Praktikum
Praktische Informatik II

Prof. Dr.-Ing. D. Rosenthal

M. Sc. D. Schröder-Spahić

M. Sc. A. Dizdarević

zugeteile Nummern: _____

Aufgabenblatt: 1

Vom: 12.3.2019

Abgabedatum: -

Name, Vorname, Matrikelnummer _____

Name, Vorname, Matrikelnummer _____

Korrekturhinweise
(Bitte nichts eintragen!)

	X1	X2	X3
Aufgabe			
Aufgabe			
Aufgabe			

X1 = Aufgabenstellung
X2 = Lauffähigkeit
X3 = Programmier-Richtlinien

Korrektur:
Ja ____
Nein ____

Sonstige Bemerkungen:

Technology
Arts Sciences
TH Köln

Prof. Dr.-Ing. Dieter Rosenthal
M.Sc. Dženita Schröder-Spahić
M.Sc. Alen Dizdarević
Praktikum „Praktische Informatik II“
TH Köln, Fakultät 07
12.3.2019

C

Aufgabenblatt 1

Themen:
Strukturen, Zeichenketten, dynamische Speicherverwaltung, Arrays

Vorbemerkung:
Beachten Sie bei der Bearbeitung der Aufgaben die unten aufgeführte Vorgehensweise:

1. Erstellen des Programmablaufes als Struktogramm, falls verlangt.
2. Erstellen des C-Codes aus dem Struktogramm, falls verlangt.
3. Schreiben ausführlicher **Kommentare** zum geschriebenen C-Code.

4. Der Bericht ist online unter "Magazin->Fakultät 07->Dozenten->Rosenthal->Praktische Informatik II->Praktika" in Ihrer passenden Praktikumsgruppe unter Übung "Abgabe Versuch X" hochzuladen.
   Dort können Sie bis zum angegebenen Zeitpunkt **eine ZIP-Datei** mit allen Unterlagen hochladen. Wollen Sie eine neuere Version online stellen, können Sie die alte löschen und eine neue hochladen. Achten Sie dabei auf die Abgabefrist!

5. In der ZIP-Datei müssen alle eigenständigen Programme als einzelnes **Code::Blocks-Projekt** in einzelnen Ordner hinterlegt werden. Des Weiteren muss das Deckblatt mit **Namen, Vorname, Matrikelnummer** und **zugeteilten Nummern** ausgefüllt in der ZIP-Datei vorhanden sein.
   Abzugebende Struktogramme sind als **jpg-Datei** hinzuzufügen.
   Alle nicht benötigten Dateien werden nicht mit hochgeladen.

6. Es wird nur eine Datei im **ZIP-Format** ohne Passwort angenommen, welche alle weiteren Dateien beinhaltet. Programme müssen als **Code::Blocks-Projekte** und nicht als einzelne C-Dateien abgegeben werden. Schriftliche Ausarbeitungen dürfen nur im **pdf-Format** hochgeladen werden. Formate nach DOC, DOCX, ODT werden nicht berücksichtigt und automatisch gelöscht.
   Je nach Aufgabenstellung kann die Abgabe weitere Teile beinhalten (z.B. schriftlich zu beantwortenden Fragen oder schriftliche Berechnungen).
   WICHTIG: Bei Nichteinhaltung gilt die komplette Abgabe als **nicht** erfolgt.

7. **Jeder** gibt einen Bericht ab. Der Bericht ist **vor** Ablauf der Abgabefrist (s.o. Titelblatt) online abzugeben. Bei unterschiedlichen Abgaben einer Gruppe, gilt die komplette Abgabe als **nicht** erfolgt

Ausschlusskriterium für die Teilnahme am Praktikumsversuch:

Das **lauffähige** und **kompilierbare** C-Programm ist vor Ort aus Ihrer Ilias-Abgabe herunterzuladen! Die Kompilierbarkeit und Lauffähigkeit wird dort auf den Rechner kontrolliert. Des Weiteren erhalten Sie eine persönliche Aufgabe, die Sie während des Praktikums lösen und danach ebenfalls abgeben müssen.

Technology
Arts Sciences
TH Köln

Prof. Dr.-Ing. Dieter Rosenthal
M.Sc. Dženita Schröder-Spahić
M.Sc. Alen Dizdarević
Praktikum „Praktische Informatik II“
TH Köln, Fakultät 07
12.3.2019

C

Aufgabenblatt 1

Aufgabe 1.1: Zeile als Zeichenkette einlesen

Abgabe: 27.03.19

Zum Einlesen von Zeichenketten gibt es einige Funktionen wie z.B fgets(), getch() und scanf(). Bei diesen Funktionen gibt es jedoch die Probleme, dass entweder nur einzelne Zeichen oder Zeichenketten mit einer festen Länge eingelesen werden können. Wird die Funktion nicht korrekt angewendet, kann es dazu kommen, dass Zeichenketten abgeschnitten werden oder es zu Speicherüberläufen kommt. Außerdem hat scanf() das Problem, dass sie nur bis zum nächsten Leerzeichen einließt.

