RWTH Aachen **SS 2015**
Lehrgebiet Theoretische Informatik Blatt 6
Rossmanith–Reidl–Sánchez 20. Mai 2015

Übung zur Vorlesung Algorithmen und Datenstrukturen

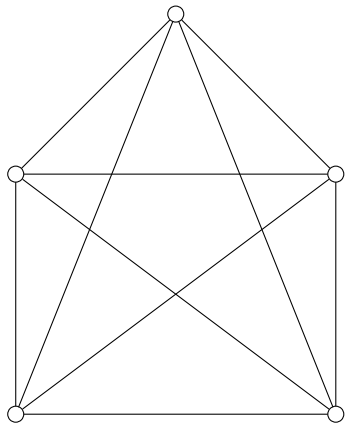
Aufgabe T14

	Quicksort	Heapsort	Mergesort	Insertion-Sort	Straight-Radix	Radix-Exchange
in-place?						
stabil?						
Laufzeit (worst-case)						
Laufzeit (Durchschnitt)						
vergleichsbasiert?						

Beantworten Sie die Fragen für alle Sortierverfahren. Gehen Sie davon aus, daß ein Vergleich in konstanter Zeit durchgeführt wird und die Anzahl der zu sortierenden Elemente n beträgt. Für Laufzeiten tragen Sie eine Funktion $f(n)$ in die Tabelle ein, um eine Laufzeit von $O(f(n))$ auszudrücken.

Aufgabe T15

Ein *Dreieck* in einem Graphen ist ein Untergraph, der aus drei Knoten besteht, welche paarweise miteinander verbunden sind.
Entwerfen Sie einen Algorithmus, der für einen Graphen mit n Knoten in $O(n^3)$ Schritten herausfinden kann, ob er ein Dreieck enthält.



Wieviele Dreiecke gibt es im oben gezeigten Graphen?

Aufgabe H11

Gegeben sei ein Array der Länge n, welches garantiert nur k verschiedene Zahlen enthält – jede aber beliebig oft. Wir gehen davon aus, daß n viel größer als k ist und k sogar viel kleiner als $\log n$ sein kann.
Erfinden und beschreiben Sie ein Sortierverfahren, welches in dieser speziellen Situation sehr schnell ist.
Genauer gesagt: Die Laufzeit soll nur $O(n \log k)$ betragen.

Aufgabe H12

Es gibt ein einfaches Verfahren, zwei quadratische $n \times n$-Matrizen miteinander zu multiplizieren. Die Laufzeit des einfachen Verfahrens ist $O(n^3)$.

Allerdings gibt es kompliziertere Verfahren, die asymptotisch deutlich schneller eine Matrixmultiplikation ausführen können.

In dieser Aufgabe wollen wir dies ausnutzen, um schneller nach Dreiecken in einem Graphen suchen zu können.

a) Stellen Sie sich vor, Sie multiplizieren eine Adjazenzmatrix eines Graphen G mit sich selbst und verwenden die resultierende Matrix wieder als Adjazenzmatrix eines neuen Graphens, den wir G' nennen. Welche interessante Beziehung besteht zwischen G und G'?

b) Entwerfen Sie jetzt einen auf schneller Matrixmultiplikation basierenden Algorithmus, der feststellen kann, ob ein Graph ein Dreieck enthält. Die Laufzeit soll dabei von der Laufzeit zur Matrixmultiplikation dominiert werden und alles andere soll nur $O(n^2)$ Zeit verbrauchen.