

Übung 7

Sortieralgorithmen

Lea Happel & Luise Zieger

In dieser und der nächsten Übung wollen wir die Laufzeitkomplexität verschiedener Sortieralgorithmen, die in der Vorlesung besprochen werden, untersuchen. Es sei n die Anzahl der Elemente, welche wir (hier) in aufsteigender Reihenfolge sortieren wollen.

Bubblesort

Beim Bubblesort-Verfahren werden wiederholt von vorn beginnend je zwei benachbarte Elemente miteinander verglichen. Ist ein Element größer als sein Nachfolger, so tauschen beide ihren Platz. Nach dem ersten Durchlauf steht somit das größte Element an letzter Stelle, sodass im zweiten Schleifendurchlauf nur noch $n - 1$ Elemente verglichen werden. Usw.

Hinweis: Ist in einer Iteration gar kein Tausch nötig, so stehen alle Elemente in der richtigen Reihenfolge und die Schleife kann vorzeitig verlassen werden.

Selectionsort

Beim Selectionsort wird in jeder Iteration das kleinste Element unter allen noch nicht einsortierten Elementen gesucht. In der ersten Iteration werden somit alle Elemente miteinander verglichen und anschließend das Minimum mit dem Element an der ersten Stelle getauscht. Im zweiten Schleifendurchlauf wird unter den $n - 1$ verbleibenden Elementen das Minimum ausfindig gemacht und sodann an die zweite Stelle gesetzt. Usw.

Aufgabe

- Schreibe je ein Unterprogramm für die zwei beschriebenen Sortieralgorithmen. Als Parameter wird ein eindimensionales Feld übergeben, welches sortiert werden soll.
- Füge beiden Funktionen eine Zählvariable hinzu, welche die Anzahl der zum Sortieren benötigten Schritte mitzählt.
- Schreibe ein Hauptprogramm, in welchem eindimensionale INTEGER-Felder generiert werden, um die Funktionen zu testen. Wie viele Schritte werden benötigt, um ein Feld zu sortieren, dessen Elemente bereits in der richtigen Reihenfolge sortiert sind/ in umgekehrter Reihenfolge sortiert sind? Wie sieht es bei zufällig erzeugten Feldelementen aus?
- ZUSATZ: Fällt dir eine Möglichkeit ein, wie der Bubblesort-Algorithmus noch weiter „optimiert“ werden kann?