S. 59/1 Ahnengalerie (kein geordneter Binärbaum)

Einfüge-Methoden werden erst bei geordneten Bäumen näher behandelt, daher hier "Setzen per Hand" in einer Testklasse.

- Binaerbaum
  Attribut, Konstruktor, Methode wurzelSetzen
- Baumelement
  Gebe-Methoden
- Datenknoten
  Attribute, Konstruktor, Gebe-Methoden
- Abschluss
- Datenelement
  Attribute (vorname, nachname, alter), Konstruktor, datenGeben()
- Testablauf
  personenErzeugen, stammbaumSetzen (wurzel muss per Hand gesetzt werden),
  stammbaumAusgeben, personenZaehlen (erfordert entsprechende Ergänzungen in den anderen Klassen)

```
public class Binaerbaum{
    private Baumelement wurzel;

    public Binaerbaum(){
        wurzel = new Abschluss();
    }

    public void wurzelSetzen(Baumelement w){
        wurzel = w;
    }
}

public abstract class Baumelement {

    public abstract Baumelement naechsterLinksGeben();

    public abstract Baumelement naechsterRechtsGeben();

    public abstract Datenelement inhaltGeben();
}
```

```
public class Datenknoten extends Baumelement{

   private Baumelement naechsterLinks, naechsterRechts;
   private Datenelement inhalt;

   public Datenknoten(Baumelement nL, Baumelement nR, Datenelement inh){
      naechsterLinks = nL;
      naechsterRechts = nR;
      inhalt = inh;
   }

   public Baumelement naechsterLinksGeben(){
      return naechsterLinks;
   }

   public Baumelement naechsterRechtsGeben(){
      return naechsterRechts;
   }

   public Datenelement inhaltGeben(){
      return inhalt;
   }
}
```

```
public class Abschluss extends Baumelement{
   public Baumelement naechsterLinksGeben(){
      return this;
   }

   public Baumelement naechsterRechtsGeben(){
      return this;
   }

   public Datenelement inhaltGeben(){
      return null;
   }
}
```

```
public class Datenelement{
   private String vorname;
   private String nachname;
   private int alter;

   public Datenelement(String vn, String nn, int a){
      vorname = vn;
      nachname = nn;
      alter = a;
   }

   public String datenGeben() {
      return vorname+ " "+ nachname + ", "+alter;
   }
}
```

```
public class Testablauf {
   private Datenelement p1, p2, p3, p4, p5, p6, p7, p8;
   private Datenknoten dk1, dk2, dk3, dk4, dk5, dk6, dk7, dk8;
   private Binaerbaum ahnentafel;

   public void personenErzeugen(){
      p1 = new Datenelement("Donald", "Duck", 4);
      p2 = new Datenelement ("Helene", "Fischer", 40);
            …
      p8 = new Datenelement ("Konrad", "Zuse", 86);
   }
   public void StammbaumSetzen(){
      //Datenknoten erzeugen, dabei in der ältesten Generation anfangen oder Setzen-Methoden
                                                                              verwenden
      dk8 = new Datenknoten(new Abschluss(), new Abschluss(), p8);
      dk7 = new Datenknoten(new Abschluss(), new Abschluss(), p7);
      dk6 = new Datenknoten(new Abschluss(), new Abschluss(), p6);
      dk5 = new Datenknoten(new Abschluss(), new Abschluss(), p5);
      dk4 = new Datenknoten(dk8, new Abschluss(), p4);
      dk3 = new Datenknoten(dk6, dk7, p3);
      dk2 = new Datenknoten(dk4, dk5, p2);
      dk1 = new Datenknoten(dk2, dk3, p1);

      ahnentafel = new Binaerbaum();
      ahnentafel.wurzelSetzen(dk1);
   }
```

Im Testablauf:

```
public void StammbaumAusgeben(){
    ahnentafel.datenAusgeben();
}
```

In Binärbaum:

```
public void datenAusgeben(){
    System.out.println(wurzel.alleDatenGeben());
}
```

In Baumelement:

```
public abstract String  alleDatenGeben();
```

In Datenknoten:

```
public String alleDatenGeben(){
    return inhalt.datenGeben() + "\n"
        +naechsterLinks.alleDatenGeben()
        + naechsterRechts.alleDatenGeben();
}
```

In Abschluss:

```
public String alleDatenGeben(){
    return "";
}
```

Im Testablauf:

```
public void personenZaehlen(){
    System.out.println("Der Stammbaum besteht aus " +
        ahnentafel.anzahlElementeGeben() +" Personen.");}
```

In Binärbaum:

```
public int anzahlElementeGeben(){
    return wurzel.anzahlDatenknotenGeben();
}
```

In Baumelement:

```
public abstract int anzahlDatenknotenGeben();
```

In Datenknoten:

```
public int anzahlDatenknotenGeben(){
    return  1+naechsterLinks.anzahlDatenknotenGeben()
        +naechsterRechts.anzahlDatenknotenGeben(); }
```

In Abschluss:

```
public int anzahlDatenknotenGeben(){
    return 0;
}
```