ÜBUNGSSTUNDE W6

Einführung in die Programmierung

Ablauf

- Nachbesprechung der Übungen (Scrabble)
- Arrays
- Invarianten
- Aufgaben
- Kahoot

NACHBESPRECHUNG DER ÜBUNGEN

Aufgabe 5: Scrabble

In dieser Aufgabe sollen Sie Scrabble-Steine legen, mittels ASCII-Art auf der Konsole. Vervollständigen Sie die Methode `drawNameSquare` in der Klasse `Scrabble`. Diese Methode nimmt einen Namen als String-Parameter und soll den Namen als in einem Quadrat angeordnete Scrabble-Steine auf der Konsole (`System.out`) ausgeben. Wenn z.B. der String `Alfred` übergeben wird, sollte folgendes Bild ausgegeben werden:

```
+---+---+---+---+---+---+
| A | L | F | R | E | D |
+---+---+---+---+---+---+
| L |               | E |
+---+               +---+
| F |               | R |
+---+               +---+
| R |               | F |
+---+               +---+
| E |               | L |
+---+---+---+---+---+---+
| D | E | R | F | L | A |
+---+---+---+---+---+---+
```

Ihr Code braucht nur Namen der Länge 3 oder länger unterstützen. Für einen Namen mit Länge 3, z.B. `Jim`, sollte die Ausgabe so aussehen:

```
+---+---+---+
| J | I | M |
+---+---+---+
| I |   | I |
+---+---+---+
| M | I | J |
+---+---+---+
```

Beachten Sie, dass Ihr Programm keinerlei andere Ausgabe als das Scrabble-Quadrat machen darf.

Aufgabe 5: Scrabble

In dieser Aufgabe sollen Sie Scrabble-Steine legen, mittels ASCII-Art auf der Konsole. Vervollständigen Sie die Methode `drawNameSquare` in der Klasse `Scrabble`. Diese Methode nimmt einen Namen als String-Parameter und soll den Namen als in einem Quadrat angeordnete Scrabble-Steine auf der Konsole (`System.out`) ausgeben. Wenn z.B. der String `Alfred` übergeben wird, sollte folgendes Bild ausgegeben werden:

