FernUniversität in Hagen

Fakultät für Wirtschaftswissenschaft

Matrikelnummer _____

Name _____

Vorname _____

Klausur: Entwurf und Implementierung von Informationssystemen (32561)
Termin: 06.09.2019, 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	**18**	**45**	**37**	100

Erreichte Punktzahl

Datum:

Note:

Copyright © 2019 FernUniversität in Hagen - Alle Rechte vorbehalten

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 (S2–16) 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!

Copyright © 2019 FernUniversität in Hagen - Alle Rechte vorbehalten

Aufgabe 1 (Objektorientierter Entwurf) 18P

a. Für Klassenbibliotheken sind verschiedene Arten der Strukturierung gebräuchlich, die als Topologien bezeichnet werden. Erläutern Sie im Folgenden ausführlich, welche Topologien von Klassenbibliotheken im Lehrbrief eingeführt werden. Anschließend sollen Sie die Topologien schematisch skizzieren (grafisch darstellen) und die spezifischen Vorteile der jeweiligen Topologie diskutieren. Um die spezifischen Vorteile der jeweiligen Topologie darzustellen, bietet es sich an, diese gegenüberzustellen und zu diskutieren. (8P)

Diese Klausur ist urheberrechtlich geschützt. Die dadurch begründeten Rechte, insbesondere das Recht der Vervielfältigung und Verbreitung sowie der Übersetzung und des Nachdrucks, bleiben, auch bei nur auszugsweiser Verwertung, vorbehalten. Kein Teil des Werkes darf in irgendeiner Form (Druck, Fotokopie, Mikrofilm oder ein anderes Verfahren) ohne schriftliche Genehmigung der FernUniversität reproduziert oder unter Verwendung elektronischer Systeme verarbeitet, vervielfältigt oder verbreitet werden.
Copyright © 2019 FernUniversität in Hagen - Alle Rechte vorbehalten

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 verteilte Anwendung gliedert sich in mehrere selbstständig lauffähigen Programme, die auf verschiedenen, räumlich getrennten Informatiksystemen gleichzeitig ausgeführt werden. ○

Ein wesentlicher Vorzug des objektorientierten Paradigmas der Softwareentwicklung gegenüber dem strukturierten Paradigma besteht in der Umsetzung der Datenkapselung. ○

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

Die Architektur einer Anwendung beschreibt ihre Struktur, d. h. ihre Gliederung in Komponenten und deren Beziehungen. ○

Die detaillierte Spezifikation einer Assoziation im objektorientierten Entwurf bedeutet die vollständige Modellierung im Klassendiagramm, wozu u. a. die Festlegung der Kardinalitäten und der Navigation gehören. ○

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

Eine Assoziation modelliert die direkte Objektverbindung zwischen genau zwei Objekten. Assoziationen bilden die Grundlage der Kooperation von Objekten verschiedener Klassen. ○

Charakteristisch für Schichten-Architekturen ist unter anderem, dass zwischen den Schichten einer Anwendung unidirektionale Abhängigkeitsbeziehungen bestehen. ○

Enthält eine Klasse mindestens eine abstrakte Operation, so können von dieser Klasse Objekte erzeugt werden. ○

Von einer Wald-Topologie wird gesprochen, wenn eine Klassenbibliothek aus lauter unabhängigen Klassen besteht, die nicht durch Vererbungsbeziehungen verbunden sind. ○

Diese Klausur ist urheberrechtlich geschützt. Die dadurch begründeten Rechte, insbesondere das Recht der Vervielfältigung und Verbreitung sowie der Übersetzung und des Nachdrucks, bleiben, auch bei nur auszugsweiser Verwertung, vorbehalten. Kein Teil des Werkes darf in irgendeiner Form (Druck, Fotokopie, Mikrofilm oder ein anderes Verfahren) ohne schriftliche Genehmigung der FernUniversität reproduziert oder unter Verwendung elektronischer Systeme verarbeitet, vervielfältigt oder verbreitet werden.
Copyright © 2019 FernUniversität in Hagen - Alle Rechte vorbehalten

Aufgabe 2 (Algorithmen und Datenstrukturen) 45P

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.

