Friedrich-Alexander-Universität Erlangen-Nürnberg

**Schriftliche Prüfung zur Vorlesung**
**“Einführung in die Programmierung für Ingenieure (C)”**

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

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

(Fachrichtung) (Semester)

Erlangen, den 14.08.1996





## Organisatorische Hinweise

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

- Die Bearbeitungszeit beträgt 120 Minuten.
- Es sind **keine** 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! **Schmierpapier darf nicht mit abgegeben werden!**
- 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 (9 Seiten inkl. Deckblatt).
- die Kenntnisnahme der obigen Informationen.

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

☐ ja ☐ nein

Der Schein soll direkt an das Prüfungsamt geschickt werden:

☐ ja ☐ nein

Erlangen, den 14.08.1996

(Unterschrift)

**Aufgabe 1: ( 14+26 Punkte)**

Schreiben Sie ein Programm zur statistischen Auswertung von Meßwerten. Ihr Programm liest dazu von der Standardeingabe zuerst die Anzahl zu verarbeitender Werte und anschließend die eigentlichen Werte ein. Aus diesen Werten berechnet das Programm den Mittelwert und die Standardabweichung und gibt sie aus.

Bearbeiten Sie dazu folgende beiden Teilaufgaben:

a) Für eine Folge $a_i, i = 1 \ldots n$ von Meßwerten berechnen sich der Mittelwert $E_a$, und die Standardabweichung $D_a$ folgendermaßen:

$$E_a = \frac{1}{n}\sum_{i=1}^{n} a_i \qquad D_a = \sqrt{\sum_{i=1}^{n}(a_i - E_a)^2}$$

Sowohl der Mittelwert, als auch die Standardabweichung sollen dabei als eigenständige Funktionen implementiert werden, die die folgenden Prototypen realisieren:

Mittelwert:

```
double average(int cnt, double *values);
```

Standardabweichung:

```
double deviation(int cnt, double *values);
```

Beide Funktionen erhalten in *cnt* die Anzahl der Meßwerte, die in dem Feld *values* enthalten sind. Die Hälfte der Punkte entfällt auf die ebenfalls zu erstellenden Struktogramme.





b) Erstellen Sie das Hauptprogramm, das die nachfolgend beschriebenen Kommandozeilenparameter verarbeiten kann, die Meßwerte von der Standardeingabe einliest, mittels der in a) erstellten Funktionen verarbeitet und die Ergebnisse auf die Standardausgabe ausgibt.

- **-a** Das Programm gibt nicht mehr automatisch Mittelwert und Standardabweichung aus, sondern nur noch den Mittelwert.
- **-d** Wie **-a**, jedoch mit der ausschließlichen Ausgabe der Standardabweichung.

Selbstverständlich können auch mehrere Optionen angegeben werden!

Achten Sie darauf, daß Ihr Programm angemessen auf falsche Kommandozeilenoptionen reagiert. Sie können aber davon ausgehen, daß nur korrekte Eingabedaten auf die Standardeingabe geliefert werden.

Erstellen Sie auch hier zuerst wieder ein Struktogramm, aus dem der Aufbau Ihrer Funktion hervorgeht, und codieren Sie die Funktion anschließend in C, wobei auf kurze aber aussagekräftige Kommentare zu achten ist! Auf die Struktogramme entfällt wieder die Hälfte der Punkte.

**Aufgabe 2: (10 Punkte)**

Markieren und verbessern Sie die Fehler (max. 15 Stück!) in dem untenstehenden Programm, welches einen Text von der Standardeingabe einliest und seitenweise mit Seitenzahlen versehen auf die Standardausgabe ausgibt. Die Standardseitenlänge von 60 Zeilen kann mittels der Option **-n Zeilenzahl** verändert werden. (Achtung: `atoi` ist eine in `stdlib.h` korrekt deklarierte Funktion zum Umwandeln eines Strings in einen Integerwert, '`\014`' ein korrektes Steuerzeichen für den Seitenumbruch!)

