Name: .... **Matr.-Nr.:** ....

	Punkte	max
1		6
2		6
3		6
4		6
5		6
6		6
Σ		36

Probeklausur Algorithmen

(Kaufmann, Sommersemester 2013)

- Die Bearbeitungszeit beträgt 90 Minuten, elektronische Hilfmittel sind verboten, nur ein A4-Blatt mit Notizen ist erlaubt.
- Von den 6 Aufgaben mit jeweils 6 möglichen Punkten werden lediglich die 5 besten bewertet (gilt für die Klausur).

Viel Erfolg!

Aufgabe 1: **(Algorithmenentwurf)** (6 Punkte)

Ein ungerichteter Graph $G = (V, E)$ heisst bipartit, wenn V in A und B zerlegt werden können, so dass alle Kanten $e = (u, w) \in E$ gilt, dass $u \in A$ und $w \in B$ ist oder umgekehrt.

Gegeben sei $G = (V, E)$ ein ungerichteter Graph. Geben Sie einen Algorithmus in Pseudocode, der bestimmt, ob G bipartit ist.

Der Graph $G = (V, E)$ ist in Adjazenzlistendarstellung gegeben.

Hinweis: Kommentare und gute Variablennamen sind wichtig.

Aufgabe 2: **(Hashing)** (6 Punkte)

Ein Anagramm eines Strings s ist ein String t, so dass t eine Permutation von s ist. Sie wollen nun in eine Liste von Strings Anagramme finden.

(a) Skizieren Sie, wie Sie mit Hilfe einer geeigneten Hashfunktion Anagramme schnell finden können. Was muss die Hashfunktion erfüllen ?

Betrachten Sie folgende Hashfunktion:

```
int hash1(String s) {
  int ret=0;
  byte[] sb=s.getBytes();
    for (int i=0; i<sb.length; i++) {
      ret=ret+sb[i];
    }
    return ret;
  }
```

(b) Zeigen Sie, dass die Hashfunktion hash1 zwei Anagramme auf den selben Hashwert abbildet.

(c) Warum ist die Funktion dennoch ungeeignet, um Anagramme zu filtern? Welche Strings werden noch auf dieselben Hashwerte abgebildet?

(d) Modifizieren Sie die Hashfunktion so, dass sie sich besser dazu eignet Anagramme zu filtern. Begründen Sie, warum sich Ihre neue Hashfunktion besser verhalten sollte und zeigen Sie, dass sie immer noch zwei Anagramme auf denselben Wert abbildet.

Aufgabe 3: **(Rekursion + Union-Find)** (6 Punkte)
Betrachten Sie im Union-Find-Problem die Weighted-Union-Rule, nach der bei der Operation $Union(A, B)$ die Größen der beiden Mengen miteinander verglichen werden. Sei o.b.d.A. A nicht größer als B, so erhält die Wurzel von A als neuen parent-Knoten die Wurzel von B. Beweisen Sie, dass die Find-Operationen in $O(\log n)$ durchführbar sind.

Aufgabe 4: **Rekursion** (6 Punkte)
Betrachten Sie folgende Rekursionsgleichung: Ist n eine Zweierpotenz, so ist

$$T(1) = 10$$

$$T(n) = 3T\left(\frac{n}{2}\right) + n$$

Geben Sie eine explizite Formel für $T(n)$, benutzen Sie nicht das Mastertheorem. Beweisen Sie die Korrektheit Ihrer Antwort.

Aufgabe 5: **(MST)** (6 Punkte)

Gegeben sei ein zusammenhängender ungerichteter Graph $G = (V, E)$ und Kostenfunktion $c : E \to \mathbb{R}^+$, so dass alle Kantenkosten paarweise verschieden sind.

Entscheiden Sie für folgende Aussagen ob sie richtig oder falsch sind! Begründen Sie jeweils ihre Antwort!

(a) Für jeden Knoten v befindet sich die billigste inzidente Kante zu v in einem minimalen aufspannenden Baum T.

(b) Für jeden Knoten v befindet sich die teuerste inzidente Kante zu v nicht in einem minimalen aufspannenden Baum T.

(c) Es kann einen Knoten v geben, für den alle inzidenten Kanten in einem minimalen aufspannenden Baum T vorhanden sind.

Aufgabe 6: **(AVL-Bäume)** (6 Punkte)

Erinnerung: Rot-Schwarz-Bäume sind binäre Suchbäume deren Knoten rot oder schwarz gefärbt sind, so dass folgende Eigenschaften gelten:

(a) Jeder Knoten ist schwarz oder rot.

(b) Die Wurzel ist schwarz.

(c) Jedes Blatt (NIL) ist schwarz.

(d) Falls ein Knoten rot ist, so sind beide Kinder schwarz.

(e) Für jeden Knoten gilt: Alle Pfade zu den Nachfahren, die Blätter sind, haben die gleiche Anzahl von schwarzen Knoten.

Wir vergleichen nun Rot-Schwarz-Bäume mit AVL-Bäumen.

(a) Zeigen Sie, dass nicht jeder Rot-Schwarz-Baum ein AVL-Baum ist.

(b) Zeigen Sie, dass sich jeder AVL-Baum zu einem Rot-Schwarz-Baum einfärben lässt.