FernUniversität in Hagen

Fakultät für Wirtschaftswissenschaft

Matrikelnummer _____

Name _____

Vorname _____

Klausur: Entwurf und Implementierung von Informationssystemen (32561)
Termin: 06.03.2020, 09:00–11:00 Uhr
Prüfer: Univ.-Prof. Dr. rer. pol. habil. S. Strecker

Aufbau und Bewertung der Klausur

Aufgabe	1 (OE)	2 (A&D)	3 (PiC)	Summe
Maximal erreichbare Punktzahl	**17**	**40**	**43**	100

Erreichte Punktzahl

Datum:

Note:

Allgemeine Hinweise

Tragen Sie bitte jetzt Ihre **Matrikelnummer**, Ihren **Namen** und **Vornamen** auf dem **Deckblatt** ein. Versehen Sie bitte zusätzlich **jede Seite** mit Ihrer **Matrikelnummer**.

Hinweise zur Bearbeitung

Für die Bearbeitung der insgesamt drei Klausuraufgaben auf den folgenden Seiten (S2–15) dieser Klausur stehen Ihnen 120 Minuten zur Verfügung.

1. Außer Schreib- und Zeichengeräten (z. B. Lineal) sind keine Hilfsmittel zugelassen.
2. Die Lösungen müssen in den vorgesehenen Raum auf den Aufgabenblättern eingetragen werden.
3. Notizen können auf den Rückseiten der Aufgabenblätter gemacht werden. Diese Anmerkungen werden in die Bewertung *nicht* einbezogen.
4. Bei Beendigung der Klausur müssen das Deckblatt und die Aufgabenblätter abgegeben werden. Trennen Sie bitte *nicht* einzelne Blätter ab.

Viel Erfolg!

Aufgabe 1 (Objektorientierter Entwurf) 17P

a. Erläutern Sie in eigenen Worten, was unter dem im Lehrbrief dargestellten Begriff »Framework« zu verstehen ist. Erläutern Sie im Anschluss, welche Vorteile mit Klassenbibliotheken und »Frameworks« einhergehen. Welche Nachteile sind mit der Verwendung verbunden? (7P)

b. Geben Sie an, ob die nachfolgend aufgeführten Aussagen zutreffen oder nicht. Tragen Sie hierzu jeweils in dem vorgegebenen Kreis ein »**R**« für richtig oder ein »**F**« für falsch ein. Für diese Aufgabe gibt es maximal 10 Punkte. Für jede richtige Antwort erhalten Sie 1 Punkt. (10P)

Eine zentrale Aufgabe des objektorientierten Entwurfs einer Anwendung ist die Spezifikation ihrer Architektur. ○

Das Singleton-Muster ist ein objektbasiertes Erzeugungsmuster, welches gewährleistet, dass auf das erzeugte Objekte ausschließlich lokal zugegriffen werden kann. ○

Von einer Wald-Topologie wird gesprochen, wenn eine Klassenbibliothek in mehrere Vererbungshierarchien zerfällt, wobei jede Hierarchie dabei mehrere logisch zusammenhängende Klassen zu einer abhängigen Komponente bündelt. ○

Schnelle Algorithmen basieren auf geeignet gewählten Datenstrukturen, z. B. ermöglichen lineare Listen eine schnellere Suche von Elementen als binäre Suchbäume. ○

Auf Attribute einer Klasse A mit dem Sicherheitsmodus `protected` kann von allen Klassen des Pakets der Klasse A direkt zugegriffen werden. ○

Charakteristisch für die Aktivitäten und Dokumente des objektorientierten Entwurfs ist, dass sie überwiegend im Grobentwurf begonnen und im Feinentwurf vervollständigt werden. ○

Dynamischer Polymorphismus kann die Laufzeit im Vergleich zum statischen Binden in gewissem Umfang erhöhen. ○

Container-Klassen müssen für alle abstrakten Klassen angelegt werden. ○

Eine Assoziation modelliert eine Gesamtheit von direkten Objektverbindungen zwischen zwei Objekten. Assoziationen bilden die Grundlage der Kooperation von Objekten verschiedener Klassen. ○

Bidirektionale Assoziationen sind aufgrund geringerer Kohäsion einfacher zu implementieren und weniger fehleranfällig als unidirektionale Assoziationen. ○

Aufgabe 2 (Algorithmen und Datenstrukturen) 40P

Teil 1: In der gesamten Aufgabe 2 wird von Ihnen erwartet, dass Sie die im Lehrbrief dargestellte, an PASCAL angelehnte Pseudocode-Notation ausnahmslos verwenden. Für das Algorithmieren mit diesem Pseudocode stehen damit die spezifischen Konzepte von PASCAL zur Verfügung, nämlich verschiedene einfache und zusammengesetzte Datentypen, Konstrukte der strukturierten Programmierung und das Prozedurkonzept. Alle Teilaufgaben sind als Codefragmente in der im Lehrbrief dargestellten Pseudocode-Notation zu erstellen. Andere Pseudocode-Notationen oder Programmiersprachen werden *nicht* gewertet.