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#define MAX_LINELENGTH 80;

int main(int argc, int *argv)
{
    int lines_per_page := 60; \* Default: 60 Zeilen *\
    int line = 0;
    void page = 1;
    char zeile[MAX_LINELENGTH+1];
    if((argc!=1) && (argc!= 3))
        printf("usage: print [-n lines_per_page]\n");
        return;
    }
    if((argc==3) && (strcmp(argv[1], '-n')=0)){
        lines_per_page = atoi(argv[3]);
        if(lines_per_page < 3) {
            printf("zu kleine Zeilenzahl: $d\n";
                      lines_per_page);
            return 1;
        }
    }
    while(zeile = gets(zeile)) {
        printf("%s\n",zeile);
        if(++line == lines_per_page-2) {
            printf("\n\t\t-- %d --\n\014", page++);
            line = 0;
        }
    }
    if(line>0) {
        for(; line<lines_per_page-2; line++)
            printf("\n");
        printf("\n\t\t-- %d --\n\014");
    }
}
```

**Aufgabe 3: (10 Punkte)**

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

```
 1  #include <stdio.h>
 2  #include <stdlib.h>
 3  int k[] = {3, -2, 3, -4, 5, 6, 7};
 4  int i = 5;
 5  int f(int *a, int *b)
 6  {
 7      i = *a;
 8      for(a += i+1; i>0; a++)
 9          *b += *a * k[i--];
10      return *b;
11  }
12  int main()
13  {
14      int i=0;
15      do
16          *k = f(k, &i);
17      while(i>0);
18      printf("*k=%d\n",*k);
19  }
```

| Zeile | | | | | |
|---|---|---|---|---|---|
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |





**Aufgabe 4: (10 Punkte)**

Ein Programm besteht aus zwei C-Dateien mit jeweils mehreren Funktionen sowie lokalen und globalen Variablen.

a) Welche Bedeutung hat in diesem Zusammenhang das Schlüsselwort **static**, d.h. was passiert, wenn innerhalb einer Datei globale Variablen als **static** gekennzeichnet sind?

b) Wie können Sie innerhalb einer C-Datei auf die globalen Variablen der jeweils anderen Datei zugreifen?

c) Welche Bedeutung hat das Schlüsselwort **static** bei der Definition lokaler Variablen?

d) Erklären Sie den Unterschied zwischen "eine Variable definieren" und "eine Variable deklarieren".

e) In welcher Weise und durch welche Programme können Sie nun die beiden C-Dateien in ein einziges lauffähiges Programm umwandeln?

**Aufgabe 5: (10 Punkte)**

Sie stehen vor der Aufgabe, ein Programm zur Verwaltung von Musik-CDs zu schreiben. Dabei soll das Programm für beliebig viele Datensätze geeignet sein. Aus diesem Grund entschließen Sie sich zur Verwendung einer verketteten Liste für die CDs.

Um auch mit Doppel- und Sammel-CDs zurecht zu kommen, trennen Sie die Verwaltungsstrukturen in die folgenden **vier** Teilstrukturen auf:

- **CD-Hüllen**, bestehend aus einem Titel, einer Herstellerbezeichnung, der Gesamtspieldauer, einer verketteten Liste tatsächlich darin enthaltener CDs und der Verkettung zur nächsten CD-Hülle.
- **CDs**, bestehend aus einer laufenden Nummer, der Gesamtspieldauer und einer verketteten Liste enthaltener Musikstücke.
- **Musikstücke**, bestehend aus Titel, Interpret und Spieldauer.
- **Spieldauern**, bestehend aus Stunden, Minuten und Sekundenangaben.

Sie können dabei für alle numerische Werte ganze Zahlen verwenden. Die Datenfelder für Namen und Bezeichnungen sollen jeweils die Speicherung von bis zu 40 Textzeichen erlauben!