Friedrich-Alexander-Universität Erlangen-Nürnberg

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

35	**10**	**10**	**10**	**15**	**80**

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

(Fachrichtung) (Semester)

Erlangen, den 17.01.1997

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 (10 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 17.01.1997

(Unterschrift)

Aufgabe 1: (35 Punkte)

Schreiben Sie ein Programm, das innerhalb eines Textes eine Zeichenkette durch eine andere ersetzt. Ihr Programm liest dazu sowohl die zu suchende Zeichenkette wie auch deren Ersatz von der Kommandozeile ein. Anschließend soll Ihr Programm einen beliebig langen Text zeilenweise von der Standardeingabe einlesen und in jeder Zeile suchen und ersetzen. Wird außerdem in der Kommandozeile die Option **-g** angegeben, werden auch mehrere Vorkommen der Zeichenkette innerhalb jeder Zeile ersetzt. Ohne diese Option wird das Durchsuchen einer Zeile nach der ersten Ersetzung abgebrochen und der Rest der Zeile unverändert ausgegeben.

Um innerhalb einer Zeile Wortgrenzen feststellen zu können, dürfen Sie die Funktion **int isspace(int ch)** (deklariert in **ctype.h**) verwenden, die überprüft, ob das übergebene Zeichen ein Whitespace (Leerzeichen, Tabulator usw.) darstellt und entsprechend einen Wert gleich Null oder ungleich Null zurückliefert. Weitere Funktionen, insbesonders aus der Zeichenkettenbibliothek dürfen **nicht** verwendet werden! Zur Vereinfachung dürfen Sie aber davon ausgehen, daß keine Zeile des einzulesenden Textes mehr als 80 Zeichen enthält.

Bearbeiten Sie dazu folgende Teilaufgaben, wobei jeweils etwa die Hälfte der Punkte auf ein zugehöriges Struktogramm entfällt. Dokumentieren Sie Ihr Programm mit kurzen, aber aussagekräftigen Kommentaren!

a) Erstellen Sie eine Funktion

int string_equal(char *s1, char *s2)

Diese Funktion soll zwei Zeichenketten vergleichen und 1 zurückliefern, falls die Zeichenketten gleich sind, sonst 0. **(6 Punkte)**

b) Entwickeln Sie für die eigentliche Suche und Ersetzung eine Funktion:
void search_replace(char *line, char *search, char *replace, int global)
Diese Funktion erhält die zu durchsuchende Zeile (**line**), die zu suchende Zeichenkette(**search**), die Ersatzzeichenkette(**replace**) und die Angabe, ob alle Vorkommen ersetzt werden sollen (**global==1**) oder nur das erste (**global==0**). Der sich ergebende Text soll durch die Funktion auf die Standardausgabe ausgegeben werden. **(15 Punkte)**

c) Entwickeln Sie das Hauptprogramm, in dem die Kommandozeilenparameter ausgewertet werden, der Text eingelesen und die Suche initiiert wird. Greifen Sie dabei auf die Funktionen zurück, die sie unter a) und b) erstellt haben. **(14 Punkte)**

Klausur “Einführung in die Programmierung für Ingenieure (C)” 17 Januar 1997

Aufgabe 2: (10 Punkte)

Was berechnet das folgende Programm **trace**, wenn es als **trace 21 12** aufgerufen wird? Geben Sie die Inhalte der verwendeten Variablen nach jeder relevanten Anweisung an! Verwenden Sie dazu die untenstehende Tabelle!
(Hinweis: `int atio(char *str)` wandelt eine Zeichenkette in einen Integerwert um)

```
#include <stdio.h>
int f(int *a)
{
    int *b = a+1;
    if(!(*a && *b)) return 1;
    for(; *(a+1); b=++a+1)
        *(a+2) = *a % *b;
    return *a;
}
int main(int argc, char *argv[])
{
    int b, a[10];
    if(argc!=3) return 0;
    a[0] = atoi(argv[1]);
    a[1] = atoi(argv[2]);
    b = f(a);
    return b;
}
```


