

Lösungsvorschläge Übungen 3

A1

Lösungsvorschlag Aufgabe 1

```
/**
 * aufaufgabe01Summe() berechnet die Summe aller Arrayelemente
 *
 * @return      Summe aller Arrayelemente
 */
public int aufgabe01Summe()
{
    int summe = -1;
    for (int i=0; i<this.anzahl; i++) {
        summe = summe + daten[i];
    }
    return summe;
}
```

Lösungsvorschlag Aufgabe 2

```
/**
 * aufaufgabe02ZaehleNullen() Gibt die Zahl den Nullen im Array zurück
 *
 * @return      Zahl der Nullen
 */
public int aufgabe02ZaehleNullen()
{
    int numNull = 0;
    for (int i=0; i<this.anzahl; i++) {
        if ( daten[i] == 0 ) {
            numNull++;
        }
    }
    return numNull;
}
```

Lösungsvorschlag Aufgabe 3

```
/**
 * aufgabe03FindeLetzteNull() Gibt den Index des Elements mit der
 * letzten Null zurück
 *
 * @return      Index des Elements mit der letzten Null
 */
```

```
*/  
public int aufgabe03FindeLetzteNull()  
{  
    int letzteNullIndex = -1;  
    for(int i=0; i<this.anzahl; i++) {  
        if ( daten[i] == 0 ){  
            letzteNullIndex = i;  
        }  
    }  
    return letzteNullIndex;  
}
```

Lösungsvorschlag Aufgabe 4

```
/**  
 * aufgabe04FindeErsteNull() Gibt den Index des Elements mit der ersten  
Null zurück  
 *  
 * @return      Index des Elements mit der ersten Null  
 */  
public int aufgabe04FindeErsteNull()  
{  
    for(int i=0; i<this.anzahl; i++) {  
        if ( daten[i] == 0 ){  
            return i;  
        }  
    }  
    return -1;  
}
```

A5

Lösungsvorschlag Aufgabe 5

```
/**  
 * aufgabe05Enthaeilt1() Wahr, wenn die Zahlenreihe mindestens eine 1  
enthaelt  
 *  
 * @return      Wahr, wenn 1 vorhanden, sonst falsch  
 */  
public boolean aufgabe05Enthaeilt1()  
{  
    for(int i=0; i<this.anzahl; i++) {  
        if ( daten[i] == 1 ){  
            return true;  
        }  
    }  
}
```

```
    }  
    return false;  
}
```

A6

Lösungsvorschlag Aufgabe 6

```
/**  
 * aufgabe06Enthaeelt2Und5() a Wahr, wenn die Zahlenreihe mindestens eine  
2 * und eine 5 enthaelt  
 *  
 * @return Wahr, wenn 1 u. 5 vorhanden, sonst falsch  
 */  
public boolean aufgabe06Enthaeelt2Und5()  
{  
    boolean enthaelt2 = false;  
    boolean enthaelt5 = false;  
  
    for(int i=0; i<this.anzahl; i++) {  
        if ( daten[i] == 2 ) {  
            enthaelt2 = true;  
        }  
  
        if ( daten[i] == 5 ) {  
            enthaelt5 = true;  
        }  
    }  
  
    return ( enthaelt2 && enthaelt5 );  
}
```

A7

Lösungsvorschlag Aufgabe 7

```
/**  
 * aufgabe07EnthaeeltFixpunkt() Wahr, wenn die Zahlenreihe mindestens  
einen * Fixpunkt enthält  
 *  
 * @return Wahr, wenn Fixpunkt vorhanden, sonst falsch  
 */  
public boolean aufgabe07EnthaeeltFixpunkt()  
{
```

```
        for(int i=0; i<this.anzahl; i++) {  
            if ( daten[i] == i ){  
                return true;  
            }  
        }  
        return false;  
    }  
}
```

A8

Lösungsvorschlag Aufgabe 8

```
/**  
 * aufgabe08ZaehleWiederholungen() Gibt die Zahl dert Wiederholungen  
 zurück.  
 * Drei aufeinanderfolgende Zahlen sind zwei Wdh (wie kann man das  
 anders zaehlen,  
 * was muss man im Code aendern?)  
 *  
 * @return      Zahl der Wdh  
 */  
public int aufgabe08ZaehleWiederholungen()  
{  
    int numWdh = 0;  
    for(int i=0; i<this.anzahl-1; i++) {  
        if (daten[i] == daten[i+1]) {  
            numWdh++;  
        }  
    }  
    return numWdh;  
}
```

A9

Lösungsvorschlag Aufgabe 9

```
/**  
 * aufgabe09ZaehleDreierWiederholungen() Gibt die Zahl der dreier  
 Wiederholungen zurück.  
 * Vier aufeinanderfolgende Zahlen sind zwei 3er Wdh (wie kann man das  
 anders zaehlen,  
 * was muss man im Code aendern?)  
 *  
 * @return      Zahl der Wdh  
 */
```

```
public int aufgabe09ZaehleDreierWiederholungen()
{
    int numWdh = 0;
    for(int i=0; i<this.anzahl-2; i++) {
        if (daten[i] == daten[i+1] && daten [i+1] == daten[i+2]) {
            numWdh++;
        }
    }
    return numWdh;
}
```

From:
<https://info-bw.de/> -

Permanent link:
<https://info-bw.de/faecher:informatik:oberstufe:java:algorithmen:arrays:uebungen3:lsg:start?rev=1633590543>

Last update: **07.10.2021 07:09**

