Algorithmen und Programmieren 3 **Matrikelnummer:** ____________
WS 2003/2004, Zwischenklausur **FAMILIENNAME:** ________________
Dienstag, 20. Januar 2004. **Vorname:** ________________

Anleitung: Streichen Sie *eine* Aufgabe deutlich auf dem Angabeblatt. Diese Aufgabe wird nicht in die Bewertung einbezogen. Wenn Sie selbst keine Aufgabe streichen, wird die erste Aufgabe nicht in die Bewertung einbezogen. Bearbeiten Sie die übrigen drei Aufgaben.

Bearbeiten Sie jede Aufgabe auf einem getrennten Blatt. Jede Aufgabe hat 10 Punkte. Bearbeitungszeit: 90 Minuten

1. Man möchte einen binären Baum wie bei einer Halde als *array* speichern: Die Kinder des Knotens `a[i]` stehen an den Stellen `a[2*i]` und `a[2*i+1]`. Nehmen Sie an, dass die Bäume (im Gegensatz zu einer Halde) nicht ausgeglichen sein müssen. Bestimmen Sie die notwendige Größe des Arrays zum Speichern von n Elementen in Abhängigkeit von n,

   (a) wenn der Baum die Höhe $2\lceil \log n \rceil$ hat,
   (b) im schlimmsten Fall,
   (c) wenn die Höhe des Baumes nur um 2 größer als die kleinstmögliche Höhe ist.

   Geben Sie außerdem an, in welchem dieser Fälle man mit $O(n)$ Speicher auskommt.

2. Eliminieren Sie die Endrekursion aus dem folgenden Programmstück, das die Anzahl der übereinstimmenden Elemente in zwei Vektoren `a[0]`, ..., `a[n-1]` und `b[0]`, ..., `b[n-1]` zählt. Gehen Sie dabei systematisch Schritt für Schritt vor. Schreiben Sie eine einzige Methode als Ersatz für `anz`.

```
int anz(int[ ] a, int[ ] b, int n) { return anz2(a,b,n,0); }
int anz2(int[ ] a, int[ ] b, int n, int s)
{ if(n==0) return s;
  if(a[n-1]==b[n-1]) return anz2(a,b,n-1,s+1);
  return anz2(a,b,n-1,s);
}
```

3. Sortieren: Das folgende Programmstück soll n verschiedene `int`-Zahlen zwischen 1 und m sortieren, die in einem Array `a[0]`, ..., `a[n-1]` gespeichert sind.

```
int [ ] count = new int[m+1];
for (int j=1; j<=m; j++) count[j]=0;
for (int i=0; i<n;  i++) count[a[i]]++;
int i=0;
for (int j=1; j<=m; j++)
     for (int k=0; k<count[j]; k++)  {a[i]=j; i++;}
```

   (a) Bestimmen Sie die asymptotische Laufzeit dieses Programms.
   (b) Warum muss man in der letzten Zeile in der Anweisung `a[i]=j` keinen Überlauf des Indexes i über die Feldgrenzen von `a` befürchten? Geben Sie eine ausführliche Begründung.
   (c) Funktioniert dieses Verfahren auch, wenn die Zahlen nicht verschieden sind? Was muss man gegebenenfalls ändern, damit es für diesen Fall funktioniert?

4. Schreiben Sie ein Programm in Java oder Haskell, das für einen gegebenen binären Suchbaum die *mittlere innere Weglänge* berechnet; sie ist um 1 kleiner als die erwartete Anzahl von Vergleichen, die zum Finden eines zufällig ausgewählten Knotens in dem Baum im Durchschnitt erforderlich sind. (Jeder Knoten wird mit gleicher Wahrscheinlichkeit gesucht.)