Universität Mannheim
Fakultät für Mathematik und Informatik
Lehrstuhl für Praktische Informatik IV
Prof. Dr. Wolfgang Effelsberg

Teilprüfung
Software- und Internettechnologie
Programmierkurs 2
Sommersemester 2004

Name: ....

Vorname: ....

Matrikel-Nr.: ....

Studienfach: ....

Hinweise:

1. Überprüfen Sie die Klausur auf Vollständigkeit (**10** Seiten).
2. Tragen Sie die Lösungen direkt in die Klausur ein. Benutzen Sie ggf. auch die Rückseiten der Aufgabenblätter.
3. Lösungen auf farbigem Konzeptpapier werden **nicht** bewertet.
4. Zugelassene Hilfsmittel: nicht programmierbarer Taschenrechner
5. Die Bearbeitungszeit beträgt 66 Minuten.

Aufgabe	max. Punktzahl	erreichte Punktzahl
1	14	
2	14	
3	15	
4	8	
5	15	
Summe	66	

Aufgabe 1: Verständnisfragen [14 Punkte]

a) [3 Punkte] Bestimmen Sie die Fehler in folgender C-Funktion, die das Minimum der Zahlen a und b berechnen soll, und geben Sie jeweils an, ob es sich um syntaktische, semantische oder logische Fehler handelt.

```
int min(int a, int b){
  if (a = b)
    return a;
  if (a < b)
    return b
  else
    return a;
}
```

b) [3 Punkte] Welchen Wert haben die Variablen a, b, c nach Ausführung des folgenden C-Codes (mit Begründung)?

```
unsigned int a = 0x1a;
unsigned int b = 3;
unsigned int c = 1;

b = a | b;
a = c && b;
c = b >> a;
```

c) [4 Punkte] Gegeben sei die folgende, nicht korrekte C-Funktion.

```
void switchCase(unsigned int note){

  if (note == 0 || note > 5) printf("ungueltige Eingabe\n");

  switch (note){
    case 4 : printf("knapp bestanden\n");
    case 5 : printf("nicht bestanden\n");
    default: printf("bestanden\n");
  }
}
```

Wie lautet jeweils die Bildschirmausgabe für die Aufrufe `switchCase(4)` und `switchCase(5)`?

Modifizieren Sie den Code der Funktion so, dass er das Gewünschte leistet.

d) [4 Punkte] Nennen Sie zwei Adressierungsarten des MSP430 und beschreiben Sie diese kurz anhand eines Codebeispiels.

Aufgabe 2: C-Programmierung [14 Punkte]

Herr Müller ist Diplom-Mathematiker und sucht für sein neues Auto nach einem geeigneten Kennzeichen. Ein für Mannheim zulässiges Pkw-Kennzeichen besteht aus dem Unterscheidungszeichen `MA`, genau zwei Zwischenbuchstaben aus $\{A, B, \ldots, Z\}$ (ohne Umlaute) und einer Zahl aus $\{100, 101, \ldots, 999\}$. Die Zwischenbuchstabenkombinationen `NS`, `SA`, `SS`, `HJ` und `KZ` werden aus historischen Gründen nicht vergeben. (Tatsächlich existieren noch weitere Einschränkungen, die wir hier allerdings nicht betrachten wollen.) Ihre Aufgabe ist es, Herrn Müller bei der systematischen Suche nach einem geeigneten Kennzeichen zu unterstützen.

a) [6 Punkte] Die Zahl des Kennzeichens soll eine Primzahl sein.

Schreiben Sie eine C-Funktion

```
int istPrimzahl(unsigned int x) {...}
```

die den Wert 1 zurückgibt, falls x eine Primzahl ist, und 0 sonst.

Zur Erinnerung: 0 und 1 sind keine Primzahlen. Eine natürliche Zahl $x \geq 2$ ist genau dann eine Primzahl, wenn sie nur durch 1 und sich selbst teilbar ist.

b) [8 Punkte] Nachdem sich Herr Müller für eine Zahl entschieden hat, möchte er eine Liste von Kandidatenkennzeichen mit dieser Zahl erzeugen.

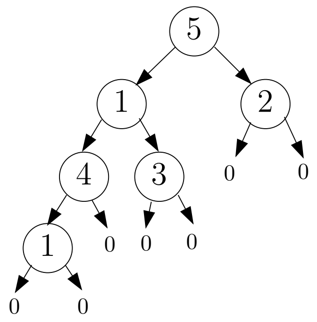
Schreiben Sie eine C-Funktion

`void printKandidaten(unsigned int x)` {...}

die alle zulässigen Mannheimer Pkw-Kennzeichen mit der Zahl x (falls vorhanden) ausgibt.