Die **Eulersche Zahl** e ist eine Konstante, die in der Differential- und Integralrechnung von zentraler Bedeutung ist. Bei dieser Zahl handelt es sich um eine transzendente und somit ebenfalls irrationale reele Zahl. Es gibt zahlreiche »äquivalente« Definitionen von e, eine davon lautet:

$$e = \sum_{k=0}^{\infty} \frac{1}{k!} = 1 + \frac{1}{1} + \frac{1}{1 \cdot 2} + \frac{1}{1 \cdot 2 \cdot 3} + \frac{1}{1 \cdot 2 \cdot 3 \cdot 4} + \dots$$

a. Entwerfen Sie eine Funktion, um in Abhängigkeit einer eingegebenen positiven natürlichen Zahl n nährungsweise den Wert der Eulerschen Zahl e *rekursiv* zu berechnen. Dabei sei n mit $\{n \in \mathbb{N} : n \geq 0\}$ die Laufweite der Summe. Hinweis: Erstellen Sie zwei Funktionen: (1) Zur Berechnung der Fakultät und (2) zur nährungsweisen Bestimmung des Wertes. Beide Algorithmen sind *rekursiv* zu berechnen und zu implementieren. (6P)

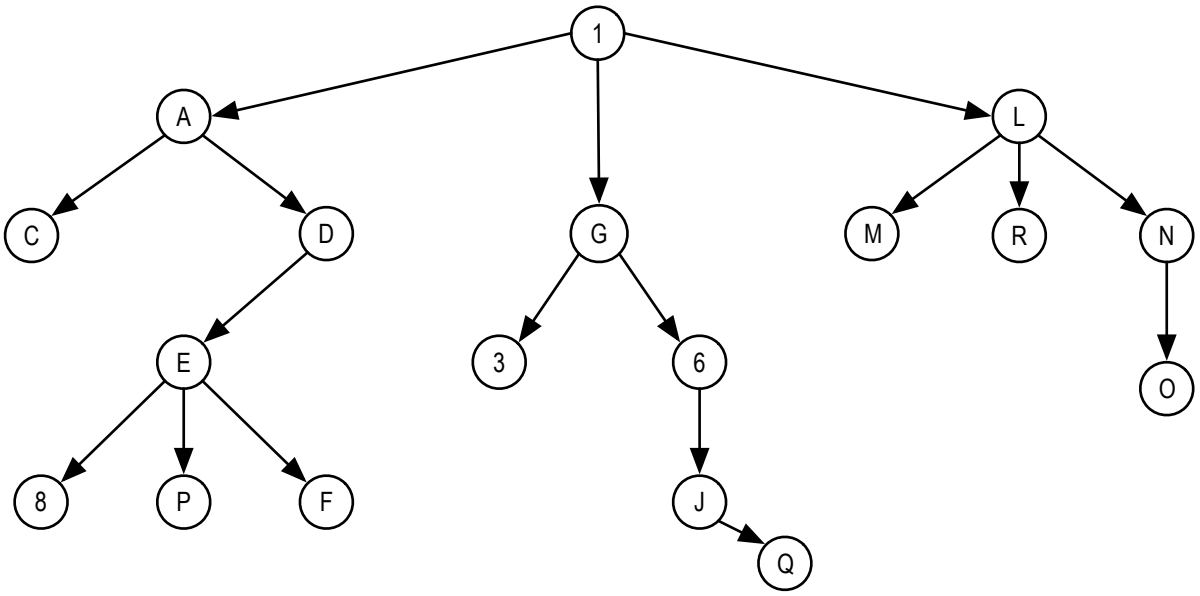
Diese Klausur ist urheberrechtlich geschützt. Die dadurch begründeten Rechte, insbesondere das Recht der Vervielfältigung und Verbreitung sowie der Übersetzung und des Nachdrucks, bleiben, auch bei nur auszugsweiser Verwertung, vorbehalten. Kein Teil des Werkes darf in irgendeiner Form (Druck, Fotokopie, Mikrofilm oder ein anderes Verfahren) ohne schriftliche Genehmigung der FernUniversität reproduziert oder unter Verwendung elektronischer Systeme verarbeitet, vervielfältigt oder verbreitet werden.
Copyright © 2019 FernUniversität in Hagen - Alle Rechte vorbehalten

