

Klausur zur Zwischen-Prüfung Prozedurale Programmierung

(IIW / IT / CI)

8. Januar 2010

Sie haben 45 Minuten Zeit zum Bearbeiten der Klausur.

Tragen Sie bitte zunächst Ihren Namen, Ihren Vornamen, Ihre Matrikelnummer, Ihr Studienfach und Ihr Studiensemester in **DRUCKSCHRIFT** in die folgenden jeweils dafür vorgesehenen Felder ein.

Name:															
Vorname:															
Matr.-Nr.:								Fach				Sem.			

Es sind alle Aufgaben zu bearbeiten. Insgesamt können bis zu 30 Punkte erreicht werden.

Aufg.	Punkte	Korr.
1		
2		
3		

Σ	
----------	--

Zur Beachtung:

Schreiben Sie bitte weder mit Bleistift noch mit Rotstift.

Es sind keinerlei Hilfsmittel in dieser Klausur zugelassen!

Vergessen Sie nicht den „Vorbehalt“ zu unterschreiben.

Hinweis zum Programmieren:

Programmieren sie die Aufgaben in ANSI-C

Vorbehalt

Ich bin darüber belehrt worden, dass die von mir zu erbringende Prüfungsleistung nur dann bewertet wird, wenn die Nachprüfung durch das Zentrale Prüfungsamt der TUHH meine offizielle Zulassung vor Beginn der Prüfung ergibt.

(Datum, Unterschrift)

Aufgabe 1 (12 Punkte)

Folgendes Programm soll aus einer beim Programmstart angegebenen CSV-Datei (Comma-Separated Values) ganze Zahlen in ein dynamisches Array einlesen und sie auf dem Bildschirm ausgeben.

Leider sind dem Programmierer dabei 12 Fehler unterlaufen. Helfen Sie ihm, diese Fehler zu finden.

Hinweis: `fscanf` liefert EOF als Rückgabewert, wenn das Dateiende erreicht ist.

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #def N 1000
4
5  void printcsv(FILE *file, int numbers)
6  {
7      int i=0; number;
8
9      if(file == NULL){
10         printf("nichts zum Ausgeben\n");
11         return 0;
12
13         while(fscanf(file,"%d",&number) != EOF && i < N){
14             numbers[i] = number;
15             printf("%d ",number);
16             i++;
17         }
18         if (i >= N) printf("maximale Zeichenanzahl erreicht\n");
19         printf("\ninsgesamt %c Zahlen ausgelesen\n",i);
20     }
21
22     int main(int argc, char argv)
23     {
24         int *numbers;
25         FILE *file;
26
27         if(argc < 2 ) {
28             printf("Fehler: bitte eine CSV-Datei beim Programmaufruf angeben!\n");
29             return -1;
30         }
31         if ((file=fopen(argv[1],"r")) == NULL){
32             printf("Fehler: konnte die Datei %c nicht oeffnen.\n", argv[1]);
33             return -1;
34         }
35         numbers = malloc(N*sizeof(numbers));
36         if (numbers = NULL) {
37             printf("Fehler: konnte benoetigten Speicher nicht reservieren.\n");
38             return -1;
39         }
40         printcsv(file, numbers);
41         fclose(file);
42         return 0
43     }
```

[illegible]

Aufgabe 2 (12 Punkte)

- a) Die sogenannten Legendre-Polynome erhalten ihre Bedeutung durch ihre vielseitige Anwendbarkeit in der Mathematik und in der theoretischen Physik. Für ein $n \in \mathbb{N}$ kann das n -te Legendre-Polynom $L_n(x)$ wie folgt rekursiv definiert werden:

$$L_n(x) = \begin{cases} 1 & \text{falls } n = 0, \\ x & \text{falls } n = 1, \\ \frac{2n-1}{n}xL_{n-1}(x) - \frac{n-1}{n}L_{n-2}(x) & \text{falls } n > 1. \end{cases}$$

Schreiben Sie eine **rekursive** Funktion f , die für einen Double-Wert x und einen Integer-Wert n den Funktionswert des n -ten Legendre-Polynoms an der Stelle x berechnet.

- b) Es gibt auch eine nicht-rekursive Darstellung der Legendre-Polynome:

$$L_n(x) = \sum_{k=0}^{kmax} (-1)^k \frac{(2n-2k)! x^{n-2k}}{(n-k)! (n-2k)! k! 2^n}$$

mit

$$kmax = \begin{cases} \frac{n}{2} & \text{falls } n \text{ gerade} \\ \frac{n-1}{2} & \text{falls } n \text{ ungerade} \end{cases}$$

Schreiben Sie nun eine **nicht-rekursive** Funktion fn , die den Wert $L_n(x)$ berechnet.

Hinweis: zur Berechnung von Fakultäten und Potenzen seien die Funktionen `long fac(int n)` bzw. `double pow(double x, int n)` bereits gegeben.

- c) Erläutern Sie Vor- und Nachteile von rekursiven Funktionen im Vergleich zu nicht rekursiven Funktionen.

Aufgabe 3 (12 Punkte)

- a) Nehmen Sie bei dieser Aufgabe an, dass die Fließkommazahlen entsprechend dem IEEE-754 Standard kodiert werden. (Der IEEE-754 Standard beschreibt die in der Vorlesung und in den Übungsaufgaben behandelte Binärdarstellung der Fließkommazahlen.)

Wandeln Sie die angegebene „binäre“ Fließkommazahl in eine Dezimalzahl um. Gegeben sind:

das **Vorzeichenbit**:

1

der **Exponent:** (mit dem Shift 127)

1	0	0	0	0	0	1	0
---	---	---	---	---	---	---	---

und die **Mantisse**: (vergessen Sie nicht die implizite Eins)

[illegible]

Wert: _____

- b) Warum können folgende Code-Zeilen zu einer Endlosschleife führen? Welche Ausgabe erzeugt folgende Code-Sequenz?

```

1      #define N 100
2      ...

13     int i,carray[N];
14     ...

55     for(i=0;i<=N;i++)
56         carray[i]=0;
57     ...

```

```

10  short int x = 2, y = 0, z = 4;
11  float f1, f2; double d1, d2;
12  while ( x > 0 ) {x = x*x; y += 1;}
13  printf("%d \n",x); /*Ausgabe: _____*/
14  printf("%d \n",y); /*Ausgabe: _____*/
15  x = 100; y = 11; x = x/y/z;
16  printf("%d \n",x); /*Ausgabe: _____*/
17  f1 = d1 = 0.1; d2 = f2 = 0.1;
18  printf("%u \n",d2 != d1);
19                                     /*Ausgabe: _____*/

```

Begründen Sie Ihre Antwort:

Begründen Sie Ihre Antwort:

- c) Stellen Sie in den ersten drei Zeilen die Dezimalzahlen -432 , 234 und -256 im 2-er Komplement (`short int`) binär dar. Die dritte Zeile enthält bereits eine solche Binär-Darstellung. Geben Sie links den entsprechenden Dezimalwert an. Addieren Sie abschließend in der letzten Zeile **alle vier** Zahlen binär.

[illegible][illegible][illegible]

	1	1	1	1	1	1	1	1	1	0	0	1	0	0	1	0
--	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

[illegible]