Aufgabe 3: Dynamische Datenstrukturen [15 Punkte]

In einem Binärbaum haben alle Knoten höchstens zwei Nachfolger, und jeder Knoten außer der Wurzel hat genau einen Vorgänger. In jedem Knoten wird ein `unsigned int` gespeichert.
Der Datentyp `knoten` sei daher wie folgt definiert:




```
typedef struct node{
  unsigned int wert;
  struct node *linkerSohn;
  struct node *rechterSohn;
} knoten;
```

Falls ein Knoten keinen linken bzw. rechten Sohn hat, gilt `linkerSohn=NULL` bzw. `rechterSohn=NULL`. Ein Blattknoten ist ein Knoten, der weder einen linken noch einen rechten Sohn hat.

a) [5 Punkte] Schreiben Sie eine C-Funktion

```
knoten* neuerKnoten(unsigned int neuerWert,
                    knoten* links,
                    knoten* rechts)
```

die einen neuen Knoten mit den übergebenen Parametern anlegt und einen Zeiger auf den Knoten zurückliefert.

b) [10 Punkte] Schreiben Sie eine **rekursive** C-Funktion

```
int enthaeltBlatt(knoten* v, unsigned int b_wert)
```

die den Wert 1 zurückgibt, wenn der Binärbaum mit der Wurzel `v` einen **Blattknoten** mit dem Wert `b_wert` enthält. Falls kein solcher Blattknoten existiert, soll die Funktion den Wert 0 zurückgeben.

Tipp: Ein Binärbaum enthält genau dann einen Blattknoten mit dem Wert `b_wert`, wenn der linke oder der rechte Teilbaum einen Blattknoten mit dem Wert `b_wert` enthält.

Aufgabe 4: Sensorknotensteuerung [8 Punkte]

An Tunneleinfahrten sind oft Warnlichter angebracht, die in regelmäßigen Abständen gelb blinken.

Schreiben Sie ein **Unterprogramm** in MSP430-Assembler, das in einer Endlosschleife ein solches Warnlicht mit der gelben Leuchtdiode des in der Übung verwendeten Sensorknotens realisiert. Ihr Unterprogramm soll die folgende Form besitzen:

```
.blink:     ...             ; Ihre Assemblerbefehle
            ...
            ...
            ret
```

Ein Hauptprogramm o.ä., das das Unterprogramm aufruft, braucht nicht angegeben zu werden.

Hinweise:

- Die drei Leuchtdioden des Sensorknotens werden über das Byte an der (absoluten) Adresse `0x0029` im Speicher des Prozessors gesteuert. Das niederwertigste Bit (Bit 0) an dieser Adresse steuert die rote, das Bit 1 die grüne und das Bit 2 die gelbe Leuchtdiode. Eine Leuchtdiode ist genau dann eingeschaltet, wenn das entsprechende Bit auf 0 gesetzt ist.
- Um die Blinkintervalle zu steuern, kann die Unterprozedur `wait` verwendet werden. Die Wartedauer wird über die Register `R14` und `R15` übergeben, wobei die höherwertigen 16 Bit der Wartedauer in `R15` erwartet werden.

Aufgabe 5: MSP430-Assembler [15 Punkte]

Herr Müller hat die Korrektur der Klausur Analysis I abgeschlossen. Zusätzlich zur Ergebnisliste mit Matrikelnummern und zugehörigen Noten möchte er den Notenspiegel aushängen, d.h. eine Tabelle, die angibt, wie oft welche Note vergeben worden ist. Nachdem er fünf Strichlisten mit fünf verschiedenen Ergebnissen produziert hat, bittet er Sie entnervt um Unterstützung.

Schreiben Sie ein **Unterprogramm** in MSP430-Assembler, das aus einer im Speicher abgelegten Ergebnisliste den zugehörigen Notenspiegel berechnet.

Der Anfang der Ergebnisliste wird in Register `R10` und die Anzahl der Listeneinträge in Register `R11` übergeben. Jeder Listeneintrag der Ergebnisliste besteht aus zwei Bytes für die Matrikelnummer und zwei Bytes für die Note. Der Einfachheit halber enthalte die Liste nur Noten aus $\{1, 2, 3, 4, 5\}$.

Der Notenspiegel soll an der Adresse gespeichert werden, die in Register `R12` übergeben wird. Jeder Listeneintrag des Notenspiegels besteht aus zwei Bytes für die Note und zwei Bytes für die Anzahl der Studenten, die diese Note erzielt haben. Gehen Sie davon aus, dass der Speicherbereich, in dem der Notenspiegel abgelegt werden soll, uninitialisiert ist.

Beispiel:

Parameter

```
R10: 0x0300
R11: #6
R12: 0x0400
```

Ergebnisliste

Adresse	Inhalt
0x0300	#16215
0x0302	#3
0x0304	#16216
0x0306	#5
0x0308	#16220
0x030a	#2
0x030c	#16227
0x030e	#3
0x0310	#16230
0x0312	#2
0x0314	#16245
0x0316	#1

Notenspiegel

Adresse	Inhalt
0x0400	#1
0x0402	#1
0x0404	#2
0x0406	#2
0x0408	#3
0x040a	#2
0x040c	#4
0x040e	#0
0x0410	#5
0x0412	#1

Ihr Unterprogramm soll die folgende Form besitzen:

```
.marks:    ...              ; Ihre Assemblerbefehle
           ...
           ...
           ret
```

Ein Hauptprogramm o.ä., das das Unterprogramm aufruft, braucht nicht angegeben zu werden.

Kommentieren Sie Ihr Programm sinnvoll! Unkommentierter Code wird nicht bewertet.

(Platz für die Lösung von Aufgabe 5)