a. Die rekursive Definition der Fakultät $f(n)$ lautet für alle natürlichen Zahlen $n \geq 0$:

$$f(n) = f(n-1) \cdot n \text{ mit } f(0) = 1 \quad (1)$$

Entwickeln Sie eine Funktion, welche die Fakultät gemäß der rekursiven Definition (s. Gl. 1) berechnet und zurückgibt. (4P)

b. Entwickeln Sie eine weitere Funktion, welche die Fakultät aus Aufgabe 1a ohne das Konzept der Rekursion berechnet und zurückgibt. (4P)

c. Erläutern Sie kurz, was unter einer »Rekursion« zu verstehen ist und welche Vor- bzw. Nachteile mit der Verwendung einer »Rekursion« gegenüber einer Iteration einhergehen. An welche Bedingungen ist ein rekursiver Aufruf einer Prozedur geknüpft? (4P)

Teil 2: Mittels zusätzlicher Zeiger können weitere Verkettungsformen genutzt werden, bspw. Ringverkettung, Doppelverkettung und Ankerkettung. Gegenüber der sequentiellen Speicherung linearer Datenstrukturen weist diese verkettete Speicherung eine Reihe von Vorteilen, aber auch Nachteilen auf. Der Einsatz der verketteten bzw. sequentiellen Speicherung hängt zudem von der jeweiligen Anwendung ab.

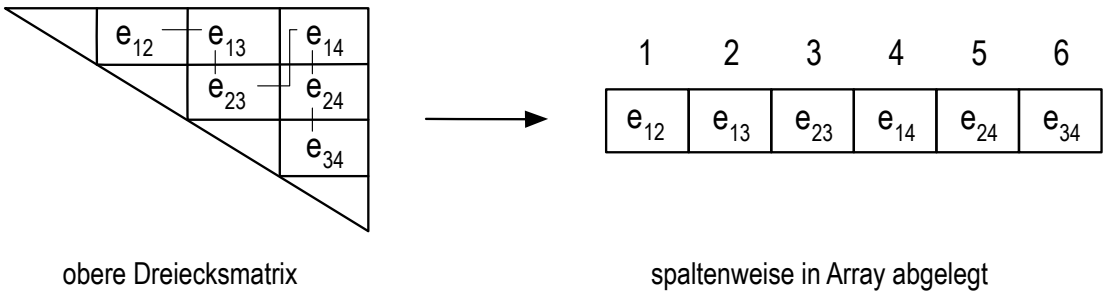
a. Geben Sie im Folgenden die Vor- und Nachteile sowie Maßgaben der Anwendung an. (7P)

Vorteile der verketteten Speicherung:

Nachteile der verketteten Speicherung:

Aspekte der Nutzung einer Anwendung, die die Wahl der Speicherungsform (verkettet oder sequentiell) beeinflussen:

b. Gegeben sei eine obere Dreiecksmatrix der Dimension n, in die die Elemente der Hauptdiagonalen nicht einbezogen sind. Die Matrixelemente e_{ij} mit $i, j \leq n$ und $i < j$, vom Typ **INTEGER** sollen sequentiell und spaltenweise in einem Array abgelegt werden. Die nachstehende schematische Darstellung verdeutlicht den Sachverhalt: (10P)



obere Dreiecksmatrix spaltenweise in Array abgelegt

Entwerfen Sie eine Prozedur, die das in der Zeichnung beschriebene Umspeichern der oberen Dreiecksmatrix (`CONST n = 100;` mit `MATRIX = ARRAY [1..n,1..n] of INTEGER;`) in das eindimensionale Array `MATSEQ` vornimmt. Geben Sie die hierfür nötigen Datendeklarationen für das Array und die Matrix an.

Wie lautet die Adressierungsfunktion für die spaltenweise sequentiell gespeicherte obere Dreiecksmatrix der Dimension n, falls die Elemente der Hauptdiagonalen nicht einbezogen werden?

c. Entwerfen Sie ein Programm für folgenden Sachverhalt: Zu ermitteln ist der Rechnungsendbetrag für mehrere einer Rechnung zugrunde liegende Artikel. Einzulesen sind die Anzahl der Artikel sowie pro Artikel Menge und Einzelpreis. Auszugeben sind Nettoend-, Umsatzsteuer- und Bruttoendbetrag. Zumindest ein Teil der Berechnungen sollte in eine Prozedur ausgelagert werden. Die Umsatzsteuer beträgt neunzehn Prozent. (11P)

Aufgabe 3 (Programmieren in C) 43P

In der gesamten Aufgabe 3 wird von Ihnen erwartet, dass Sie die im Lehrbrief dargestellte Programmiersprache C ausnahmslos verwenden. Für die Implementierung mit dieser Programmiersprache stehen Ihnen damit die spezifischen Sprachkonzepte von C zur Verfügung.