Um dies zu beheben soll eine Funktion mit folgendem Prototyp implementiert werden:

char* zeichenketteEinlesen(void)

- Ein einzelnes Zeichen vom standard-input-stream (**stdin**) einlesen
- Neuen benötigten dynamischen Speicherplatz anfordern, damit die alte Zeichenkette und das neue Zeichen abgespeichert werden kann.
- Alte Zeichenkette und neu eingelesenes Zeichen in neuen Speicher kopieren.
- Dynamischen Speicherplatz der alten Zeichenkette wieder freigeben.
- Wiederholen der vorherigen Punkte bis zum Einlesen des String-Ende-Zeichen '\0'
- Eingelesene Zeichenkette zurückgeben

Hinweis:
- Achten Sie auf das '\0'-Zeichen am Ende jeder Zeichenkette, die zurück gegeben wird.
- Mit der Funktion **realloc()** kann die oben angegebene Vorgehensweise verkürzt werden.

Erstellen Sie eine **main()** in der Sie 3 Pointer zum Speichern von Zeichenketten anlegen. Rufen Sie die oben angegebene Funktion **char* zeichenketteEinlesen(void)** 3 mal auf und speichern die zurückgegebenen Adressen in jeweils einen der Pointer ab. Nachdem Sie alle Zeichenketten abgespeichert haben sollen diese auf dem Bildschirm ausgeben werden. Geben Sie danach den dynamischen Speicher wieder frei.
Bauen Sie ihre **main()** so auf, dass sie die vorher genannten Schritte beliebig oft wiederholen können. (Schleife)

Technology
Arts Sciences
TH Köln

Prof. Dr.-Ing. Dieter Rosenthal
M.Sc. Dženita Schröder-Spahić
M.Sc. Alen Dizdarević
Praktikum „Praktische Informatik II“
TH Köln, Fakultät 07
12.3.2019

C

Aufgabenblatt 1

Aufgabe 1.2: Speicherung der Attribute eines Fahrzeugs

Abgabe: 03.04.19

Bei C gibt es die Möglichkeit eigene normale und komplexe Datentypen zu erstellen. Dies hat den Nutzen, dass Daten strukturiert und zusammen in einem Datentyp abgespeichert werden können. In unserem Beispiel soll die Struktur **typedef struct fahrzeug{..};** als Speicherplatz für die Eckdaten eines Fahrzeuges dienen.

Um die Struktur als eigenen Datentyp zu deklarieren nutzen wir **typedef**.

Die Struktur enthält folgende Eigenschaften:

char *hersteller	- Herstellernamen als Zeichenkette z.B.: "Audi" oder "VW"
char *modell	- Modelnamen als Zeichenkette z.B.: "Golf 4", "A5" oder "Cougar"
char *seriennummer	- erfundene Seriennummer als Zeichenkette z.B.: "1113N331hheU", "Lttz7731Qe" oder "Agg53123456rrT"
int leerGewicht	- Leergewicht
int leistung	- Leistung des Fahrzeugs
int anzReifen	- Anzahl der Reifen

Erstellen Sie eine **main()** in der Sie 4 Strukturen (**fahrzeug f1, f2, f3** und **f4)** deklarieren und setzen Sie alle Eigenschaften per Konsoleneingabe. Nutzen Sie zum Füllen aller Eigenschaften, die vom Typ **char*** sind, die in Aufgabe 1.1 erstellte Funktion **char* zeichenketteEinlesen(void)**. Geben Sie alle 4 Fahrzeuge strukturiert in der Konsole aus.
z.B.:
Fahrzeug 1
Hersteller: Audi
Modell: Q8
Leergewicht: ... kg
etc.

Achten Sie darauf, dass vor dem Programmende alle dynamischen Speicherplätze (in den Strukturen) freigegeben werden.

Prof. Dr.-Ing. Dieter Rosenthal
M.Sc. Dženita Schröder-Spahić
M.Sc. Alen Dizdarević
Praktikum „Praktische Informatik II“
TH Köln, Fakultät 07
12.3.2019

Technology
Arts Sciences
TH Köln

C

Aufgabenblatt 1

Aufgabe 1.3: Sortieren von Fahrzeugen

Abgabe: 10.04.19

In C kann es dazu kommen, dass Datensätze in einer nicht sortierten Form an das Programm weitergegeben werden. Dann ist es nützlich, dass das Programm die Datensätze sortiert. Dazu gibt es verschiedene Möglichkeiten dies, um zu setzen. In dieser Aufgabe sollen Sie einen einfachen Sortieralgorythmus für ein dynamisch erzeugtes Array von **Fahrzeugen** schreiben.

Schreiben Sie dazu folgende Funktion:

Void fahrzeugSortByID(Fahrzeug daten, int Anzahl);**
Diese Funktion soll die Fahrzeuge in dem übergebenen Array nach ihrer **Seriennummer** sortieren.
In der Variable **Anzahl** soll die Anzahl der Fahrzeuge in dem Array übergeben werden. Diese Information ist wichtig, da man nicht anhand eines Arrays ermitteln kann wie viele Elemente dieses enthält.

Schreiben Sie eine Main in der Sie ein Array mit 5 verschiedenen Fahrzeugen erstellen und diese anschließend von Ihrer Funktion sortieren lassen.

Hinweise: Diese Aufgabe kann mit Schleifen, die ineinander verschachtelt werden, gelöst werden.
Nutzen Sie Ihre Ergebnisse der Aufgabe 1.2 als Grundlage für diese Aufgabe