Ein Programm zum Zahlenraten

Arbeite mit dem folgenden BlueJ Projekt: https://codeberg.org/qg-info-unterricht/bluej-binarysearch

Aufgaben:

A1

- Beschreibe die Funktion der privaten Methode `initZahlenreihe`. Antwort: `initZahlenreihe` erzeugt eine Liste mit Zahlen, die aufsteigend sortiert sind. Der Abstand zwischen zwei Elementen der Liste ist höchstens 10.
- Implementiere die Methode `printZahlenreihe`

Lösungsvorschlag

Mit foreach-Schleife:

```
public void printZahlenreihe() {
        System.out.println("Zahlenreihe:");
        for(int zahl: zahlenreihe) {
            System.out.print(zahl + " ");
        }
        System.out.println();
    }
```

Oder mit for-schleife:

```
public void printZahlenreihe() {
        System.out.println("Zahlenreihe:");
        for(int i=0; i<zahlenreihe.length; i++) {
            System.out.print(zahlenreihe[i] + " ");
        }
        System.out.println();
    }
```

A2

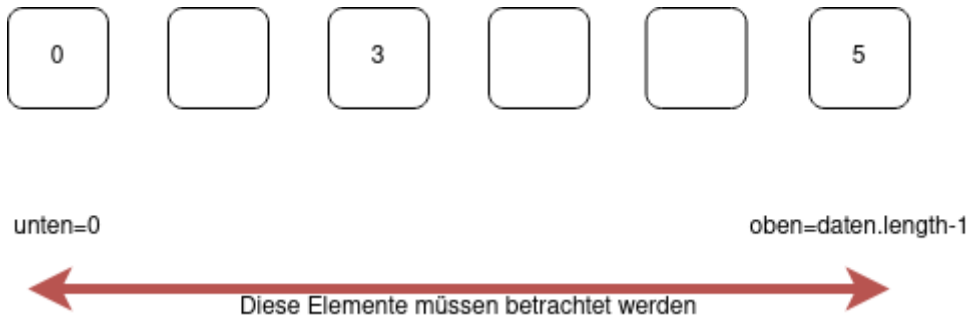
Implementiere eine Methode `binaereSuche`, welche den Index des gesuchten Elements zurückgibt oder `-1`, wenn der gesuchte Wert nicht gefunden wird. Folge dabei den Tipps und Aufgabenstellungen unten.

(1) Programmablaufplan

Erstelle ein Flussdiagramm anhand der folgenden Beschreibung.

```
  public int binaereSuche(int needle)
```

- Die Methode `binaereSuche` arbeitet auf dem zuvor erzeugten Array und nimmt als Argument die gesuchte Zahl entgegen. Der Rückgabewert ist der Index des Arrayelements, in dem der gesuchte Wert steht - oder -1, wenn dieser nicht gefunden wurde.
- Du führst Buch welcher Teil des Arrays zu durchsuchen ist und welcher Teil des Arrays nicht mehr in Frage kommt. wenn deine Methode startet, musst du das gesamte Array betrachten (kleinster Index `0`, größter Index `daten.length-1`)

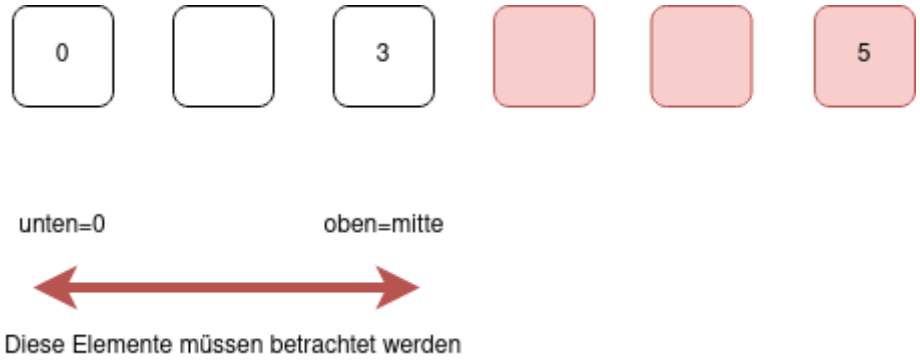


- Jetzt musst du den Index des mittleren Elements finden, und prüfen, ob der Inhalt größer, kleiner oder gleich dem gesuchten Wert ist.

```
  int mitte = (oben+unten)/2; //Wenn oben+unten ungerade ist, wird ''mitte''
abgerundet
```

- Ist der Wert des Arrayelements **gleich** dem gesuchten Wert, wird der Indexwert mit `return` zurückgegeben, denn das Element ist gefunden.
- Wenn der gesuchte Wert **kleiner** ist als der Inhalt von `daten[mitte]`, kann die obere Hälfte des Arrays ausgeschlossen werden, indem man als neue obere Grenze `mitte-1` festlegt.

```
if ( gesucht < daten[mitte]) {
   oben = mitte-1;
}
```