a. Für eine Folge von reellen Messwerten $v_1, v_2, \ldots, v_n$ mit $n \geq 1$, ist der Mittelwert $\overline{m}_n$ gemäß Gl. 2:

$$\overline{m}_n = \frac{1}{n} \cdot \sum_{i=1}^{n} v_i \tag{2}$$

zu berechnen. In dem folgenden Programmfragment finden Sie innerhalb der `main`-Funktion einen Vektor mit 5 Werten, deren Mittelwert zweimal bestimmt werden soll. Zum einen mit einer nicht-rekursiven Funktion `ermittle_mw()` und zum anderen mit einer rekursiven Funktion `ermittle_mw_rek()`. Für beide Funktionen existieren bislang nur die Funktionsköpfe. Ergänzen Sie den Funktionsrumpf für beide Funktionen an den dafür vorgesehenen Stellen. Hinweis: Nutzen Sie bei der Formulierung der rekursiven Berechnung die Beziehung aus Gl. 3: (10P)

$$\overline{m}_n = \frac{1}{n} \cdot (m_{n-1} \cdot (n-1) + v_n) \tag{3}$$

```
#include <stdio.h>
#define LAENGE 5

void main(void){
double werte[LAENGE] = {11.7, 10.5, 11.0, 10.9, 12.2};
double resultat, resultat_rek;
int laenge = 5;

resultat = ermittle_mw(werte, laenge-1);
resultat_rek = ermittle_mw_rek(werte, laenge-1);
printf("1: .2Lf 2: .2Lf", resultat, resultat_rek);

double ermittle_mw(double *v, int pos){
```

```

```

```
}
```

```
double ermittle_mw_rek(double *v, int pos){
```

```

```

```
}
```

Hinweis: `*v` ist ein Zeiger auf das 1. Vektorelement mit dem Index 0. Die Variable `pos` ist der Index des letzten Vektorelements.

b. Entwerfen Sie ein Funktion, welche einen beispielhaften Vektor von integer-Werten (Ausgangsfolge), mit dem Sortierverfahren *Sortieren durch Austauschen* (engl. »bubble sort«) sortiert. Der eingegebene Vektor (Ihrer Wahl bzw. durch Übergabe) soll durch paarweise miteinander vergleichen sowie Austauschen der Vektoreinträge, die Zielreihenfolge aufsteigend sortieren. Entwickeln Sie zunächst eine Funktion zum Tauschen zweier integer-Werte (`void swap(...)`): (8P)

```
static void swap(int *px, int *py){
```

```

```

```
}
```

```
void bubble(int *vector, int length){
```

```
}
```

c. Entwerfen Sie ein C-Programm, welches folgende Anforderungen berücksichtigt:

- Das C-Programm erfragt vom Benutzer die Anzahl N Studenten, zu welchen im nächsten Schritt Daten eingegeben werden sollen.
- Als nächstes werden N Datensätze bestehend aus Name und Note eingelesen.
- Das Programm gibt die Noten aller Studenten aus, welche die beste Note erhalten haben (es können mehrere Studenten zugleich die besten sein). Die beste Note ist nicht notwendigerweise 1.0, sondern die beste erreichte Note.
- Das Programm berechnet schließlich die Durchschnittsnote aller Studenten und gibt die Durchschnittsnote aus.
- Beachten Sie, dass Sie selbst Speicher auf dem Heap bereitstellen müssen.
- Überprüfen Sie, ob die Speicherallokation funktioniert hat.
- Überprüfen Sie außerdem, ob die eingegebene Note gültig ist (Note zwischen 1.0 und 5.0).

Gegeben ist in einem C-Programm folgende Struktur, auf die Sie zurückgreifen sollen: (10P)

```
#include <stdio.h>
#include <stdlib.h>

typedef struct
{
 char nachname[30];
 float note;
} student;
```

d. Gegeben ist ein C-Programm mit einigen Ausdrücken. Markieren Sie mit einem »**X**«, welche der folgenden Ausdrücke compiliert oder die Compilation des Programms mit einer Fehlermeldung verhindern. (5P)

```
int main(void){
int i;
char c = 'c';
```

/* Programmcode */	/* compiliert */	/* compiliert nicht */
`int *****ip;`	◯	◯
`i = 'a' + c`	◯	◯
`c = (i==1);`	◯	◯
`i++ = c;`	◯	◯
`while (i--) i++`	◯	◯

```
}
```

e. Entwerfen Sie ein Programm, das die Bücher einer Buchhandlung verwaltet. Dazu benötigen Sie eine Typenvereinbarung bzw. Struktur (`book`), die Inventarnummer, einen Titel (maximal 30 Zeichen), einen Preis, den Umsatzsteuersatz und eine Stückzahl für bis zu 100 Bücher. Das entworfene C-Programm liest als erstes den Umsatzsteuersatz ein. Anschließend sollen innerhalb einer `for`-Schleife die restlichen Daten aller Bücher eingelesen und der Preis inkl. der Umsatzsteuern berechnet werden. (10P)