b. Erläutern Sie kurz, was unter dem »Komplexitätsbegriff« im Kontext von Algorithmen zu verstehen ist. Was sind Zwecke und Ziele der Komplexitätsbestimmung? Erörtern und zeigen Sie ausführlich anhand der Bedingungen, wie sich die Komplexität laut der im Lehrbrief eingeführten $\mathcal{O}$-Notation bestimmen lässt. Geben Sie das Wachstumsverhalten ausgewählter Standardfunktionen an: (1) Sequentielle Suche, (2) Bubblesort (»Sortieren durch Austauschen«) und (3) Heapsort (»Haldensortierung«). (8P)

Diese Klausur ist urheberrechtlich geschützt. Die dadurch begründeten Rechte, insbesondere das Recht der Vervielfältigung und Verbreitung sowie der Übersetzung und des Nachdrucks, bleiben, auch bei nur auszugsweiser Verwertung, vorbehalten. Kein Teil des Werkes darf in irgendeiner Form (Druck, Fotokopie, Mikrofilm oder ein anderes Verfahren) ohne schriftliche Genehmigung der FernUniversität reproduziert oder unter Verwendung elektronischer Systeme verarbeitet, vervielfältigt oder verbreitet werden.
Copyright © 2019 FernUniversität in Hagen - Alle Rechte vorbehalten

2) Mit dem Begriff »Traversieren« wird das Durchlaufen sämtlicher Knoten eines Baumes in einer bestimmten Reihenfolge bezeichnet. In der Regel wird mit dem Traversieren die Bearbeitung vieler oder aller Knoten bzw. Datenobjekte verbunden sein. Für Binärbäume eignen sich insbes. auch rekursive Traversierungsalgorithmen.

a. Wandeln Sie den nachfolgend dargestellten k-nären Baum, mit einer Ordnung von $k = 3$, in einen Binarbaum um. (8P)



Diese Klausur ist urheberrechtlich geschützt. Die dadurch begründeten Rechte, insbesondere das Recht der Vervielfältigung und Verbreitung sowie der Übersetzung und des Nachdrucks, bleiben, auch bei nur auszugsweiser Verwertung, vorbehalten. Kein Teil des Werkes darf in irgendeiner Form (Druck, Fotokopie, Mikrofilm oder ein anderes Verfahren) ohne schriftliche Genehmigung der FernUniversität reproduziert oder unter Verwendung elektronischer Systeme verarbeitet, vervielfältigt oder verbreitet werden.
Copyright © 2019 FernUniversität in Hagen - Alle Rechte vorbehalten

Diese Klausur ist urheberrechtlich geschützt. Die dadurch begründeten Rechte, insbesondere das Recht der Vervielfältigung und Verbreitung sowie der Übersetzung und des Nachdrucks, bleiben, auch bei nur auszugsweiser Verwertung, vorbehalten. Kein Teil des Werkes darf in irgendeiner Form (Druck, Fotokopie, Mikrofilm oder ein anderes Verfahren) ohne schriftliche Genehmigung der FernUniversität reproduziert oder unter Verwendung elektronischer Systeme verarbeitet, vervielfältigt oder verbreitet werden.
Copyright © 2019 FernUniversität in Hagen - Alle Rechte vorbehalten

3) Betrachtet sei ein Sortieralgorithmus für sequentiell gespeicherte Objekte. Die Tabelle unten zeigt die Schritte eines solchen Algorithmus sowie die Ausgangs- und Zielfolge.

