Name, Vorname, Unterschrift:	Matrikelnummer:

Klausur
Kompaktkurs Einführung in die Programmierung
(Dr. S. Zimmer)
23. März 2009

Wichtig:

- Die Angabe besteht aus 6 Seiten — prüfen Sie bitte Ihr Exemplar auf Vollständigkeit.
- Kennzeichen Sie jedes Blatt lesbar mit Ihrem Namen und dieses Deckblatt zusätzlich mit Ihrer **Matrikelnummer und Unterschrift**; trennen Sie die Heftung nicht auf.
- Bearbeiten Sie die Aufgaben auf den Angabenblättern. Sollte der Platz für einen Aufgabenteil nicht reichen, benutzen Sie bitte die Rückseite der Angabenblätter und machen Sie einen entsprechenden Vermerk bei der Aufgabe.
- Insgesamt können Sie 40 Punkte erreichen, bei jeder Aufgabe sind die für die jeweiligen Teilaufgaben vorgesehenen Punkte angegeben. Zum Bestehen sind 17 Punkte erforderlich.
- Als Hilfsmittel ist nur ein Ausdruck der Vorlesungsfolien erlaubt (handschriftliche Anmerkungen sind OK).
- Eigene Funktionen dürfen in den folgenden Teilaufgaben weiterverwendet werden; von den Standardfunktionen sind, soweit nicht bei der Aufgabe anders angegeben, nur die zur Ausgabe zulässig (`std::cout` bzw. `printf`, was immer Ihnen davon lieber ist).
- Die Bearbeitungszeit beträgt 75 Minuten.

Aufg.	1a)	1b)	1c)	1d)	2a)	2b)	2c)	3a)	3b)	Summe	Note
Pkt.	/6	/4	/4	/5	/3	/4	/5	/4	/5	/40	

1 Bergpanorama (max. 6+4+4+5=19 Punkte)

Eine Zeichenkette, die nur aus den Buchstaben `'H'` und `'R'` (und natürlich der Endemarkierung `'\0'`) besteht, wird hier zur Darstellung eines Gebirges verwendet, indem sie den Umriss folgendermaßen beschreibt:

- Jedes Zeichen hat eine ganzzahlige Höhe, die von links nach rechts berechnet wird, anfangend bei einer Höhe 0.
- Ein `'H'` („Hoch“) bekommt den aktuellen Wert der Höhe, anschließend wird diese (für die Bearbeitung des nächsten Zeichens) um eins erhöht.
- Für ein `'R'` („Runter“) wird zuerst der aktuelle Wert der Höhe um eins erniedrigt, der entstehende Wert ist die Höhe des `'R'` und mit diesem Wert wird auch weitergerechnet.

Das Gebirge zur Zeichenkette `"HHHRRHHHRRR"` sieht dann (jedes Zeichen in seiner Höhe eingezeichnet) so aus (ein kleines Gebirge mit zwei Gipfeln):

```
      HR
  HR  H  R
 H  RH    R
H
```

Zeichen	H	H	H	R	R	H	H	H	R	R	R
Höhe	0	1	2	2	1	1	2	3	3	2	1

a) Schreiben Sie eine Funktion `hoehe` mit einer Zeichenkette s und einer ganzen Zahl i als Parameter und ganzzahligem Ergebnis, das die Höhe vom Zeichen mit Index i der Zeichenkette angibt (also den zugehörigen Wert aus obiger Tabelle). Sie dürfen davon ausgehen, dass s keine unzulässigen Zeichen enthält und i nichtnegativ ist. Wenn s weniger als $i+1$ Zeichen enthält (es also kein Zeichen mit Index i gibt), soll das Ergebnis -1 sein.

(Sollte Ihnen diese Funktion Schwierigkeiten bereiten, setzen Sie sie einfach in den folgenden Teilaufgaben als gegeben voraus.)

b) Schreiben Sie eine Funktion `max_hoehe` mit einer Zeichenkette (die wieder – wie auch in allen weiteren Teilaufgaben – keine unzulässigen Zeichen enthalte) als Parameter, die die maximale Höhe des Gebirges als Ergebnis zurückliefert (im Beispiel von oben also 3).

c) Nun eine Funktion `nichtnegativ` mit einer Zeichenkette s als Parameter und einem Wahrheitswert als Ergebnis, der genau dann `true` ist, wenn kein Zeichen von s eine negative Höhe hat.

d) Schreiben Sie schließlich eine Funktion mit einer Zeichenkette s als Parameter und ohne Funktionsergebnis, die das Gebirge in der Art des obigen Beispiels druckt:

```
        HR
  HR   H   R
 H   RH     R
H
```

Sie dürfen dabei davon ausgehen, dass kein Zeichen von s eine negative Höhe hat. Es sollen nicht mehr Zeilen gedruckt werden, als Platz für das Gebirge benötigt wird.

2 Vektoren (max. 3+4+5=12 Punkte)

Zum Speichern von Vektoren soll in dieser Aufgabe ein Strukturtyp verwendet werden, der aus einer ganzen Zahl und einem Zeiger besteht. Die Zahl wird die Länge des Vektors (Anzahl seiner Komponenten) speichern, der Zeiger wird auf die erste Komponente eines Feldes verweisen, das die Vektorkomponenten (Typ `double`) enthält.

a) Geben Sie einen passenden Strukturtyp `struct vektor` an:

b) Schreiben Sie eine Funktion mit einem ganzzahligen Parameter n, die einen Vektor (vom eben definierten Typ) mit n Komponenten zurückliefert. Für das benötigte Feld soll die Funktion neuen Speicherplatz anfordern (`new` oder `malloc`), die Vektorlänge soll natürlich richtig belegt sein.

c) Schreiben Sie nun eine Funktion mit einem Vektor als Parameter und ohne Funktionsergebnis, die die Komponenten auf dem Bildschirm ausgibt. Zwischen zwei Komponenten soll ein Komma gedruckt werden, hinter der letzten Komponente kein Komma, sondern ein Zeilenumbruch `'\n'`.

3 Binärbäume (max. 4+5=9 Punkte)

In dieser Aufgabe werden Binärbäume bearbeitet mit der in der Vorlesung vorgestellten Datenstruktur

```
struct knoten { int x;   struct knoten *l, *r; };
```

für die Knoten; ein Binärbaum wird wie üblich durch einen Zeiger auf seine Wurzel dargestellt, mit Ausnahme des leeren Binärbaums, der durch den Nullzeiger dargestellt wird.

a) Schreiben Sie eine Funktion mit einem Binärbaum als Parameter, die die Zahl der im Baum enthaltenen Knoten als Ergebnis zurückliefert.

b) Wir nennen in dieser Aufgabe einen Baum *schief*, wenn in jedem seiner Knoten der linke Sohn (d.h., der dort wurzelnde Teilbaum) wenigstens so viele Knoten enthält wie der rechte Sohn. Schreiben Sie eine Funktion mit einem Binärbaum als Parameter, die einen schiefen Baum als Ergebnis hat. Dieser entstehe, indem an allen Knoten im Eingangsbaum, die das Schiefheitskriterium verletzen, die Söhne vertauscht werden.

   Volle Punkte gibt's hier nur, wenn Sie überflüssige Baumdurchläufe vermeiden! Platz zum Schreiben ist hier recht knapp bemessen, ggf. bitte auf der Rückseite weiterschreiben.