

SelectionSort – Teil II

1. Begriffe

Beschreibe die beiden folgenden Begriffe im Hinblick auf Sortierverfahren:

out-of-place _____

in-place _____

2. *in-place*-SelectionSort

Der SelectionSort-Algorithmus kann auch in-place erfolgen. Beschreibe in freier Sprache, Pseudocode oder Ablaufdiagramm, welche Schritte hierbei durchgeführt werden müssen und was dabei zu beachten ist.

3. Programmierung

Erstelle im Paket `selectionsort` eine neue Klasse `InPlace` mit `main`-Methode.

Programmiere darin den in-place-SelectionSort-Algorithmus anhand dem in Aufgabe 2 erarbeiteten Ablauf.

4. Zusatzaufgabe: Laufzeitmessung

Mit der Methode `System.currentTimeMillis()` kann man sich die Millisekunden seit dem 1.1.1970 zurückgeben lassen. Da diese Zahl sehr groß ist, reicht ein einfacher `int`-Wert nicht aus, in Java gibt es deshalb für große ganze Zahlen den Datentyp `long`. Die Methode gibt einen Wert von diesem Datentyp zurück.

Speichert man nun die Millisekunden *direkt vor* dem Sortiervorgang und zieht man diese vom Wert *direkt nach* dem Sortiervorgang ab, so erhält man die Laufzeit des Sortiervorgangs in Millisekunden.

Aufgabe: Erzeuge ein Array mit 25000, 50000, 100000, 200000 zufälligen Werten. Lasse diese dann mit deinem in-place-SelectionSort-Algorithmus sortieren und miss die dafür benötigten Zeiten. Miss für jede Arraygröße 5 Zeiten und vergleiche die Durchschnittswerte.

Tabelle 1: Messwerte zur Sortierung von 25000 Zahlen

Messung	1	2	3	4	5	Durchschnitt
Laufzeit						

Tabelle 2: Messwerte zur Sortierung von 50000 Zahlen

Messung	1	2	3	4	5	Durchschnitt
Laufzeit						

Tabelle 3: Messwerte zur Sortierung von 100000 Zahlen

Messung	1	2	3	4	5	Durchschnitt
Laufzeit						

Tabelle 4: Messwerte zur Sortierung von 200000 Zahlen

Messung	1	2	3	4	5	Durchschnitt
Laufzeit						

- Wie verändert sich die Laufzeit, wenn die Größe des Arrays verdoppelt, verdreifacht, ... wird?
- Wie lange würde es damit dauern, das Telefonbuch von Berlin mit ca. 3 Millionen Einträgen zu sortieren? (*grober Richtwert!*)