Praktikum 11 12 Dynamischer Speicher/Pointerarray

Gegeben sei eine Binärdatei mit Telefonbucheinträgen. Jeder Datensatz bestehe aus drei Zeichenketten (name, surname, phone). Dabei sei jede Zeichenkette in der Datei durch ein Byte Länge und nachfolgend die Zeichen des Strings gespeichert.

04	'H'	'a'	'n'	's'

Im Programm werde jeder Datzensatz durch folgende Struktur abgebildet:

```
typedef struct
{
  char * name;
  char * surname;
  char * phone;
}tpers;
```

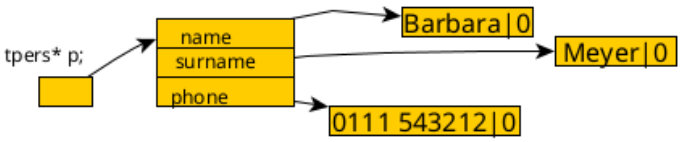
Die Struktur enthält lediglich Pointer. Um die Daten aus der Datei mit einem solchen struct modellieren zu können, muss der Speicher für die Strings noch mittels malloc oder calloc bereit gestellt werden.

Schreiben Sie ein Programm, das aus 2 c-Modulen besteht.

Eine Datei **pers.c** stelle zunächst eine Funktionen readPers zum Lesen aus der Datei und putPers zur Ausgabe eines Datensatzes auf der Konsole bereit.

Dabei ist der notwendige Speicherplatz für die Stuktur und die Zeichenketten dynamisch mittels malloc oder calloc zu allokieren.

ES ist sinnvoll, eine Funktion readStr zu bauen, die jeweils eine Zeichenkette aus der geöffneten Datei in dafür bereit zu stellenden Speicher einliest. Sie liest zunächst die Länge aus der Datei mit fgetc, allokiert den nötigen Speicher (denken Sie an die terminierende 0) und liest dann aus der Datei den String mit fread direkt in den allokierten Speicher.



Eine Datei pers.h enthalte die Strukturtypvereinbarung sowie die Prototypen der öffentlichen Funktionen (hier zunächst readPers und putPers)

Aufgabe 1:

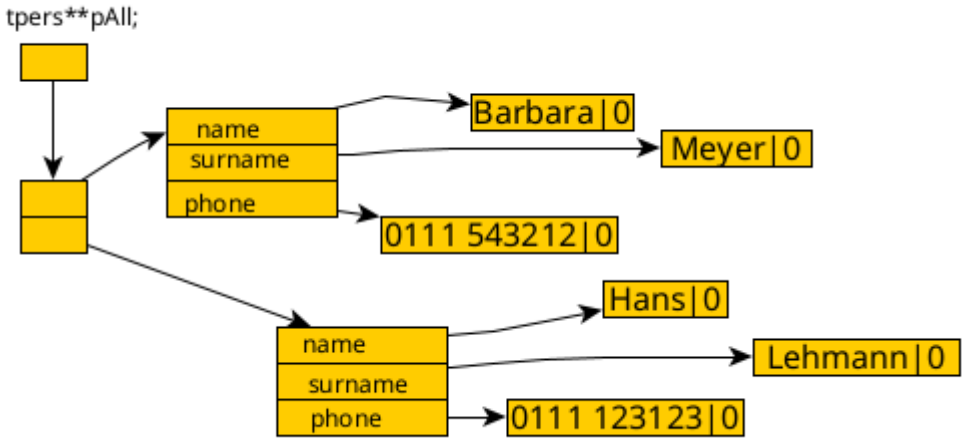
Schreiben Sie ein Programm, das die Datensätze der Datei Satz für Satz liest, ausgibt und danach die allokierten Speicherbereiche wieder freigibt.

Aufgabe 2:

Nun sollen die Daten komplett eingelesen und danach ausgegeben werden. Da wir nicht wissen, wieviele Datensätze die Datei enthält, wollen wir ein dynamisches Pointerarray anlegen. Dazu vereinbaren Sie einen Pointer der Form: **`tpers ** pAll=NULL;`**

Nach dem erfolgreichen Lesen eines Datensatzes, soll der Speicherbereich des Pointerarrays um einen Pointer vergrößert und am Ende der Pointer auf den jeweils neuen Datensatz gespeichert werden.

Es entsteht ein Konstrukt der folgenden Form:



Das ganze Konstrukt sieht dann für zwei Datensätze aus, wie auf dem Bild dargestellt. Das wirkt zunächst kompliziert, ist aber es aber eigentlich nicht. Da bei Pointern auch Arrayschreibweise verwendet werden kann, ist pAll[0]->name der String Barabara und pAll[1]->surname ist Lehmann u.s.w..

Aufgabe 3:

Sortieren Sie den Datenbestand nach dem Einlesen und geben Sie ihn dann in alphabetischer Reihenfolge aus. Dabei werden nun nur die Pointer auf die Daten getauscht, die Daten selbst bleiben, wo sie sind.

Weiterführende Aufgaben:

Schreiben Sie ein Programm zum Erfassen neuer Datensätze. Es ist auch möglich, eines der bereits programmierten Anwendungen dahingehend zu erweitern.