**Lineare Suche**

|  |
| --- |
| Mithilfe der **linearen Suche** kann eine unsortierte lineare Datenstruktur durchsucht werden. Die Suche beginnt beim ersten Element. Stimmt dieses mit dem Suchschlüssel überein, kann die Suche beendet werden. Andernfalls fährt man mit dem zweiten Element fort. Die Suche läuft so lange, bis das gesuchte Element gefunden oder das Ende der Datenstruktur erreicht wurde. Dann kann man sicher sein, dass das gesuchte Element nicht vorhanden ist. |

**Aufgabe 1: Idee des Algorithmus**

Vervollständigen Sie das Struktogramm zur linearen Suche.

Ein Bild, das Text, Screenshot, Reihe, Schrift enthält.

Automatisch generierte Beschreibung

???

???

???

???

???

???

**Aufgabe 2: Exemplarischer Durchlauf des Algorithmus**

Geben Sie für die gegebene Liste aus Zahlen für jeden Schleifendurchlauf die entsprechenden Werte der Variablen und logischen Ausdrücken an.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Liste: [7, 13, 2, 36, 203, 8] object: 203 | | | | | |
|  | gefunden | Aktuelles Listenelement | Inhalt | Inhalt == object | gefunden |
| 1 | false | 0 | 7 | false | false |
| 2 |  |  |  |  |  |
| 3 |  |  |  |  |  |
| 4 |  |  |  |  |  |
|  |  |  |  |  |  |
| Ausgabe: | | | | | |

**Aufgabe 3: Effizienzbetrachtung**

Geben Sie an, wie viele Vergleiche man im besten Fall, im Durchschnittsfall und im schlechtesten Fall durchgeführt werden müssen.

|  |  |  |
| --- | --- | --- |
|  | **Beschreibung des Falls** | **Zahl der Vergleiche** |
| **Bester Fall** |  |  |
| **Durchschnitt** |  |  |
| **Schlechtester Fall** |  |  |

**Aufgabe 4: Implementierung**

Implementieren Sie die Methode lineareSuche(T obj) für die Klasse List in Java.