- Sollte der gesuchte Wert **größer** sein als `daten[mitte]` kann die untere Hälfte ausgeschlossen werden, dazu muss der Wert von `unten` auf `mitte+1` gesetzt werden.

```
if ( gesucht > daten[mitte]) {
```

```
    unten = mitte+1;
}
```

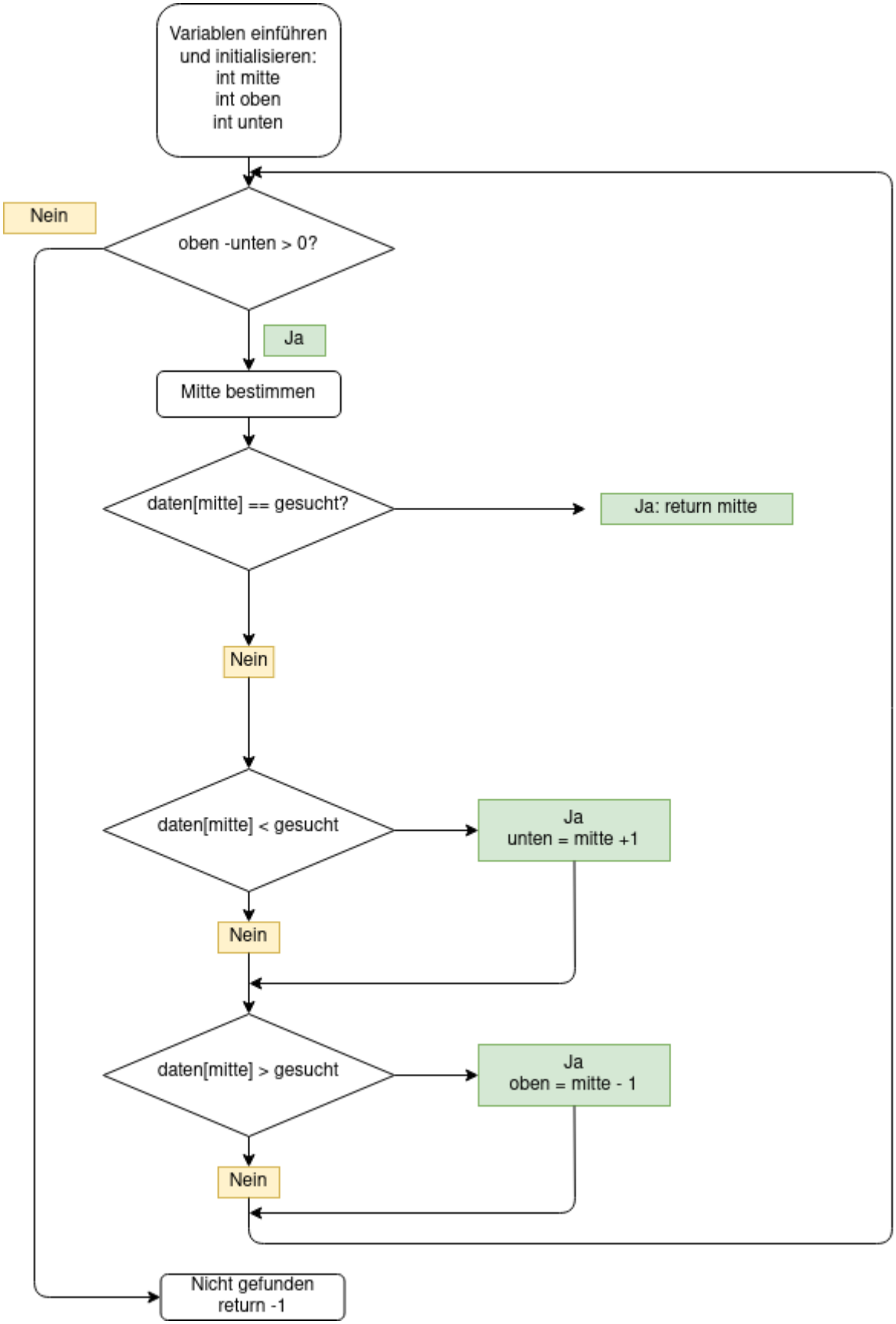
- Das ganze muss wiederholt werden, solange der Suchbereich `oben-unten` noch mindestens ein Element umfasst.

(2) Implementation

Implementiere die Methode wie entworfen und teste sie.

Hilfestellungen

Möglicher PAP



Mögliches Methodengerüst

```
public int binarySearch(int needle) {
        int minindex
```

```
        int maxindex
        int middleindex
        int middlevalue

        while (   ) {

            if ( ) {
                return middleindex;
            }


            if ( ) {
               maxindex =
            } else {
               minindex =
            }

            middleindex =
            middlevalue =

        }

        return -1;
    }
```

From:
https://www.info-bw.de/ -

Permanent link:
https://www.info-bw.de/faecher:informatik:oberstufe:algorithmen:binaere_suche:binsuchprogramm:start?rev=1747117481

Last update: **13.05.2025 06:24**