Friedrich-Alexander-Universität Erlangen-Nürnberg

Schriftliche Prüfung zur Vorlesung
„Einführung in die Programmierung für Fertigungstechniker”

Aufgabe 1	Aufgabe 2	Aufgabe 3	Aufgabe 4	Aufgabe 5	Gesamt
40	**10**	**10**	**10**	**10**	**80**

(Name) (Vorname) (Matrikel-Nr.)

(Fachrichtung) (Semester)

Ihre Scheine gehen direkt an das Prüfungsamt !

Erlangen, den 19.08.1994

Organisatorische Hinweise

Die folgenden Informationen bitte aufmerksam lesen und die Erklärung am Blattende unterschreiben.

- Die Bearbeitungszeit beträgt 120 Minuten.
- Es sind **k e i n e** Hilfsmittel zugelassen.
- Schmierpapier darf nicht mit abgegeben werden. Die Lösung einer Aufgabe soll auf das Aufgabenblatt in den dafür vorgesehenen freien Raum geschrieben werden, der jeweils der Aufgabe folgt. Sollte der Platz nicht ausreichen, so können Sie die Rückseiten der Aufgabenblätter mitverwenden.
- Bei Bedarf sind von der Aufsicht Schmierpapier (farbig) und zusätzliche Lösungsblätter (weiß) zu erhalten. Vermerken Sie vor dessen Verwendung unbedingt Ihren Namen und Ihre Matrikelnummer darauf!
- Fragen zu den Prüfungsaufgaben können grundsätzlich nicht beantwortet werden.
- Tragen Sie Ihren Namen und Vornamen, Ihre Matrikelnummer, Studienfachrichtung und Fachsemesterzahl auf dem Deckblatt der Klausur ein.
- Bitte legen Sie Ihren Studenten- und einen Lichtbildausweis zur Kontrolle bereit.
- Sie dürfen den Raum nicht verlassen, bevor Ihre Personalien überprüft wurden, und Sie die Klausurunterlagen der Aufsicht zurückgegeben haben.
- Bleiben Sie bitte am Ende der Bearbeitungszeit solange an Ihrem Platz sitzen, bis alle Klausurunterlagen eingesammelt sind.

Erklärung

Durch meine Unterschrift bestätige ich

- den Empfang der vollständigen Klausurunterlagen (8 Seiten inkl. Deckblatt)
- die Kenntnisnahme der obigen Informationen.

Ich bin damit einverstanden, daß mein Prüfungsergebnis unter Angabe der Matrikel-Nr. veröffentlicht wird.

☐ ja ☐ nein

Erlangen, den 19.08.1994

(Unterschrift)

Aufgabe 1: (40 Punkte)

Schreiben Sie ein Programm, welches einen beliebig langen Text (einzelne Zeilen sind durch *Newline* voneinander getrennt) von der Tastatur einliest und transformiert auf den Bildschirm ausgibt. Als Transformation soll Ihr Programm **generell** alle Leerzeichen am Ende einer jeden Textzeile entfernen. Ist bei Aufruf Ihres Programmes die **Aufrufoption -r** gesetzt, sollen **zusätzlich** alle Zeilen, welche lediglich aus Leerzeichen bestehen, komplett (also inkl. *Newline*-Zeichen) aus dem Text entfernt werden.

Bei Fehlaufrufen sollte Ihr Programm mit einer Fehlermeldung und einem Fehlercode ungleich null terminieren.

Erstellen Sie zuerst ein Struktogramm und dokumentieren Sie Ihr Programm mit kurzen, aber aussagekräftigen Kommentaren! Auf das Struktogramm entfällt die Hälfte der Punkte!

Aufgabe 2: (10 Punkte)

Was berechnet das folgende (sinnlose) Programm? Geben Sie die Inhalte der verwendeten Variablen nach jeder ausgeführten Anweisung an! Verwenden Sie dazu die untenstehende Tabelle!

```
int k[] = { 2, 3, 10 };          (1)
int *j  = k+1;                   (2)
void f(int *k)                   (3)
{  int m = *k;                   (4)
   if (m/2*2 == m)               (5)
      *j   += (m * m);           (6)
   else                          (7)
      k[1] *= (m + m);           (8)
}                                (9)
void main()                      (10)
{  int m=*k;                     (11)
   f(k);                         (12)
   m += k[1];                    (13)
   f(j);                         (14)
   m += ++k[2];                  (15)
}                                (16)
```

Zeile							

Aufgabe 3: (10 Punkte)

Markieren und verbessern Sie die Fehler in dem aus der Vorlesung bekannten C-Programm, welches prüfen soll, ob ein vom Anwender eingegebenes Datum eine bestimmte Form besitzt. Ein eingegebenes Datum der Form **tag.monat.jahr** (z.B. 1.1.1994) bzw. **tag. monatsname jahr** (z.B. 1. Januar 1994) soll als gültig, alle anderen Eingaben dagegen sollen als ungültig erkannt werden.

Als Fehler markierte korrekte Teile führen zu Punktabzug. Die Tatsache, daß das Programm kaum Kommentare enthält und kein Struktogramm beigefügt ist, wird ausnahmsweise nicht als Fehler betrachtet.

```
#include stdio.h

int main()
{
    char buffer[BUFSIZ],mname[2];
    int  tag,monat,jahr;

    printf("\n\t\tFormale Datums-Pruefung\n");

    while(1) {  /* Verlassen der Schleife, wenn gets()
                   NULL-Pointer liefert oder eine */
                   Leerzeile eingegeben wird (s.u.)
                */
       printf("\nBitte ein Datum eingeben : ");

       if (!gets(buffer) || buffer[0])
          break;

       rv=sscanf(buffer,"%d. 19s %d",&tag,mname,&jahr);
       if (rv=3)
          printf("gueltig: %s\n",buffer);
       else
          rv=sscanf(buffer,"%d.%d           %d",&tag,&mo-
nat,&jahr);
          if (rv==3)
             printf("gueltig: %s\n",buffer);
          else
             printf("ungueltig: %s\n",buffer);

    }
    return;
}
```


Aufgabe 4: (10 Punkte)

Angenommen, ein C-Programm enthält eine **for**-Schleife der Form

```
for ( Ausdruck1; Ausdruck2; Ausdruck3 )
      Anweisung
```

Wie ist diese Schleife dann durch eine **while**-, wie durch eine **do-while**-Schleife zu ersetzen ?

Aufgabe 5: (10 Punkte)

Entwerfen Sie geeignete Datenstrukturen zur sortierten Speicherung von ganzen Zahlen in Form eines Binärbaumes (gehen Sie hier davon aus, daß nicht mehr als 100 Zahlenwerte zu sortieren sein werden)!