Ausgangsfolge	:	42	51	13	48	92	16	09	68
1. Schritt	:	42	13	48	51	16	09	68	92
2. Schritt	:	13	42	48	16	09	51	68	92
3. Schritt	:	13	42	16	09	48	51	68	92
4. Schritt	:	13	16	09	42	48	51	68	92
5. Schritt	:	13	09	16	42	48	51	68	92
6. Schritt	:	09	13	16	42	48	51	68	92
Zielfolge	:	09	13	16	42	48	51	68	92

a. Benennen Sie den Algorithmus und erläutern Sie stichpunktartig den Sortiervorgang. (3P)

Diese Klausur ist urheberrechtlich geschützt. Die dadurch begründeten Rechte, insbesondere das Recht der Vervielfältigung und Verbreitung sowie der Übersetzung und des Nachdrucks, bleiben, auch bei nur auszugsweiser Verwertung, vorbehalten. Kein Teil des Werkes darf in irgendeiner Form (Druck, Fotokopie, Mikrofilm oder ein anderes Verfahren) ohne schriftliche Genehmigung der FernUniversität reproduziert oder unter Verwendung elektronischer Systeme verarbeitet, vervielfältigt oder verbreitet werden.
Copyright © 2019 FernUniversität in Hagen - Alle Rechte vorbehalten

b. Entwickeln Sie für den dargestellten Sortieralgorithmus eine Prozedur. Folgende Datendefinition für ein zu sortierendes Feld steht Ihnen zur Verfügung: (7P)

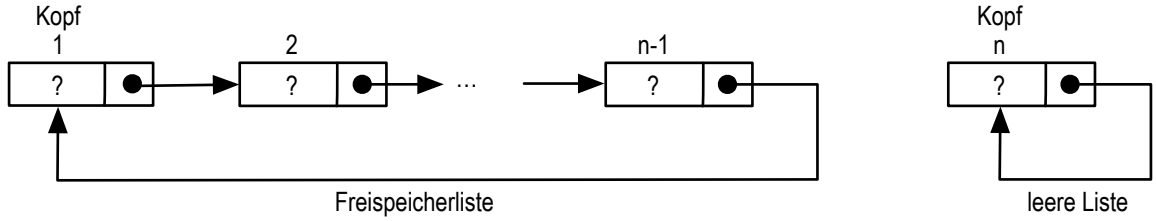
```
DATA
 CONST n = 8;
 TYPE INDEX = [1..n];
  FELD = ARRAY [1..n] OF INTEGER;
 VARIABLE feld :FELD;

PROCEDURE sortalgorithmus(VARIABLE feld :FELD);
DATA
 VARIABLE i,j :INDEX;
          element :INTEGER;
BEGIN
```

```
END sortalgorithmus;
```

Diese Klausur ist urheberrechtlich geschützt. Die dadurch begründeten Rechte, insbesondere das Recht der Vervielfältigung und Verbreitung sowie der Übersetzung und des Nachdrucks, bleiben, auch bei nur auszugsweiser Verwertung, vorbehalten. Kein Teil des Werkes darf in irgendeiner Form (Druck, Fotokopie, Mikrofilm oder ein anderes Verfahren) ohne schriftliche Genehmigung der FernUniversität reproduziert oder unter Verwendung elektronischer Systeme verarbeitet, vervielfältigt oder verbreitet werden.
Copyright © 2019 FernUniversität in Hagen - Alle Rechte vorbehalten