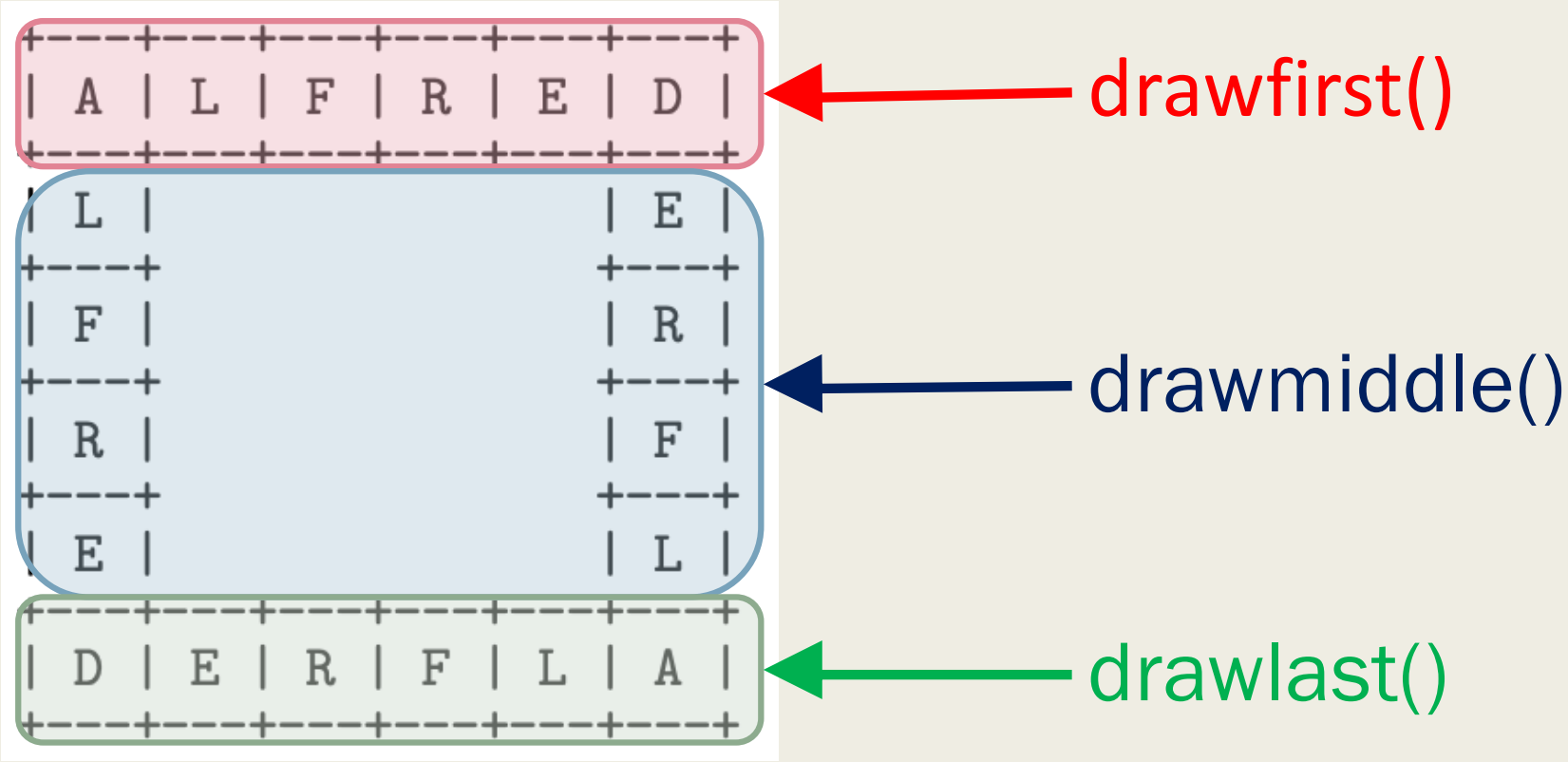
```
+---+---+---+---+---+---+
| A | L | F | R | E | D |
+---+---+---+---+---+---+
| L |                   | E |
+---+                   +---+
| F |                   | R |
+---+                   +---+
| R |                   | F |
+---+                   +---+
| E |                   | L |
+---+---+---+---+---+---+
| D | E | R | F | L | A |
+---+---+---+---+---+---+
```

Ihr Code braucht nur Namen der Länge 3 oder länger unterstützen. Für einen Namen mit Länge 3, z.B. `Jim`, sollte die Ausgabe so aussehen:

