2.8 Üben und Vertiefen

Aufgabe 1

Es sind individuelle Lösungen möglich.

Aufgabe 2

Die Verwendung einer Liste ist immer dann geeignet, wenn in einem Programm mehrere (gleichartige) Elemente gespeichert werden sollen, die in derselben Art und Weise behandelt werden sollen.

Beispiele: Einkaufsliste, Klassenliste, Warteschlange, …

Aufgabe 3

a) siehe Snap!-Datei `2-8_Aufgabe3_TicTacToe_Loesung.xml`

b) siehe Snap!-Datei `2-8_Aufgabe3_TicTacToe_Loesung.xml`

c) Einer der Spieler hat gewonnen, wenn in der Liste an den entsprechenden Indizes das gleiche Symbol hinterlegt ist. Dabei gibt es folgende 8 Möglichkeiten:

- 1, 2, 3 (erste Reihe)
- 4, 5, 6 (zweite Reihe)
- 7, 8, 9 (dritte Reihe)
- 1, 4, 7 (erste Spalte)
- 2, 5, 8 (zweite Spalte)
- 3, 6, 9 (dritte Spalte)
- 1, 5, 9 (Diagonale links oben nach rechts unten)
- 3, 5, 7 (Diagonale links unten nach rechts oben)

d) siehe Snap!-Datei `2-8_Aufgabe3_TicTacToe_Loesung.xml`

e) siehe Snap!-Datei `2-8_Aufgabe3_TicTacToe_Loesung.xml`

f) siehe Snap!-Datei `2-8_Aufgabe3_TicTacToe_Loesung.xml`

Aufgabe 4

a) Die Aussage ist falsch. Da eine leere Liste keine Elemente beinhaltet, hat sie Länge 0.

b) Die Aussage ist falsch. Ein Element kann auch an einer konkreten Position innerhalb der Liste eingefügt werden.

c) Die Aussage ist falsch. Das zuvor dort gespeicherte Element, sowie alle nachfolgenden Elemente rutschen um eine Position nach hinten.

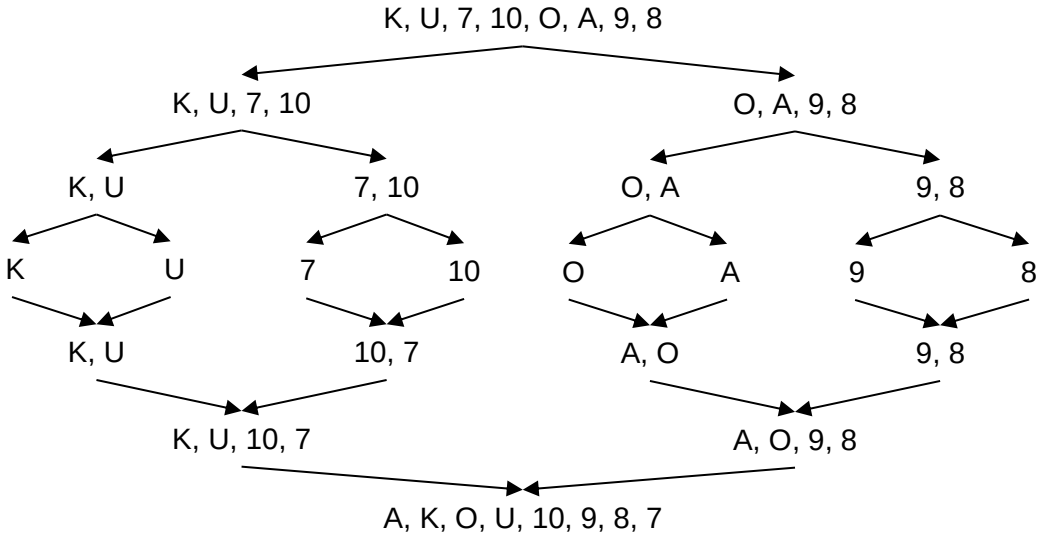
d) Die Aussage ist korrekt. Da die Länge einer Liste immer dem höchsten Index entspricht, sind keine leeren Plätze innerhalb der Liste möglich.

© C.C.Buchner Verlag, Bamberg

Aufgabe 5

a) Unter Verwendung des Mergesort-Algorithmus wird die zu sortierende Liste rekursiv in kleinere Hälften zerlegt, bis einzelne Elemente erreicht sind. Diese gelten als bereits sortiert. Anschließend werden die sortierten Hälften durch wiederholtes Zusammenführen in sortierter Reihenfolge verbunden. Meist wird dazu ein Hilfsarray verwendet. Kurz gesagt verwendet der Mergesort-Algorithmus zwei Grundprinzipien: Schrittweise Aufteilung der Liste und geordnetes Zusammenführen dieser.

b) Ursprungszustand:



Aufgabe 6

Im Folgenden wird von der „verbesserten“ Variante des Bubblesort-Algorithmus (siehe Lösung zu S.39/A2b). Geht man nicht von dieser „verbesserten“ Variante aus, so machen die Begriffe Worst-/Best-Case keinen Sinn, da immer gleichviele Listendurchläufe nötig sind.

a) Der Worst-Case beim Bubblesort-Algorithmus tritt ein, wenn die Liste genau falsch herum sortiert ist. Ist dies der Fall, so wird pro Durchlauf der Liste genau 1 Zahl richtig positioniert – die Reihenfolge der anderen Zahlen ändert sich nicht.

b) Der Best-Case beim Bubblesort-Algorithmus tritt ein, wenn die Liste bereits korrekt sortiert ist. In diesem Fall ist nur genau ein Listendurchlauf nötig.

c) Im Worst-Case benötigt der Bubblesort-Algorithmus bei n Listenelementen n^2 Vergleiche. Im Best-Case werden nur n Vergleiche benötigt.

d) Beim Mergesort-Algorithmus gibt es keinen Unterschied zwischen Best- und Worst-Case. Die Ursprungsliste wird immer komplett in einelementige Teillisten aufgeteilt und dann sortiert wieder zusammengefügt.

© C.C.Buchner Verlag, Bamberg

Aufgabe 7

a)

	B	C	D	E	F
A	7;A	9;A	∞	∞	14;A
A,B		9;A	22;B	∞	14;A
A,B,C			20;C	∞	11;C
A,B,C,F			20;C	20;F	
A,B,C,F,D				20;F	
A,B,C,F,D,E					

Nimmt man den kürzesten Weg von A nach E, so beträgt die Fahrzeit 20 Minuten (A – C – F – E).

b)

	B	C	D	E	F
A	7;A	9;A	∞	∞	14;A
A,B		9;A	22;B	∞	14;A
A,B,C			20;C	∞	14;A
A,B,C,F			20;C	23;F	
A,B,C,F,D				23;F	
A,B,C,F,D,E					

Durch die Bausperren ändert sich die Fahrzeit des kürzesten Weges von A nach E zu 23 Minuten. Außerdem muss ein anderer Weg genommen werden: A – F – E

© C.C.Buchner Verlag, Bamberg