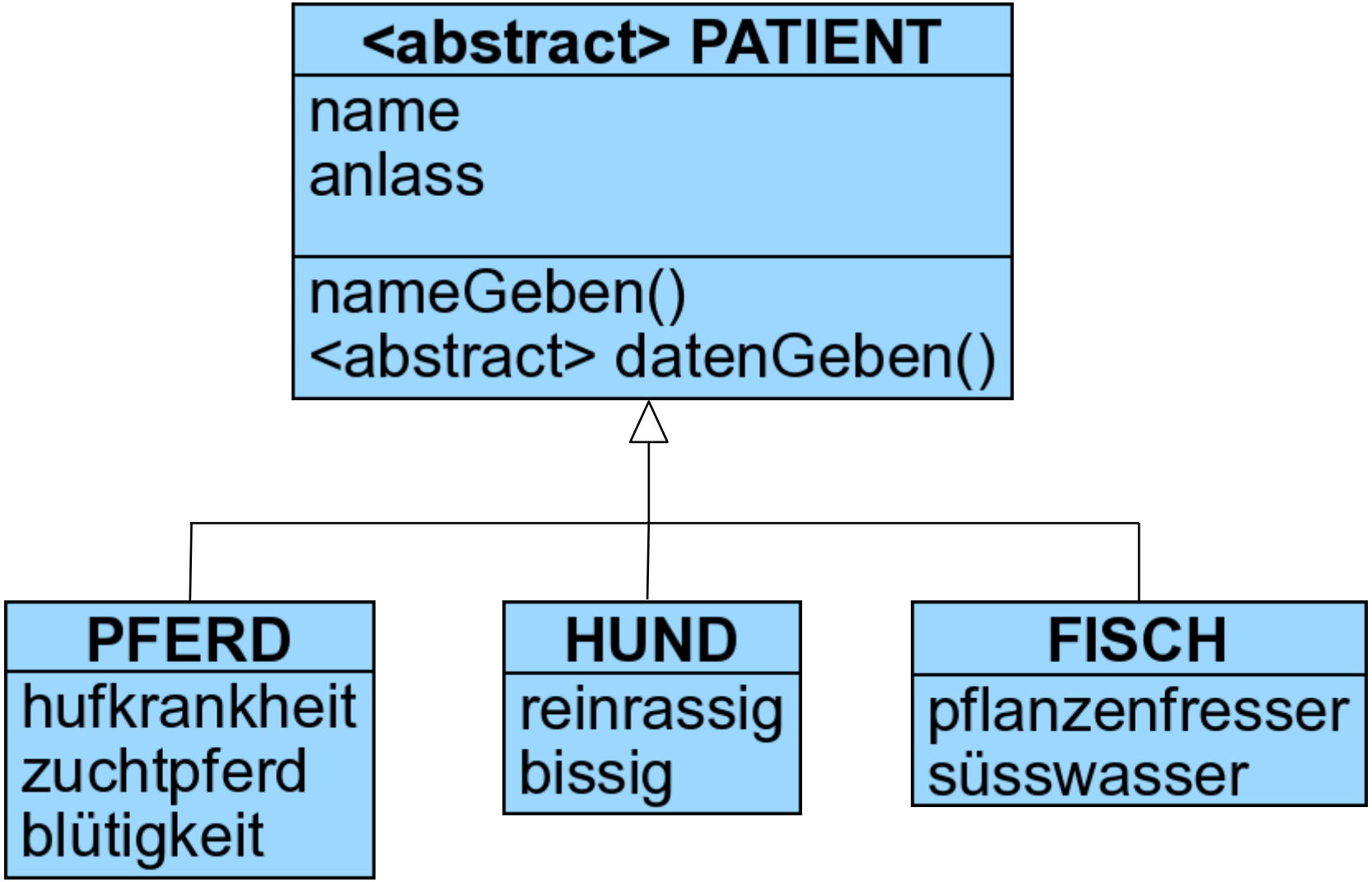
Eine Warteschlange ist eine zusammengesetzte Daten-struktur:
Es ist festgelegt, wie bestimmte Objekte zueinander angeordnet sind und welche Operationen für den Zugriff auf diese Objekte zur Verfügung stehen.

