Algorithmische Mathematik I

Wintersemester 2009/2010
Prof. Dr. Mario Bebendorf
Dr. Jan Hamaekers

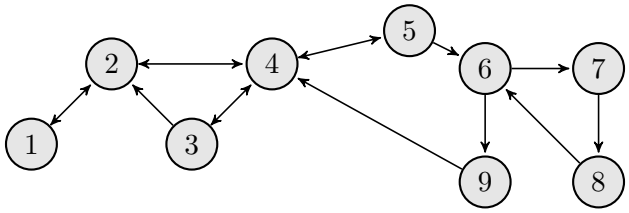
universitätbonn

Übungsblatt 6.

Abgabe am **Mittwoch, 2.12.2009 (vor der Vorlesung)**.

Aufgabe 1. (Graphen)

Wir betrachten den folgenden gerichteten Graphen, in dem $\leftrightarrow$ für eine Doppelkante steht (d.h. Kurzform für je eine Kante *vom* Knoten und eine Kante *zum* Knoten).



a) Geben Sie alle einfachen Pfade von Knoten 1 zu Knoten 8 sowie von Knoten 8 zu Knoten 1 an.

b) Geben Sie die Adjazenzmatrix für obigen Graphen an. Geben Sie auch die Adjazenzmatrix an, wenn alle Kanten *ungerichtet* wären.

c) Gehen Sie nun davon aus, dass alle Kanten ungerichtet sind. Geben Sie die maximale Anzahl Kanten an, die entfernt werden können, so dass der Graph zusammenhängend bleibt.

d) Zeichnen Sie einen Graph mit Knoten $1, 2, \ldots, 9$, der die folgende Adjazenzmatrix besitzt.

	1	2	3	4	5	6	7	8	9
1		1			1				
2	1		1				1		
3		1							
4					1				
5	1			1					
6			1				1		
7		1				1		1	1
8							1		
9				1			1		

Muss der Graph gerichtet gezeichnet werden oder kann er auch ungerichtet sein?

(10 Punkte)

Aufgabe 2. (Darstellung vollständiger Graphen)

Ein ungerichteter Graph $G = (V, E)$ heißt vollständig, wenn je zwei Knoten aus V miteinander durch eine Kante verbunden sind. Für $n = |V|$ bezeichnen wir den zugehörigen vollständigen Graphen mit $\mathcal{K}_n$.

Zeigen Sie: Für alle $n = 1, 2, 3, \ldots$ kann man jeden Graphen $\mathcal{K}_{2n+1}$ in einem Zug zeichnen, d.h. ohne zwischendurch abzusetzen und ohne mehrmals über eine Kante zu gehen.
Hinweis: Überlegen Sie sich zunächst, wie Sie einen $\mathcal{K}_{2n+1}$ zeichnen könnten, wenn Sie dies bereits für einen $\mathcal{K}_{2n-1}$ wissen. Beweisen Sie dann die Aussage mittels vollständiger Induktion.

(10 Punkte)

Aufgabe 3. (Heap-Sort)

Ein Element kann aus einem Heap $z_1, \ldots, z_n$ dadurch entfernt werden, dass die entstehende Lücke jeweils durch das kleinere der beiden Elemente z_{2i} bzw. z_{2i+1} ersetzt wird, bis keine darunterliegenden Elemente existieren.
Zeigen Sie den Satz 4.27 aus der Vorlesung:
Falls $z_1, \ldots, z_n$ die Heapeigenschaft besitzt, so kann durch obige Vorgehensweise ein Element mit Aufwand $O(\log n)$ so entfernt werden, dass die resultierende Folge die Heapeigenschaft besitzt.

(10 Punkte)

Programmieraufgabe 3. (Heap-Sort)

Implementieren Sie den Heap-Sort Algorithmus aus der Vorlesung.
Wie auch für die letzte Programmieraufgabe sollen Arrays von *dynamischer*[1] Länge des Typs `signed int` akzeptiert werden.
Wenden Sie Ihre Implementierung auf die Zahlenfolge
503 087 512 061 908 170 897 275 653 426 154 509 612 677 765 703
an. (10 Punkte)

Achtung: Die Punkte für die Programmieraufgabe werden gesondert gezählt!
Die Programmieraufgaben werden im CIP-Pool (Wegelerstraße 6, Raum E02) abgegeben. Bezüglich des konkreten Termines der Abgabe der Programmieraufgabe hängen in der Woche vom 7.12.-11.12.2009 im CIP-Pool Listen aus. Siehe dazu auch die Vorlesungshomepage `http://wissrech.ins.uni-bonn.de/teaching/algmath/ws09_i`.

Abgabe innerhalb der Woche 14.12.–18.12.2009 im CIP-Pool

[1] Auf der Vorlesungswebseite finden Sie Hinweise zur dynamischen Speicherverwaltung.