Aufgabe 3: (10 Punkte)

Markieren und verbessern Sie die Fehler (Mehr als 10! Falsch angestrichene Fehler führen zu Punktabzug!) in folgendem C-Programm, das innerhalb eines einzugebenden Textes nach dem Auftreten einer in der Kommandozeile angegebenen Zeichenkette sucht und entweder alle Zeilen, die das Suchmuster enthalten ausgibt oder - bei Option **-v** - alle, die die Zeichenkette nicht enthalten. Bei der Option **-n** erscheinen zusätzlich Zeilennummern in der Ausgabe.

```
include <stdio.h>
include <string.h>
define  OPT_V "-v";
define  OPT_N "-n";

int main(int argc, char argv[])
{
  int opt_n = 0;
  int opt_v = 0;
  int count = 0;
  int ret   = 0;
  int found,i;
  char suchbegriff, zeile[81];
  if (argc<2) {
     printf("usage: %s [-v] [-n] searchpattern\n");
     return;
  }
  suchbegriff = argv[argc];
  for(i=1,i<argc-1,i--)
     if(strcmp(argv[i], OPT_V)==0) opt_v = 1;
     if(strcmp(argv[i], OPT-N)==0) opt_m = 1;
  }
  while(gets(zeile)=!Null) do {
     count++;
     found = strstr(zeile,suchbegriff):1?0;
     if(found && !opt_v) || (!found && opt_v)) {
        if(opt_n)
           printf("%d %c\n", count, zeile)
        else
           printf("%s\n", zeile);
     }
     if(found) ret := 1;
  }
  return ret;
}
```


Aufgabe 4: (10 Punkte)

Gegeben ist das folgende Programmstück:

```
for(i=j;i<100;i++)
{
   if(j > 10) continue;
   j = do_something(j,i);
   if(j < 10) break;
}
```

Setzen Sie dieses Programmfragment so in die nachfolgend geforderten Schleifenkonstrukte um, daß sich die Semantik nicht verändert.

a) Formulieren Sie das Programmstück mit Hilfe einer "while" Schleife.

b) Formulieren Sie das Programmstück mit Hilfe einer "do-while" Schleife.

Aufgabe 5: (15 Punkte)

Für eine Firma wird ein neues Personalverwaltungsprogramm erstellt. Ihre Aufgabe ist es die notwendigen Strukturen zu entwickeln, die für die Verwaltung **beliebig vieler** Personen, Räume und Abteilungen notwendig sind.

Nach gründlicher Analyse der Aufgabenstellung entscheiden Sie, daß Sie unter anderem die folgenden Strukturen benötigen:

- **Abteilungen**, bestehend aus einer Anschrift, einer ganzzahligen Abteilungsnummer, einem Verweis auf den Datensatz des Abteilungsleiters und einer verketteten Liste an Räumen, aus der die Abteilung aufgebaut ist. Da eine unbekannte Anzahl von Abteilungen existieren kann, werden die Abteilungen ebenfalls als verkettete Liste verwaltet.
- **Räume**, bestehend aus einem Verweis auf die Abteilung, zu der sie gehören, einem Raumnamen (15 Zeichen) und einer verketteten Liste aller in diesem Raum arbeitenden Personen. Außerdem sollen sie die Verwaltung von beliebig vielen Räumen in Form einer einfachen, verketteten Liste ermöglichen.
- **Personen**, bestehend aus Vor- und Nachnamen, einem Einstellungsdatum, dem monatlichen Gehalt und der Anschrift der Person. Wie bei den Räumen soll auch hier wieder eine verkettete Liste zur Verwaltung verwendet werden.
- **Anschriften**, bestehend aus Straße, Hausnummer (5 Zeichen), Postleitzahl (5 Zeichen) und Ort.
- **Datumsangaben**, bestehend aus Tag, Monat und Jahr - jeweils als ganze Zahlen.

Soweit nicht anders erwähnt sollen alle Textfelder 40 Zeichen fassen können. Für die numerischen Datenfelder sind passende Datentypen zu wählen!