Bisher: Verwaltung **gleichartiger** Objekte

Wie können Objekte, die nicht einer Klasse zuzuordnen sind, in einer Warteschlange verwaltet werden?
z.B. Hund, Katze, Goldfisch als Patienten beim Tierarzt

➔ Verwendung abstrakter Klassen


PATIENT
private String name
private String anlass
public PATIENT(String n, String a)
public String nameGeben()
public String anlassGeben()
PATIENTENWARTESCHLANGE
private int anzahlPatienten
private Patient[] warteliste
public PATIENTENWARTESCHLANGE()
public void hintenEinfügen(Patient p)
public Patient vorneEntnehmen()
public boolean istLeer()
public void alleAusgeben()
public int anzahlGeben()



Verändere die Patientenverwaltung so, dass sie zu einer Tierarztpraxis passt.

```
public abstract class Patient {
    protected String name;
    protected String anlass;

    public Patient(String n, String a) {
        name = n;
        anlass = a;
    }

    public String nameGeben(){
        return name;
    }

    public String anlassGeben(){
        return anlass;
    }

    public abstract String datenGeben();
}
```

```
public class Pferd extends Patient{

    private String hufkrankheit;
    private boolean zuchtpferd;
    private String bluetigkeit;

    public Pferd(String n, String a, boolean zpf, String bl){
        super(n, a);
        hufkrankheit = "keine";
        zuchtpferd = zpf;
        bluetigkeit = bl;
    }

    public String datenGeben() {
        return "Pferd: " + name + ", Anlass: " + anlass  +
                    ", Hufkrankheit: " + hufkrankheit +
                    ", Zuchtpferd: " + zuchtpferd + ", " +bluetigkeit;
    }
```

```
public class Hund extends Patient{

    private boolean reinrassig;
    private boolean bissig;

    public Hund(String n, String a, boolean rr, boolean b) {
        super(n, a);
        reinrassig = rr;
        bissig = b;
    }

     public String datenGeben() {
         return "Hund: " + name + ", Anlass: " + anlass  +
                     ", reinrassig: " + reinrassig + ", bissig: " + bissig;
    }
}
```

```
public class Fisch extends Patient{

    private boolean pflanzenfresser;
    private boolean suesswasser;

    public Fisch(String n, String a, boolean pfr, boolean sw) {
        super(n, a);
        pflanzenfresser = pfr;
        suesswasser = sw;
    }

     public String datenGeben() {
         return "Fisch: " + name + ", Anlass: " + anlass  +
                     ", Pflanzenfresser: " + pflanzenfresser +
                     ", Suesswasser: " + suesswasser;
    }
}
```

Was muss in der Klasse Patientenwarteschlange geändert werden?

Nichts!!!☺

Allenfalls eine kleine Änderung in alleAusgeben() wäre sinnvoll:

```
public void alleAusgeben(){
        System.out.println("Warteliste:");
        for(int i=0; i<anzahlPatienten; i++){
            System.out.println(warteliste[i].datenGeben());
        }
    }
```

Teste dein Programm, indem du entsprechende Objekte erzeugst, am besten in einer eigenen Testklasse oder dir die entsprechende Datei aus dem Mebis-Kurs herunterlädst.

An welcher Stelle werden hier Patienten erzeugt? Gibt es eine andere vielleicht sogar sinnvollere Möglichkeit?

Welche Nachteile hat die Implementierung einer Warte-schlange mit Hilfe eines Feldes?

- Die Feldlänge wird vorab festgelegt und kann nicht mehr verändert werden.
- Beim Entfernen eines Elements müssen alle anderen um eine Position nach vorne geschoben werden (zeitaufwändig!) .

Lösung:
Dynamische Datenstrukturen, die je nach Bedarf Wachsen oder Schrumpfen können.