

Aufgabe 1

Ein anderer Blick auf die Sortierverfahren... Ein Sortieralgorithmus heißt *stabil*, wenn er die relative Ordnung "gleich großer" Schlüssel nicht verändert. Eine Anwendung: Wir sortieren die Einträge in einem Telefonbuch zuerst nach den Vornamen und danach nach den Nachnamen. Haben wir im zweiten Schritt ein stabiles Sortierverfahren verwendet, sind alle Einträge mit Anschlüssen auf den gleichen Nachnamen intern nach den Vornamen sortiert:

- Klein, Georg
- Klein, Harald
- Meier, Ansgar
- Meier, Bert
- Meier, Hans

...
Untersuchen Sie für die fünf in der Vorlesung behandelten Sortierprobleme, ob diese stabil sind.

Selection Sort:

Meier, Ansgar		Klein, Georg		Klein, Georg	
Klein, Georg		Meier, Ansgar		Klein, Harald	
Meier, Bert	->	Meier, Bert	->	Meier, Bert	->
Klein, Harald		Klein, Harald		Meier, Ansgar	instabil, Ansgar
Meier, Hans		Meier, Hans		Meier, Hans	"wandert"

Insertion Sort:

Meier, Ansgar		Klein, Georg		Klein, Georg	
Klein, Georg		Meier, Ansgar		Klein, Harald	
Meier, Bert	->	Meier, Bert	->	Meier, Ansgar	->
Klein, Harald		Klein, Harald		Meier, Bert	stabil
Meier, Hans		Meier, Hans		Meier, Hans	

Bubble Sort:

Meier, Ansgar		Klein, Georg		Klein, Georg		Klein, Georg
Klein, Georg		Meier, Ansgar		Meier, Ansgar		Klein, Harald
Meier, Bert	->	Meier, Bert	->	Klein, Harald	->	Meier, Ansgar
Klein, Harald		Klein, Harald		Meier, Bert		Meier, Bert
Meier, Hans		Meier, Hans		Meier, Hans		Meier, Hans

-> stabil

Merge Sort:

Meier, Ansgar		Klein, Georg		Klein, Georg	
Klein, Georg		Meier, Ansgar		Klein, Harald	
Meier, Bert	->	Klein, Harald	->	Meier, Ansgar	->
Klein, Harald		Meier, Bert		Meier, Bert	stabil
Meier, Hans		Meier, Hans		Meier, Hans	

Quick Sort:

Meier, Ansgar		Klein, Georg			
Klein, Georg		Klein, Harald			
Meier, Bert	->	Meier, Bert	->	instabil, Ansgar	"wandert"
Klein, Harald		Meier, Ansgar			
Meier, Hans		Meier, Hans			

Aufgabe 2

Sortieren Sie die Folge HANNOVER mit

• Insertion Sort

1. Schritt: HANNOVER A an Stelle 1 schieben, da H (an Stelle 1) $> A$
2. Schritt: AHNNOVER N überprüfen: Keine Änderungen, da A und $H < N$
3. Schritt: AHNNOVER N überprüfen: Keine Änderungen, da A , H und anderes $N \leq N$
4. Schritt: AHNNOVER O überprüfen: Keine Änderungen, da A , H und die beiden $N < O$
5. Schritt: AHNNOVER V überprüfen: Keine Änderungen, da A , H , N und $O < V$
6. Schritt: AHNNOVER E an 2. Stelle schieben, da $A < E$, aber $H > E$
7. Schritt: AEHNNOVR R an vorletzte Stelle setzen, da $O < R$, aber $V > R$
8. Schritt: AEHNNORV fertig!

• Bubble Sort und

1. Schritt: AHNNOVER Die ersten beiden Elemente werden überprüft: Keine Änderungen
2. Schritt: AHNNOVER Das zweite wird mit dem dritten Element verglichen: Keine Änderungen
3. Schritt: AHNNOVER Das dritte mit dem vierten: Keine Änderungen
4. Schritt: AHNNOVER Das vierte mit dem fünften: Keine Änderungen
5. Schritt: AHNNOVER Das fünfte mit dem sechsten: Keine Änderungen
6. Schritt: AHNNOEVR Das sechste wird mit dem siebten verglichen: Das E tauscht mit dem V den Platz
7. Schritt: AHNNOERV Das V tauscht hier mit dem R den Platz, weil es größer ist.
8. Schritt: AHNNOERV Wieder die ersten beiden Elemente: Keine Änderungen
9. Schritt: AHNNOERV auch hier keine Änderungen
10. Schritt: AHNNOERV keine Änderungen
11. Schritt: AHNNOERV *gähnt*
12. Schritt: AHNNEORV E und O tauschen ihre Plätze, weil E kleiner ist als O
13. Schritt: AHNNEORV keine Änderungen
14. Schritt: AHNNEORV das V wird den Platz nicht mehr verlassen, da es als das größte Element gefunden und platziert wurde
15. Schritt: AHNNEORV das hatten wir schon zweimal
16. Schritt: AHNNEORV keine Änderungen
17. Schritt: AHNNEORV keine Änderungen
18. Schritt: AHNENORV das E tauscht mit dem N den Platz, weil es kleiner ist als das N
19. Schritt: AHNENORV keine Änderungen
20. Schritt: AHNENORV hier auch nicht
21. Schritt: AHNENORV keine Änderungen
22. Schritt: AHNENORV wieder von vorne
23. Schritt: AHNENORV keine Änderungen
24. Schritt: AHENNORV das E kommt wieder eine Position nach vorne
25. Schritt: AHENNORV keine Änderungen
26. Schritt: AHENNORV
27. Schritt: AHENNORV
28. Schritt: AHENNORV
29. Schritt: AHENNORV neue Runde
30. Schritt: AEHNNORV das E tauscht mit dem H, es werden auch die übrigen Elemente nochmal miteinander verglichen. und so weiter...

• Merge Sort

1. Schritt: AHNNOVER Paarweises Tauschen zwischen a und H, N und N, O und V, E und R (nur A und H wechseln)
2. Schritt: AHNNEORV jetzt in Viererblocks: AH und NN werden sortiert vereint und OV mit ER.
3. Schritt: AEHNNORV jetzt die Viererblocks zusammen: Es ergibt sich die Lösung

- Merge Sort

Geben Sie alle zum Verständnis wichtigen Zwischenschritte an!

```
HANNOVER
HANN  OVER
HA NN  OV  ER
H A  N N  O V  E R
A H  N N  O V  E R
AH NN  OV  ER
AHNN  EORV
AEHNNORV
```

Die Folge wird in zwei Hälften aufgeteilt, diese werden wiederum aufgeteilt. Das geht so lange, bis die Folge in ihre Einzelteile aufgeteilt wurde. Jetzt wird jedes Element mit seinem Nachbarn verglichen und zusammengeführt. Das passiert so lange, bis aus den Einzelteilen wieder ein Ganzes geworden ist.

Aufgabe 3

Wir arbeiten die Klasse Sort um. Sie haben bereits Selection Sort implementiert: fortan soll dies durch eine Klassenmethode namens selectionSort geschehen, der ein Feld von Objekten, die Comparable implementieren, übergeben wird.

Schreiben Sie folgende Klassenmethoden: comparisons liefert die Anzahl der Vergleiche compareTo der zuletzt ausgeführten Sortiermethode. time liefert die Zeit, die die zuletzt ausgeführte Sortiermethode gebraucht hat. Verwenden Sie dazu System.currentTimeMillis()!

Aufgabe 4

Erweitern Sie die Klasse aus Aufgabe 3 um Insertion Sort, Bubble Sort und Merge Sort.