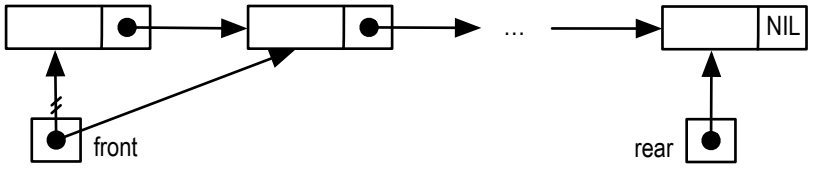
c. Sowohl Nutzdatenliste als auch die Freispeicherliste enthalten ein Kopfelement. Die Nutzdatenliste umfasst alle durch Listenelemente belegten Bereiche eines Adressraums. Eine leere Freispeicherliste besteht nur aus diesem Element. In einer auf dieser Freispeicherliste basierenden Nutzdatenliste können also maximal $n-1$ Elemente eingefügt werden. Das Kopfelement der Freispeicherliste besitzt stets den Index 1 und das Kopfelement der Nutzdatenliste stets den Index n. Diese Werte ermöglichen jederzeit den Einstieg in die Listen. Formulieren Sie das beschriebene Vorgehen als Initialisierungsoperation in einer Prozedur aus. Die Initialisierung der Listenelemente soll mit 0 erfolgen. Das letzte Listenelement soll mit 1 initialisiert werden. Beachten Sie die besondere Funktion der Kopfelemente. (5P)




```
PROCEDURE init (VARIABLE link:ZEIGERFELD);
DATA
 ZEIGER = [1..n];
 ZEIGERFELD:ARRAY [1..n] OF ZEIGER;
```

```
END init;
```

d. Das Entnehmen eines Elements veranschaulicht die folgende schematische Darstellung. Formulieren Sie das Entnehmen eines Elements als Prozedur `remove` aus. Beachten Sie, dass ein Entnehmen nur bei nicht leerer Schlange möglich ist. Folgende Datenvereinbarung steht Ihnen zur Verfügung: (8P)



Diese Klausur ist urheberrechtlich geschützt. Die dadurch begründeten Rechte, insbesondere das Recht der Vervielfältigung und Verbreitung sowie der Übersetzung und des Nachdrucks, bleiben, auch bei nur auszugsweiser Verwertung, vorbehalten. Kein Teil des Werkes darf in irgendeiner Form (Druck, Fotokopie, Mikrofilm oder ein anderes Verfahren) ohne schriftliche Genehmigung der FernUniversität reproduziert oder unter Verwendung elektronischer Systeme verarbeitet, vervielfältigt oder verbreitet werden.
Copyright © 2019 FernUniversität in Hagen - Alle Rechte vorbehalten

```
DATA
 CONST n = 200;
 TYPE
  ORDER = RECORD
          orderno:ARRAY [1..8] OF CHAR;
          company:ARRAY [1..20] OF CHAR;
          processor:ARRAY [1..6] OF CHAR;
          END;
  ORDERLINK = ↑ORDERELEM;
  ORDERELEM = RECORD;
              order:ORDER;
              next:ORDERLINK;
              END;

 VARIABLE
  conorder:ORDER;
  front, rear:ORDERLINK;
  entries:INTEGER;
  full, empty:BOOLEAN;

PROCEDURE remove (VARIABLE front, rear:ORDERLINK;
                  VARIABLE empty:BOOLEAN;
                  VARIABLE entries:INTEGER;
                  VARIABLE conorder:ORDER);
```

Diese Klausur ist urheberrechtlich geschützt. Die dadurch begründeten Rechte, insbesondere das Recht der Vervielfältigung und Verbreitung sowie der Übersetzung und des Nachdrucks, bleiben, auch bei nur auszugsweiser Verwertung, vorbehalten. Kein Teil des Werkes darf in irgendeiner Form (Druck, Fotokopie, Mikrofilm oder ein anderes Verfahren) ohne schriftliche Genehmigung der FernUniversität reproduziert oder unter Verwendung elektronischer Systeme verarbeitet, vervielfältigt oder verbreitet werden.
Copyright © 2019 FernUniversität in Hagen - Alle Rechte vorbehalten

Aufgabe 3 (Programmieren in C) 37P

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.