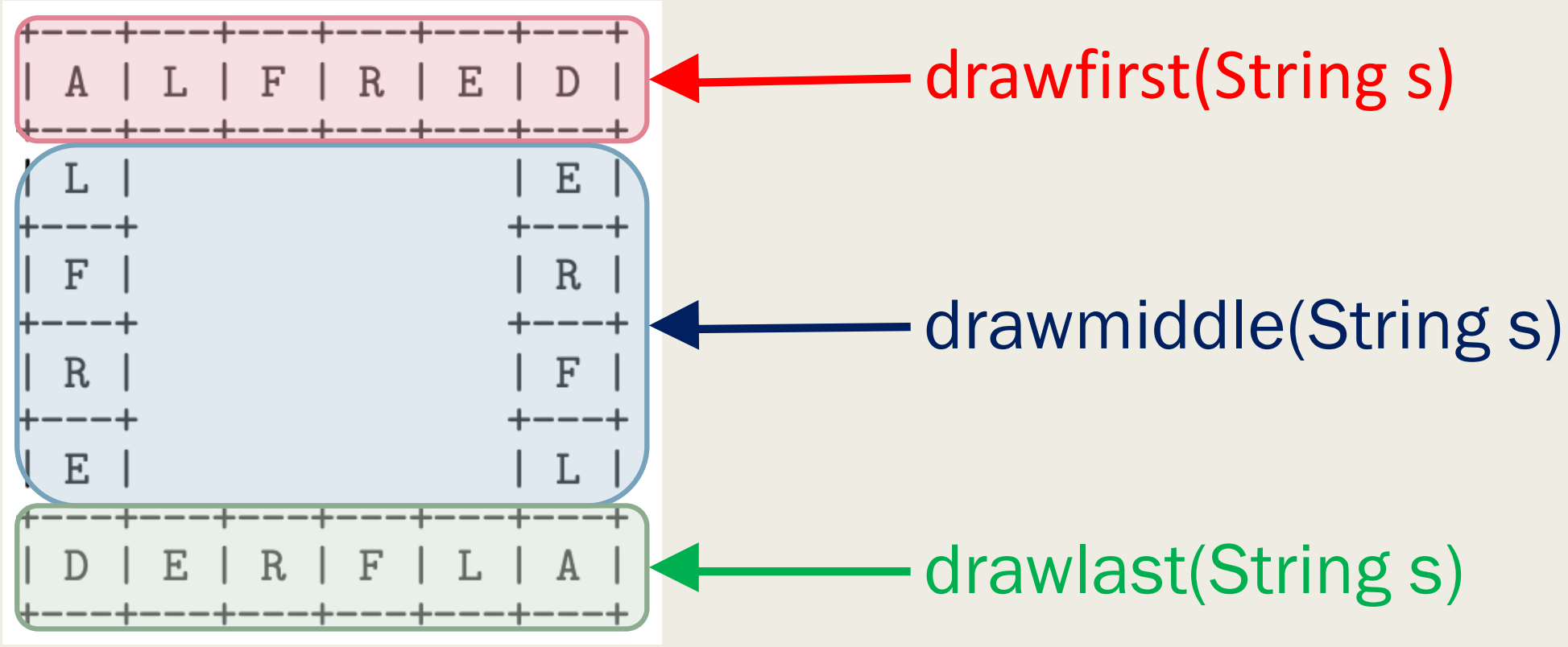
```
+---+---+---+
| J | I | M |
+---+---+---+
| I |   | I |
+---+---+---+
| M | I | J |
+---+---+---+
```

Beachten Sie, dass Ihr Programm keinerlei andere Ausgabe als das Scrabble-Quadrat machen darf.

