Algorithmen und Datenstrukturen (PI.ADS.AD.VO)	schriftliche Einzelpruefung	20.10.2008		1

Aufgabe 1 [20]

Gegeben seien folgende Funktionen

```
void very_strange(int n, int digit) {
  for (int i=0; i<n; ++i) very_strange(n-1,digit);
}

void strange(int n, int digit) {
  if (digit==0) return;
  for (int i=0; i<n; ++i) strange(n,digit-1);
}

void not_so_strange(int n, int digit) {
  strange(n, digit);
  for (int i=0; i<7; ++i) not_so_strange(n/digit, digit);
  strange(n, digit);
}
```

Nehmen Sie an, dass für den Parameter *digit* die vierte Stelle Ihrer Matrikelnummer plus 1 übergeben wird und finden Sie Laufzeitabschätzungen in Theta-Notation (abhängig von n) für

a. [4] die Funktion *strange*,

b. [6] die Funktion *very_strange*,

c. [10] und die Funktion *not_so_strange*.

Aufgabe 2 [20]

a. [12] Sortieren Sie die Zeichen der Zeichenkette "STARMANIA" alphabetisch aufsteigend mit dem Quicksort-Verfahren. Geben Sie alle benötigten Zwischenschritte so genau an, dass der Ablauf des Algorithmus' klar ersichtlich wird.

b. [8] Nehmen Sie an, es wären die Ziffern Ihrer Matrikelnummer mit Quicksort zu sortieren. Als Pivotelement würde immer die letzte (ganz rechte) Ziffer gewählt. Wie müssen die Ziffern ursprünglich angeordnet sein, damit Quicksort die schlechteste Laufzeit erreicht, und wie müssen sie angeordnet sein, damit Quicksort die bestmögliche Laufzeit erreicht?

Aufgabe 3 [24]

Addieren Sie zu Ihrer Matrikelnummer die Zahl 43876951. Nehmen Sie an, die Ziffern der so erhaltenen Zahl wären (in der gleichen Reihenfolge in der sie in der Zahl auftreten) in einem Array gespeichrt. Sortieren Sie dieses Array

a. [8] mit Heapsort

b. [8] mit Bucketsort

c. [8] mit Counting Sort

Geben Sie alle benötigten Zwischenschritte so an, dass der Ablauf des jeweiligen Algorithmus' klar ersichtlich wird.

Aufgabe 4 [16]

Gegeben sei eine Datenstruktur zur Realisierung einer Hashtabelle in C++, die double hashing als Kollisionsbehandlung verwendet. Die beiden Methoden *hash1* und *hash2* berechnen die erste respektive zweite Hashfunktion für einen Schlüsselwert, den sie als Parameter erhalten. *size* sei eine vordefinierte Konstante, die die Größe der Hashtabelle festlegt.

```
enum status {frei,belegt,wiederfrei};
class HTable {
  int val[size];
  status wiederfrei[size];

  int hash1(int);
  int hash2(int);

public:
  bool find(int);
};
```

Schreiben Sie die Definition der Methode *bool find(int)* in einem C++-ähnlichen Pseudocode. Die Funktion sucht nach einem Schlüsselwert, der als Parameter übergeben wird und liefert *true*, falls der Schlüsselwert gefunden wurde, *false* sonst.

Aufgabe 5 [20]

a. [10] Fügen Sie die Werte 5, 8, 3, 4, 6, 7, 1, 2, 9, 11, 10 in dieser Reihenfolge in einen ursprünglich leeren *B+-Baum der Ordnung 1* ein.

b. [4] In dem so erstellten B+-Baum sind die Einträge in der Blattebene alle sortiert. Ist das eher ungewöhnlich, oder würden Sie das erwarten? (Kurze Begründung Ihrer Antwort.)

c. [6] Löschen Sie die Werte 5, 6 und 4 (in dieser Reihenfolge) aus dem oben erstellten B+-Baum.

Geben Sie jeweils den Zustand des Baumes nach jeder Einfüge- bzw. Löschoperation an.