Universität Mannheim
Fakultät für Mathematik und Informatik
Lehrstuhl für Praktische Informatik IV
Professor Dr. W. Effelsberg

Teilprüfung
Software- und Internet-Technologie
März 2004: Programmierkurs 2

Name: .... Vorname: ....

Matrikel-Nr.: .... Semester: .... Fach: ....

Hinweise:

1. Bitte füllen Sie sofort den Kopf des Deckblattes aus.
2. Unterschreiben Sie die Klausur auf der letzten Seite.
3. Überprüfen Sie bitte Ihr Klausurexemplar auf Vollständigkeit (**11** Seiten).
4. Tragen Sie die Lösungen – soweit möglich – direkt in die Klausur ein.
5. Zugelassene Hilfsmittel: nicht programmierbarer Taschenrechner
6. Bearbeitungszeit: 66 Minuten.

Aufgabe	max. Punktzahl	Punkte
1	18	
2	12	
3	13	
4	12	
5	11	
Summe	66	

Aufgabe 1: Verständnisfragen *[18 Punkte]*

a) [3 Punkte] Geben Sie ein Programmfragment in C an, das den Präfix-Operator `++` enthält. Das Programm soll so gestaltet sein, dass sich sein Verhalten ändert, wenn man stattdessen den Postfix-Operator verwendet. Erläutern Sie das Verhalten Ihres Programmes in beiden Fällen.

b) [3 Punkte] In einem C-Programm sei ein Feld wie folgt deklariert:

```
int feld[3][4];
```

Geben Sie ein Programmfragment in C an, mit dem der Inhalt dieses Feldes am Bildschirm ausgegeben werden kann. Dabei darf der `*`-Operator nicht verwendet werden.

c) [3 Punkte] Lösen Sie die gleiche Aufgabe wie in Teil b), aber ohne dabei den `[ ]`-Operator zu benutzen.

d) [3 Punkte] Gegeben sei die folgende Funktion:

```
void output(int* a) {
  for( ; (*a) > 0; (*a)--)
    printf("%i\n",(*a));
}
```

Was gibt die Funktion aus, wenn sie mit dem Aufruf

```
int x = 6;
output(&x);
```

aufgerufen wird? Welchen Wert hat `x` nach Ausführung der Funktion?

e) [3 Punkte] Zu Beginn eines Unterprogrammes in M68000-Assembler soll der Inhalt der Datenregister `D0` bis `D7` auf den Stack gerettet werden, bei Verlassen des Unterprogrammes wird der ursprüngliche Inhalt wiederhergestellt. Geben Sie die dazu jeweils erforderlichen Anweisungen in M68000-Assembler an.

f) [3 Punkte] Es stehe in Register `D0` der Wert `$1234` und in Register `A0` der Wert `$FEDC`. Nun wird der Befehl

```
MOVE.L   #1,$4444(A0,D0)
```

ausgeführt. Welche Bytes im Speicher werden dadurch verändert, und welchen Wert haben sie anschließend?

Aufgabe 2: C-Programmierung *[12 Punkte]*

Die Zahl π kann durch die folgende Berechnung approximiert werden:

$$\pi = 4 - \frac{4}{3} + \frac{4}{5} - \frac{4}{7} + \frac{4}{9} - \frac{4}{11} + \dots$$

Schreiben Sie ein vollständiges C-Programm, das eine solche Approximation für π berechnet:

- Lesen Sie zunächst eine echt positive ganze Zahl n von der Standardeingabe ein, bei unzulässiger Eingabe wird das Programm mit Fehlermeldung beendet.
- Berechnen Sie dann die Annäherung von π mit der obigen Formel, indem Sie die ersten n Summanden verwenden (4 ist der erste Summand, $-\frac{4}{3}$ der zweite usw.).
- Geben Sie das Ergebnis auf der Standardausgabe aus.

(Platz zum Lösen von Aufgabe 2)

Aufgabe 3: Programmierung mit Zeigern *[13 Punkte]*

Der Datentyp `knoten` einer verketteten Liste sei wie folgt definiert:

```
typedef struct node{
  int value;
  struct node* next;
} knoten;
```

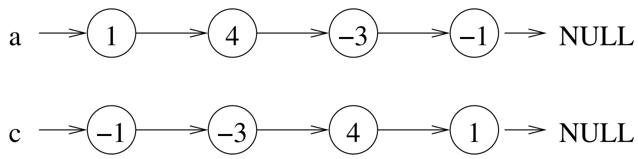
Dieser Datentyp wird nun verwendet, um Integer-Vektoren beliebiger Länge zu implementieren.

Schreiben Sie eine C-Funktion

```
knoten* umkehrung(knoten* a) {...}
```

die einen solchen Vektor als Eingabe erhält und einen neuen Vektor zurückgibt, der die gleichen Werte in umgekehrter Reihenfolge enthält. Wichtig ist dabei, dass der ursprüngliche Vektor nicht verändert werden darf!

Bsp.: `c = umkehrung(a);`



(Platz zum Lösen von Aufgabe 3)

Aufgabe 4: Assembler-Programmierung *[12 Punkte]*

Das folgende einfache C-Programm beschreibt die Steuerung eines Thermostats.

```
int main() {
  int soll_temperatur, ist_temperatur;
  int heizen = 0;
  while(1) {
    soll_temperatur = getRegler();
    ist_temperatur = getSensor();
    if(ist_temperatur < soll_temperatur)
      heizen = 1;
    else
      heizen = 0;
  }
  return 0;
}
```

Das Thermostat liest also die Soll-Temperatur vom Regler und die Ist-Temperatur von einem Sensor (Thermometer) ab. Sie lässt dann die Heizung genau so lange eingeschaltet, wie die Ist-Temperatur unter der Soll-Temperatur liegt.

Schreiben Sie ein gleichwertiges Programm in M68000-Assembler. Gehen Sie dabei davon aus, dass die Soll-Temperatur bzw. die Ist-Temperatur als Worte unter den Adressen `$8000` bzw. `$8002` bereitstehen. Die Heizung wird unter der Adresse `$8004` an- bzw. ausgeschaltet, indem man 0 bzw. 1 an diese Adresse schreibt.

Bem.: Es ist <u>kein</u> vollständiges Assembler-Programm gefragt. Insbesondere müssen keine Assemblerdirektiven (wie `ORG $0` usw.) angegeben werden!

Das Programm ist ausführlich zu kommentieren; unkommentierte Programme werden nicht bewertet!

(Platz zum Lösen von Aufgabe 4)

Aufgabe 5: Rekursive Funktionen *[11 Punkte]*

Das Quadrat einer echt positiven ganzen Zahl x kann rekursiv berechnet werden wie folgt:

$$x^2 = \begin{cases} 1 & \text{falls } x = 1 \\ (x-1)^2 + 2x - 1 & \text{sonst} \end{cases}$$

Schreiben Sie eine rekursive Funktion `SQUARE` in M68000-Assembler, die mit dieser Methode das Quadrat berechnet. Der Eingabewert x steht dabei in Register `D0`, der Rückgabewert x^2 soll in Register `D1` stehen. Sie können davon ausgehen, dass die Zahl in Register `D0` tatsächlich eine echt positive ganze Zahl ist.

Das Programm ist ausführlich zu kommentieren; unkommentierte Programme werden nicht bewertet!