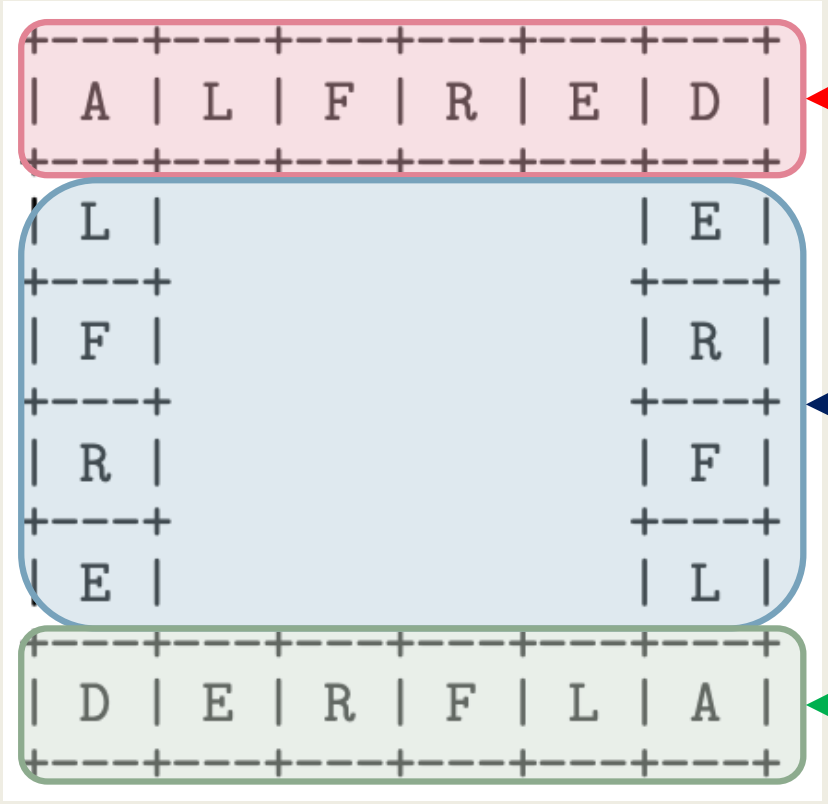
Beispiel Analysieren



Beispiel Analysieren



Beispiel Analysieren

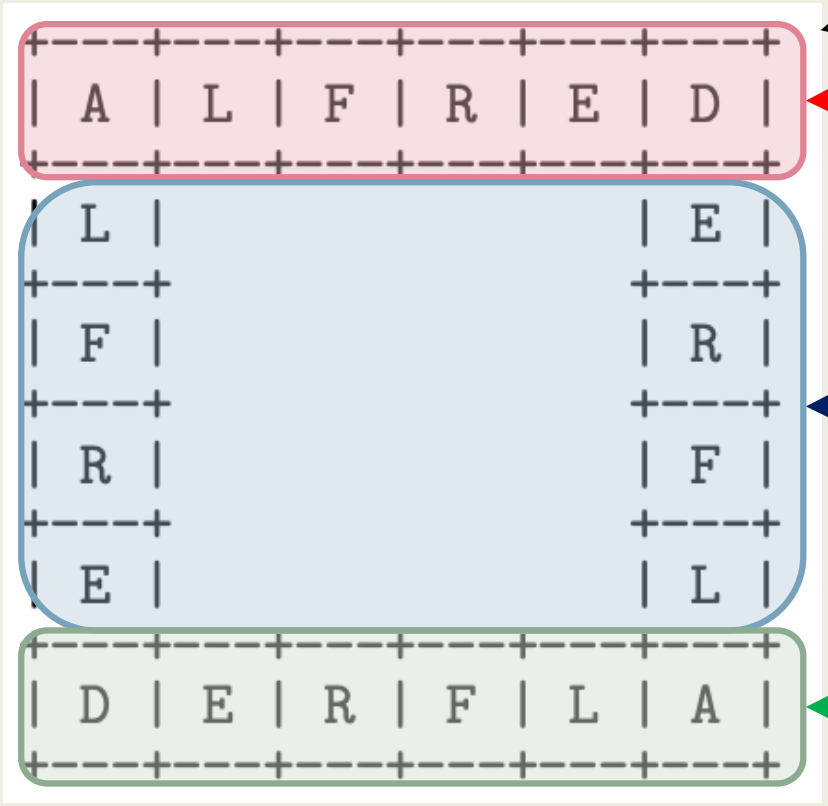


drawfirst(String s)

drawmiddle(String s)

drawlast(String s){
    drawfirst("reversed s")
}

Beispiel Analysieren



Grossbuchstaben

drawfirst(String s)

drawmiddle(String s)

drawlast(String s){
    drawfirst(reversed s)
}

```
public static void drawNameSquare(String name) {
    name = name.toUpperCase();
    drawfirst(name);
    drawmiddle(name);
    drawlast(name);
}
```

```
private static void drawfirst(String name) {

    System.out.println("+---".repeat(name.length())+"+");
    for (int i = 0;i<name.length();i++) {
        System.out.print("| "+name.charAt(i)+" ");
    }
    System.out.println("|");
    System.out.println("+---".repeat(name.length())+"+");
}
```

```
private static void drawmiddle(String name) {
    for (int i = 1;i<name.length()-1;i++) {
        System.out.print("| "+name.charAt(i)+" |");
        System.out.print(" ".repeat(-1+4*(name.length()-2)));
        System.out.print("| "+name.charAt(name.length()-i-1)+" |");
        System.out.println();
        if (i<name.length()-2)System.out.println("+---+"+" ".repeat(-
1+4*(name.length()-2))+"+---+");
    }

}
```

```
private static void drawlast(String name) {
    String reversed= "";
    for (int i =name.length()-1;i>=0;i--) {
        reversed+=name.charAt(i);
    }
    drawfirst(reversed);

}
```

THEORIE AN BEISPIEL

Videos

- Hoare Tripple (Gohar): https://www.youtube.com/watch?v=ooatiji7JNk
- Weakest Precondition (Cédric): https://www.youtube.com/watch?v=GnjuI0dN70I
- Pass by Value vs Pass by Reference (Inder): https://www.youtube.com/watch?v=ee3VKzZRpZc

