

Aufgabenblatt 7

Generische Sortiermethoden

Implementieren Sie die beiden folgenden Sortierverfahren als generische Methoden mit Typbeschränkung.

1. Hybrides QuickSort:

Das Sortierverfahren arbeitet wie QuickSort mit eliminierter Endrekursion. Sobald die Größe des Teilfelds jedoch kleiner als z.B. $N = 100$ wird, wird mit `insertionSort` sortiert. Es ist hilfreich, `insertionSort` so anzupassen, dass ein Teilfeld `a[li], ..., a[re]` sortiert werden kann:

```
void insertionSort(T[] a, int li, int re)
```

2. Hybrides QuickSort mit 3-Median-Strategie:

Erweitern Sie das Verfahren aus 1. durch die 3-Median-Strategie.

Testen

- Testen Sie Ihr Verfahren für ein kleines mit etwa 10 – 20 Zahlen. Vollziehen Sie die Aufrufstruktur mit Papier und Bleistift nach.
- Testen Sie die Verfahren für zufällig generierte Integer-Feldern. Mit

```
(int) (Math.random() * M)
```

 lassen sich gleichverteilte zufällige ganze Zahlen aus $[0, M)$ erzeugen.
- Sortieren Sie alle Wörter, die in der Textdatei aus Aufgabe 1 vorkommen (Roman *Der Prozess* von Franz Kafka).
- Erzeugen Sie zufällige rote, schwarze und gemischte Spielkarten (siehe Aufgabe 4) und sortieren Sie diese. Beachten Sie, dass Spielkarten das Interface `Comparable` implementieren müssen.

Laufzeitmessungen

Führen Sie Laufzeitmessungen für Felder mit zufällig erzeugten Spielkarten durch und vergleichen Sie die beiden implementierten Verfahren mit einer entsprechenden Sortiermethode aus `java.util.Arrays`. Füllen Sie folgende Tabelle (nur die weißen Zellen) aus:

Verfahren	100_000 zufällige Spielkarten	200_000 zufällige Spielkarten	100_000 sortierte Spielkarten	200_000 sortierte Spielkarten
Hybrides QuickSort				
Hybrides QuickSort mit 3-Median				
Arrays.sort				