Begriff „Array“

Mit „Array“ wird in der Informatik eine **nummerierte Liste** bezeichnet, in der man eine große Anzahl von Objekten verwalten kann. In einem Array kann man z.B. 50 Sprites eines Spiels, 100 Mitarbeitern einer Firma oder 20.000 Konten einer Bank speichern.

Hinweis: Das englische Wort „Ar**ray**“ betont man auf der 2. Silbe.
Es bedeutet soviel wie „Anordnung“.

Wir werden zu Beginn hauptsächlich mit Arrays arbeiten, die Zahlen (und nicht komplexe Objekte) enthalten. Das vereinfacht dann etwas später das Erarbeiten von Sortieralgorithmen.
Wie man Arrays für Objekte benutzt wird am Ende der Unterrichtsreihe erarbeitet.

Darstellung als nummerierte Liste

Arrays speichern Werte bzw. Objekte. Diese werden von 0 bis zum Ende durchnummeriert.
Die Nummer zu jedem Wert heißt **Index**. Dabei handelt es sich um eine natürliche Zahl (int).

Das folgende Beispiel enthält einen Lottotipp:
Die erste getippte Zahl ist die 7, die zweite die 12, die dritte die 25, usw.
Es gibt sechs Werte, also ist die Liste mit den Indizes von 0 bis 5 nummeriert.

Index	*0*	*1*	*2*	*3*	*4*	*5*
Wert	7	12	25	28	33	41

Ein anderes Beispiel: Die Namen einer WhatsApp-Gruppe sind in einem Array gespeichert:

Index	*0*	*1*	*2*	*3*	*4*
Wert	„Kati“	„Can“	„Max“	„Celine“	„John“

Besonderheiten von Arrays

In einem Array kann man nur Werte **eines** Datentyps (z.B. int) bzw. Objekte **einer** Klasse (z.B. Sprite) speichern, also nicht z.B. Zahlen und Texte gemischt.

Die „**Länge**“ des Arrays, also die Anzahl von Werten bzw. Objekten, die das Array aufnehmen kann, wird zu Beginn bei der Deklaration festgelegt und kann im Programmverlauf nicht geändert werden. Falls man zu Beginn des Programms nicht genau weiß, wie viele Objekte man speichern muss (es können z.B. Objekte im Verlauf dazukommen oder verschwinden) gibt es andere Möglichkeiten wie dynamische Listen, die später in der Q1 behandelt werden.

In Java gibt es außerdem die Besonderheit, dass für Arrays, die Zahlen enthalten, bei der Erzeugung alle Werte **automatisch auf 0** gesetzt werden (in anderen Sprachen, z.B. in C, ist das nicht so).

Arrays im Quellcode

Im Beispiel wird ein Array für Integer-Werte verwendet, um einen Lottotipp zu speichern.

```
public class Lottotipp
{
   private int[] tipp;

   public Lottotipp()
   {
      tipp = new int[6];
   }

   public void setBeispielTipp()
   {
      tipp[0] = 7;
      tipp[1] = 12;
      tipp[2] = 25;
      tipp[3] = 28;
      tipp[4] = 33;
      tipp[5] = 41;
   }
```

Bei der Deklaration geben die **eckigen Klammern** an, dass „tipp“ ein Array ist.

Arrays müssen mit **new** erzeugt werden. Dabei wird in den eckigen Klammern die Länge, d.h. die **Anzahl** von Werten, die das Array speichern kann, festgelegt.
„tipp“ speichert also 6 Integer-Werte.

In setBeispielTipp() wird den 6 Elementen des Arrays jeweils ein Wert zugewiesen.
Das erste Element, tipp[0], erhält den Wert 7, das zweite den Wert 12, usw.

Zum Sprachgebrauch:
„tipp am Index 0 wird der Wert 7 zugewiesen.“

Die einzelnen Elemente eines Arrays kann man im Quellcode wie Variablen verwenden.
Zum Beispiel in der Ausgabe auf der Konsole:

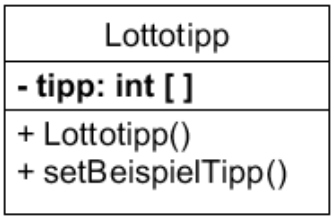
```
   public void ausgabe()
   {
      Console.println("Erste  Zahl: " + tipp[0]);
      Console.println("Zweite Zahl: " + tipp[1]);
      ...
```

Ebenso kann man einzelne Elemente von Arrays in Rechnungen verwenden (bei Lottozahlen hätte das eigentlich keine Anwendung, aber hier als Beispiel):

```
   public void summe()
   {
      int summe;
      summe = tipp[0] + tipp[1];
      ...
```

Arrays im Klassendiagramm

Arrays werden wie andere Attribute, nur mit den **eckigen Klammern** dargestellt. Das Attribut „tipp“ enthält also eine Liste von Integer-Werten, nicht nur einen Wert.