Arrays – Arbeiten

```
public static void main(String[] args) {
    int[] ar = {1,5,3,1};
    int max = Integer.MIN_VALUE;

    for (int i =0;i<ar.length;i++){
        if (ar[i]>max){
            max = ar[i];
        }
    }
    System.out.println("Max: "+max);
}
```

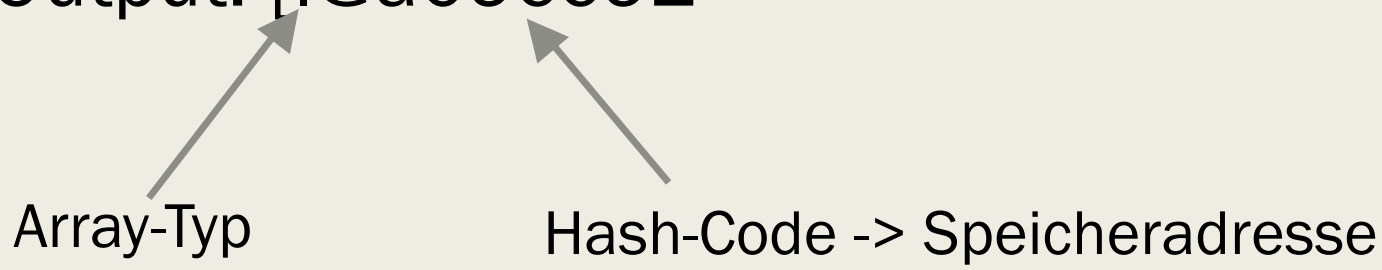
```
public static void main(String[] args) {
    int[] ar = {1,5,3,1};
    int max = Integer.MIN_VALUE;

    for (int i : ar){
        if(i>max){
            max=i;
        }
    }
    System.out.println("Max: "+max);
}
```

Arrays – toString

```
public static void main(String[] args) {
    int[] ar = {1,5,3,1};
    System.out.println(ar);
}
```

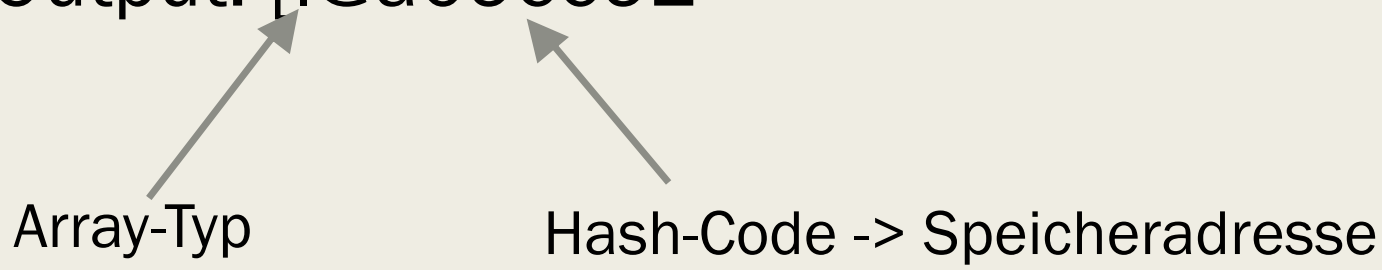
Output: [I@a09ee92



Arrays – toString

```
public static void main(String[] args) {
    int[] ar = {1,5,3,1};
    System.out.println(ar);
}
```

Output: [I@a09ee92



Arrays – toString

```
import java.util.Arrays;

public class EProgWSix {
    public static void main(String[] args) {
        int[] ar = {1,5,3,1};
        System.out.println(Arrays.toString(ar));
    }

}
```

Arrays – toString

```
import java.util.Arrays;

public class EProgWSix {
    public static void main(String[] args) {
        double[] ar = {1,5,3,1};
        System.out.println(Arrays.toString(null));
    }

}
```

