Sortieren 2: Quick-Sort optimiert

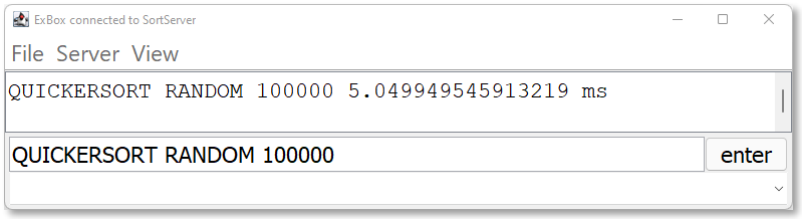
Aufgabe 1: Beschleunigung Quick-Sort

Quick-Sort zählt zu den schnellsten bekannten Sortieralgorithmen (ist aber nicht stabil). Dennoch lässt er sich noch beschleunigen, indem für kurze, zu sortierende (Array-)Intervalle ein anderes, einfacheres Sortierverfahren verwendet wird, wie z.B. Insertion-Sort, welches wir hier im Praktikum einsetzen möchten. Für diesen Zweck wird ein Schwellwert (threshold) definiert, ab dem auf das einfache Verfahren umgeschaltet wird. Wir bezeichnen diesen neuen Sortieralgorithmus - bescheiden wie wir sind - als Quicker-Sort.

Ergänzen Sie die Klasse SortServer um die zusätzliche Methode quickerSort. Nehmen Sie als Schwelle für die Umschaltung zunächst konstant 50.

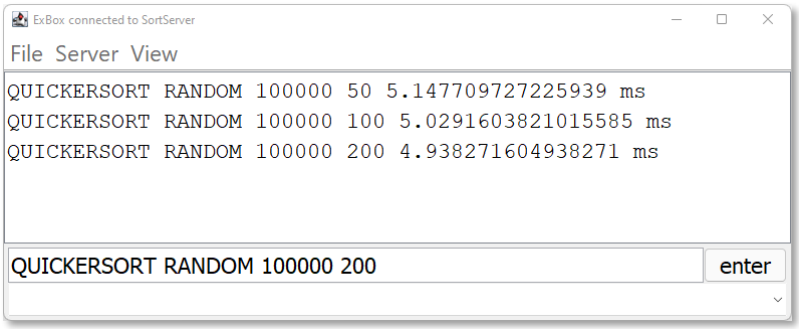
Hinweis:

- Bei der Messung wird wie im Praktikum 11 (Sortieren I) der Quicker-Sort eine bestimmte Zeit ausgeführt und der Durchschnitt berechnet.

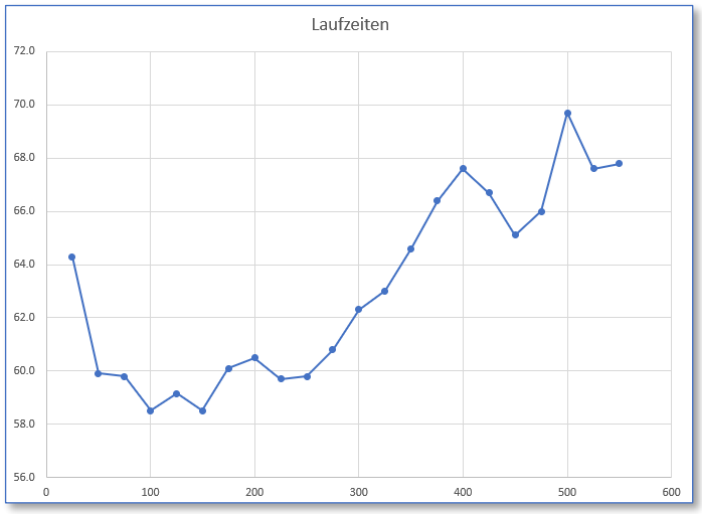


Aufgabe 2: Bestimmung der optimalen Schwelle

Verwenden Sie Ihren Quicker-Sort-Algorithmus und bestimmen Sie empirisch (d.h. durch Messen) einen guten Schwellwert. Übergeben Sie beim Aufruf des Sort-Servers für den Quicker-Sort zusätzlich den gewünschten Schwellwert.



Sie können die Werte in Excel auftragen und so das Minimum optisch bestimmen. Bei welchem Schwellwert hat Ihr Algorithmus das Optimum?



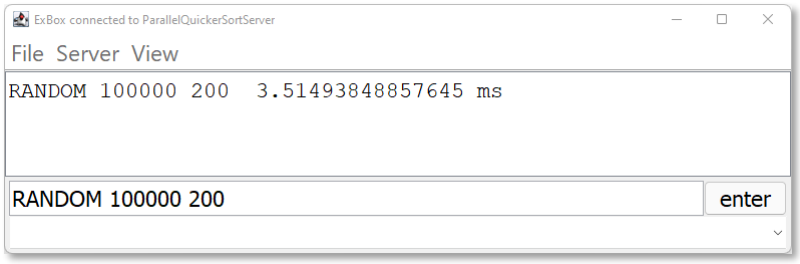
Hinweis:

- Vergleichen Sie die Zeiten Ihres Algorithmus mit der zur Verfügung gestellten Stream-Methode (Aufruf z.B. mittels «STREAM RANDOM 10000».

Die Aufgabe muss nicht abgegeben werden.

Aufgabe 3: Paralleler Quicker-Sort

Auf einem Rechner mit einem Mehrkernprozessor kann Quicker-Sort noch beschleunigt werden, indem die Cores besser ausgelastet werden. Implementieren Sie basierend auf Ihrem QuickerSort-Algorithmus einen solchen parallelen Sortieralgorithmus in der Klasse ParallelQuickerSortServer.



Hinweis:

- Verwenden Sie wie in der Vorlesung gezeigt Threads zur Lösung (Beispiel-Klasse NaiveParallelQuicksort2 auf den Folien).
- Da Threads aufwändig sind, macht es Sinn, diese nur ab einer bestimmten Array-Grösse zu verwenden. Die Konstante SPLIT_THRESHOLD legt diese fest. Darunter wird der normale Quicker-Sort verwendet.
- Falls bei der Abgabe auf Moodle einen `OutOfMemory-Fehler` erhalten → erhöhen Sie `SPLIT_THRESHOLD` z.B. auf 50'000.
- Ist die Lösung schneller?