1) Die FernUniversität in Hagen besitzt eine Vielzahl an Dienstfahrzeugen, die aktuell noch durch eine Sachbearbeiterin manuell erfasst werden, d. h. diese Sachbearbeiterin führt einen Aktenordner mit allen Daten.

a. Sie werden als externer IT-Dienstleister beauftragt, ein Informationssystem zur Verwaltung von Dienstfahrzeugen zu entwerfen und zu implementieren. Dafür soll eine Datenvereinbarung namens `car` und einem dazugehörigen Typ `car_t` implementiert werden. Folgende Eigenschaften sind für die Erfassung zu berücksichtigen: Marke (VW, BMW oder PORSCHE), Maximalgeschwindigkeit, Antiblockiersystem (ABS, enthalten/nicht enthalten) und Türenanzahl. Entscheiden Sie selbst, welche Datentypen für diese Eigenschaften sinnvoll sind. Erstellen Sie im ersten Schritt eine Funktion, welche ein Dienstfahrzeug (`car_t`) liefert, dessen Eigenschaften sie per Parameter übergeben bekommt. Sowohl die Marke des Dienstfahrzeuges als auch die Datenvereinbarung für das jeweilige Dienstfahrzeug sind als Datenstrukturen zu implementieren. (8P)

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

Diese Klausur ist urheberrechtlich geschützt. Die dadurch begründeten Rechte, insbesondere das Recht der Vervielfältigung und Verbreitung sowie der Übersetzung und des Nachdrucks, bleiben, auch bei nur auszugsweiser Verwertung, vorbehalten. Kein Teil des Werkes darf in irgendeiner Form (Druck, Fotokopie, Mikrofilm oder ein anderes Verfahren) ohne schriftliche Genehmigung der FernUniversität reproduziert oder unter Verwendung elektronischer Systeme verarbeitet, vervielfältigt oder verbreitet werden.
Copyright © 2019 FernUniversität in Hagen - Alle Rechte vorbehalten

b. Nun soll ebenfalls der Beschaffungsbetrag eines Dienstfahrzeuges festgehalten werden. Unglücklicherweise sind die Rechnungen bei dem Versuch, diese zu digitalisieren, abhanden gekommen, so dass der Anschaffungspreis nun geschätzt werden muss. Entwickeln Sie dafür eine Funktion, die ein Auto enthält und den Preis folgendermaßen schätzt: Die Maximalgeschwindigkeit muss mit 50 multipliziert werden. Anschließend wird das Ergebnis mit der Türenanzahl multipliziert. Wenn das Auto ein Antiblockiersystem hat, muss ein Betrag in Höhe von 5000 dazugerechnet werden. Handelt es sich um einen BMW, wird der bisherige Gesamtbetrag mit 1.25 multipliziert, bei einem Porsche wird der Betrag mit 2 multipliziert. (5P)

c. Abschließend soll der Gesamtpreis für drei Dienstfahrzeuge berechnet werden. Entwickeln Sie dazu eine Funktion für die folgenden Dienstfahrzeuge: (i) VW, 190 km/h, mit ABS, 5 Türen; (ii) BMW, 210 km/h, mit ABS, 3 Türen; (iii) PORSCHE, 260 km/h, ohne ABS, 4 Türen. Speichern Sie die Fahrzeuge in einem Array und berechnen Sie in einer Schleife den Gesamtpreis aller Autos und geben diesen in der Standardausgabe (z. B. Konsole) aus. (5P)

```
int main()
{
```

```
}
```

Diese Klausur ist urheberrechtlich geschützt. Die dadurch begründeten Rechte, insbesondere das Recht der Vervielfältigung und Verbreitung sowie der Übersetzung und des Nachdrucks, bleiben, auch bei nur auszugsweiser Verwertung, vorbehalten. Kein Teil des Werkes darf in irgendeiner Form (Druck, Fotokopie, Mikrofilm oder ein anderes Verfahren) ohne schriftliche Genehmigung der FernUniversität reproduziert oder unter Verwendung elektronischer Systeme verarbeitet, vervielfältigt oder verbreitet werden.
Copyright © 2019 FernUniversität in Hagen - Alle Rechte vorbehalten

d. Welche Ausgabe liefert das folgende Programm? (4P)

```
#include <stdio.h>

struct person{
 char name[30];
 long int knr;
};

int main()
{
static struct person kunde[4] = {"Schmitz", 20123, "Mueller", 82765, "
    Adams", 98761};

printf("%s",kunde[0].name);
printf("%c",kunde[2].name[0]);
printf("%li",kunde[1].knr);
printf("%li",kunde[3].knr);
}
```

e. Welchen Wert haben die Variablen a, i und j nach der Zuweisung? (3P)

```
#include <stdio.h>

struct person{
 char name[30];
 long int knr;
};

int main(){

int i, j;
float a = 1.5, b = 2.5;

i = (int) a;
j = (int) (a * b);
a = (float) (i + j);
}
```