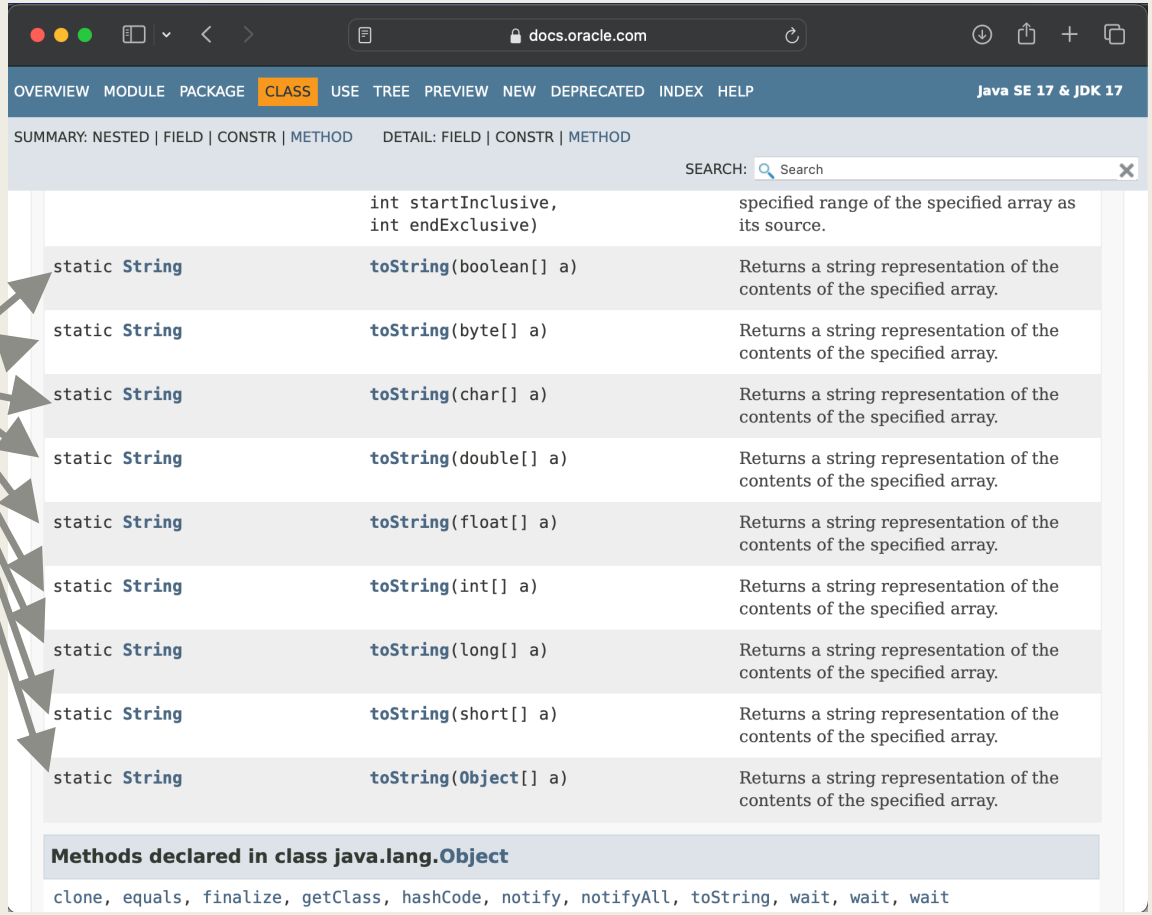
java: reference to toString is ambiguous
both method toString(double[]) in java.util.Arrays and method toString(java.lang.Object[]) in java.util.Arrays match

Arrays – toString

```
import java.util.Arrays;

public class EProgWSix {
   public static void main(String[] args) {
      double[] ar = {1,5,3,1};
      System.out.println(Arrays.toString(null));
   }

}
```



java: reference to toString is ambiguous
  both method toString(double[]) in java.util.Arrays and method toString(java.lang.Object[]) in java.util.Arrays match

Arrays – toString

```
import java.util.Arrays;

public class EProgWSix {
    public static void main(String[] args) {
        double[] ar = {1,5,3,1};
        ar = null;
        System.out.println(Arrays.toString(ar));
    }

}
```