Die Anzahl der Werte wird nicht angegeben.

Verständnisaufgabe

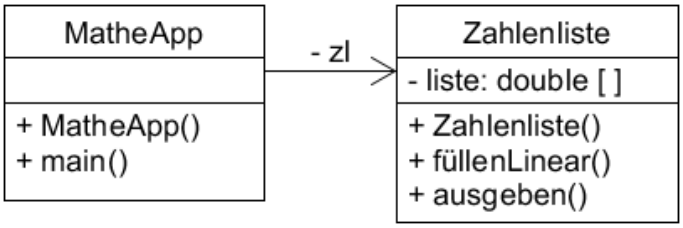
Lies den folgenden Quellcode und beantworte die Fragen:

1. Gib die Länge des Arrays „tipp“ an.
2. Gib den kleinsten und den größten Index des Arrays an.
3. Allgemein für ein Array der Länge **n**: was ist der kleinste, was der größte Index?
4. Gib die Werte des Arrays in Form einer Tabelle (mit den Indizes) nach Ablauf von main() an.
5. Was ist die Ausgabe auf der Konsole?

```
public class LottoTipp
{
   private int[] tipp;

   public LottoTipp()
   {
      tipp = new int[6];
   }

   public void setBeispielTipp()
   {
      tipp[0] = 15;
      tipp[1] = 17;
      tipp[2] = tipp[0] + tipp[1];
   }

   public void ausgeben()
   {
      Console.println(
         tipp[0] + " " + tipp[1] + " " + tipp[2] + " " +
         tipp[3] + " " + tipp[4] + " " + tipp[5]);
   }
}

public class Lotto
{
   LottoTipp lt;

   public Lotto()
   {
      lt = new LottoTipp();
   }

   public void main()
   {
      lt.setBeispielTipp();
      lt.ausgeben();
   }
}
```

Aufgabe 1

Die „MatheApp“ kann Folgen von Zahlen erzeugen. Implementiere beide Klassen im abgebildeten Klassendiagramm.



a) Deklariere das Attribut „liste“ als Array für **double**-Werte.
Im Konstruktor Zahlenliste() wird das Array mit Platz für **fünf** Werte erzeugt.

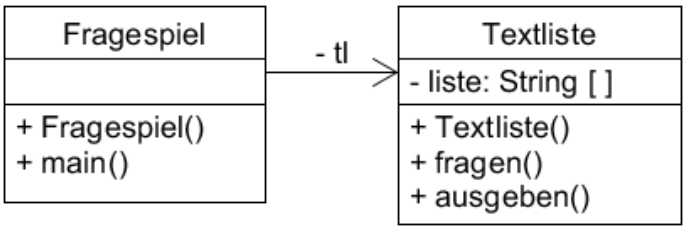
b) In der Methode füllenLinear() werden den fünf Elementen des Arrays die Werte von 1,0 bis 1,8 zugewiesen (in Abständen von 0,2).

c) Die Methode ausgeben() gibt jeden Wert des Arrays auf der Konsole aus.

d) Implementiere die Hauptklasse. Die main-Methode soll die Liste einmal füllen und ausgeben.

Aufgabe 2

Das „Fragespiel“ stellen der Benutzerin bzw. dem Benutzer Fragen und speichert die Antworten.



a) Deklariere das Attribut „liste“ als Array für **String**-Werte.
Im Konstruktor Textliste() wird das Array mit Platz für **vier** Werte erzeugt.

b) Die Methode fragen() stellt dem Benutzer vier Fragen (denke dir selbst welche aus).
Die Antworten, die der Benutzer eintippt, werden den vier Elementen des Arrays zugewiesen.
Tipp: verwende die Methode Console.readln(), um eine ganze Zeile mit Leerzeichen einzulesen.

c) Die Methode ausgeben() gibt die Antworten, die im Array gespeichert sind, aus.

d) Implementiere die Hauptklasse. Die main-Methode ruft einmal fragen() und ausgeben() auf.