Diese Klausur ist urheberrechtlich geschützt. Die dadurch begründeten Rechte, insbesondere das Recht der Vervielfältigung und Verbreitung sowie der Übersetzung und des Nachdrucks, bleiben, auch bei nur auszugsweiser Verwertung, vorbehalten. Kein Teil des Werkes darf in irgendeiner Form (Druck, Fotokopie, Mikrofilm oder ein anderes Verfahren) ohne schriftliche Genehmigung der FernUniversität reproduziert oder unter Verwendung elektronischer Systeme verarbeitet, vervielfältigt oder verbreitet werden.
Copyright © 2019 FernUniversität in Hagen - Alle Rechte vorbehalten

2) Der gesamte, durch ein Programm zur Laufzeit belegbare Speicherplatz kann in einzelne Speicherbereiche gegliedert werden, in denen jeweils verschiedenartige Informationen abgelegt werden. Zur dynamischen Reservierung von Speicherplatz aus dem Heap werden Funktionen der C-Standardbibliothek eingesetzt, die meist in der Definitionsdatei `stdlib.h` vereinbart sind.

a. Erläutern Sie die Konzepte `malloc`, `calloc` und `free`, die im Zuge der dynamischen Speicherreservierung Anwendung finden. Erläutern Sie zusätzlich mögliche Zusammenhänge zwischen diesen Funktionen. Die Beschreibung sollte u. a. Erläuterungen zu Art und Ort der Speicherreservierung, zu Rückgabewerten und zum Vorgehen bei Fehlern enthalten. (6P)

Diese Klausur ist urheberrechtlich geschützt. Die dadurch begründeten Rechte, insbesondere das Recht der Vervielfältigung und Verbreitung sowie der Übersetzung und des Nachdrucks, bleiben, auch bei nur auszugsweiser Verwertung, vorbehalten. Kein Teil des Werkes darf in irgendeiner Form (Druck, Fotokopie, Mikrofilm oder ein anderes Verfahren) ohne schriftliche Genehmigung der FernUniversität reproduziert oder unter Verwendung elektronischer Systeme verarbeitet, vervielfältigt oder verbreitet werden.
Copyright © 2019 FernUniversität in Hagen - Alle Rechte vorbehalten

b. Entwerfen Sie eine Funktion, die eine natürliche Zahl n übergeben bekommt und dann die Summe aller Quadratzahlen bis zu der eingegebenen Zahl ausgibt. Die Berechnung kann nur für $n > 0$ erfolgen; bei falscher Übergabe wird die Funktion verlassen. Das entwickelte C-Programm ruft die Funktion `quadsum(int n)` auf, die diese Summe rekursiv ermittelt und über die Standardausgabe ausgibt, wobei zusätzlich noch die Quadratzahl ausgegeben werden soll. (6P)

```
int quadsum(int n)
{
```

```
}
```

Diese Klausur ist urheberrechtlich geschützt. Die dadurch begründeten Rechte, insbesondere das Recht der Vervielfältigung und Verbreitung sowie der Übersetzung und des Nachdrucks, bleiben, auch bei nur auszugsweiser Verwertung, vorbehalten. Kein Teil des Werkes darf in irgendeiner Form (Druck, Fotokopie, Mikrofilm oder ein anderes Verfahren) ohne schriftliche Genehmigung der FernUniversität reproduziert oder unter Verwendung elektronischer Systeme verarbeitet, vervielfältigt oder verbreitet werden.
Copyright © 2019 FernUniversität in Hagen - Alle Rechte vorbehalten