Output: *null*

Arrays – toString

```
import java.util.Arrays;

public class EProgWSix {
    public static void main(String[] args) {
        double[] ar = {1,5,3,1};
        ar = null;
        System.out.println(Arrays.toString(ar));
    }

}
```

```
public static String toString(double[] a) {
    if (a == null)
        return "null";
    int iMax = a.length - 1;
    if (iMax == -1)
        return "[]";

    StringBuilder b = new StringBuilder();
    b.append('[');
    for (int i = 0; ; i++) {
        b.append(a[i]);
        if (i == iMax)
            return b.append(']').toString();
        b.append(", ");
    }
}
```

Output: *null*

Arrays – 2D

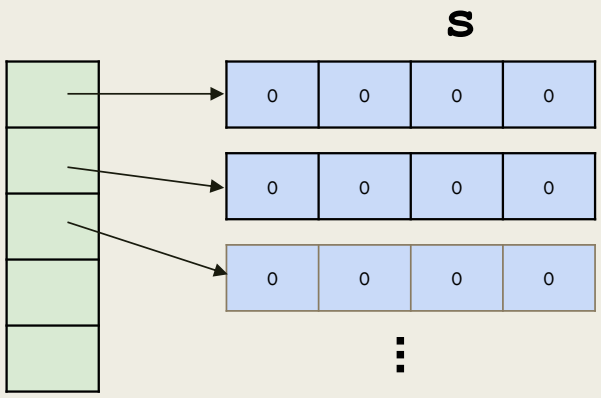
Java hat auch 2-dimensionale Arrays. Ein 2D-Array mit Elementen des Typs `type` wird so deklariert:

```
type [] [] array2d;
```

Der Array muss natürlich mit **`new`** erschaffen werden, z.B.:

```
int [] [] x = new int [zeilen] [spalten];
```

Ein 2D-Array ist ein lineares Array von Zeilen. Der **`new`** Operator initialisiert die Elemente der Zeilen mit dem Nullwert des Typs, z.B. 0 für **`int`**.



`array2d[`**`z`**`][`**`s`**`]` selektiert Zeile **`z`** und darin das **`s`**-te Element (die **`s`**-te Spalte)

Invarianten – Rezept (eprog.ch)

1. Loop Termination Condition

$$\{x \geq 0\}$$

2. Combine Postcondition and Loop Body

$$\{res = a + b\} \rightarrow \{res = a + b - x\}$$

3. Add any conditions (due to used methods)

$$\{x \geq 0 \;\&\&\; res = a + b - x\}$$

```
public int compute(int a, int b) {
   // Precondition:  a >= 0
   int x;
   int res;

   x = a;
   res = b;

   while (x > 0) {
      x = x - 1;
      res = res + 1;

   }

   // Postcondition:  res = a + b
   return res;
```

Invarianten – Rezept (eprog.ch)

1. Loop Termination Condition

$$\{i <= n\}$$

2. Combine Postcondition and Loop Body

$$\{result = 5^n\} \rightarrow \{result = 5^i\}$$

3. Add any conditions (due to used methods)

$$\{k == 5\}$$

$$\{i \leq n \&\& result = 5^i \&\& k == 5\}$$

```
public int compute(int n,int k){
   //Precondtion: n>=0
   int i =0;
   int result = 1;

   while (i<=n) {
      result *= k;
      i++;
   }
   //Postcondition result =
     Math.pow(5,n);
   return result;
}
```

Invarianten – Rezept (eprog.ch)

1. Loop Termination Condition

$$\{\boldsymbol{counter \leq n + 1}\}$$

2. Combine Postcondition and Loop Body

$$\{\boldsymbol{result = n!}\} \rightarrow \{\boldsymbol{res = (counter - 1)!}\}$$

3. Add any conditions (due to used methods)

$$\{counter \leq n + 1 \;\&\&\; result = (counter - 1)!\}$$

```
public static int factorial(int n) {
   // Precondition: n >= 0
   int counter;
   int result;
   counter = 1;
   result = 1;
   while (counter <= n) {
      result = result * counter;
      counter = counter + 1;
   }

   // Postcondition: result = n!

   return result;
}
```

Klassen und Objekte - Eclipse

AUFGABEN

Aufgaben

- Invarianten – Lösungen auf eprog.ch
- Prüfungsaufgabe - Auswertung
- Feedback

https://forms.gle/VauwMYKukLb1DuAN6

KAHOOT