Universität Augsburg
Institut für Informatik

Übung zur Vorlesung Informatik 1 WS 2017/18
Fakultät für Angewandte Informatik
Lehrprofessur für Informatik
Prof. Dr. Lorenz, Marius Brendle, Johannes Metzger, Lev Sorokin 10.01.2018

Übungsblatt 10

Abgabe: 17.01.2018, 12:00 Uhr (Postkasten der Veranstaltung und E-Mail an Tutor)

- Dieses Übungsblatt muss im Team abgegeben werden (Einzelabgaben sind nicht erlaubt).
- Bitte zur Angabe von Namen, Übungsgruppe und Teamnummer das **Deckblatt** verwenden!
- Die **Zeitangaben** geben zur Orientierung an, wie viel Zeit für eine Aufgabe später in der Klausur vorgesehen wäre; gehen Sie davon aus, dass Sie zum jetzigen Zeitpunkt wesentlich länger brauchen und die angegebene Zeit erst nach ausreichender Übung erreichen.

* leichte Aufgabe / ** mittelschwere Aufgabe / *** schwere Aufgabe

Aufgabe 37 * *(Mehrdimensionale Datenstrukturen: Arbeitsspeicherskizzen und Speicherlecks, 12 Minuten)*

Im folgenden sind jeweils Daten und eine Datenstruktur zur Verwaltung dieser Daten vorgegeben. Fertigen Sie jeweils eine Skizze an, die den im Arbeitsspeicher reservierten Speicherbereich zeigt und wie die Daten in diesem Speicherbereich abgelegt werden.

Allgemeine Hinweise:

- Zeichnen Sie eine Speicherzelle pro Datenwert für die Datentypen `char`, `int`, `float`, `double` und adresswertige Datentypen, unabhängig von Speicherbedarf dieser Datentypen.
- In die Speicherzellen schreiben Sie die Datenwerte, falls diese vom Typ `char`, `int`, `float` oder `double` sind.
- Speicherzellen mit undefinierten Werten lassen Sie leer.
- Adresswerte / Werte von Zeigern geben Sie als Pfeile an wie auf den Vorlesungsfolien eingeführt. Verwenden Sie keine konkreten Adresswerte.
- Zwischen nicht zusammenhängenden Speicherbereichen lassen Sie einen Abstand.

a) (*, 2 Minuten)
Wie wird die Matrix

$$\begin{pmatrix} 7 & 1 \\ 0 & 4 \\ 3 & 6 \end{pmatrix}$$

im Arbeitsspeicher abgelegt, wenn sie durch einen Doppelzeiger `int **m` repräsentiert wird, der notwendige Speicherbereich dynamisch genau passend reserviert wurde und die Werte in einer Zeile jeweils in aufeinanderfolgenden Speicherzellen abgelegt werden?

b) (*, 2 Minuten)
Wie wird die Matrix

$$\begin{pmatrix} 7 & 1 \\ 0 & 4 \\ 3 & 6 \end{pmatrix}$$

im Arbeitsspeicher abgelegt, wenn sie durch einen Einfachzeiger `int *m` repräsentiert wird, der notwendige Speicherbereich dynamisch genau passend reserviert wurde und die Werte in einer Spalte jeweils in aufeinanderfolgenden Speicherzellen abgelegt werden?

c) (*, 2 Minuten)
Wie wird die Matrix

$$\begin{pmatrix} 7 & 1 \\ 0 & 4 \\ 3 & 6 \end{pmatrix}$$

im Arbeitsspeicher abgelegt, wenn sie durch ein zweidimensionales Feld `int m[4][3]` repräsentiert wird und die Werte in einer Zeile jeweils in aufeinanderfolgenden Speicherzellen abgelegt werden?

d) (*, 2 Minuten)
Wie wird die Zeichenketten-Liste
`""`
`"A"`
im Arbeitsspeicher abgelegt, wenn sie durch ein Feld von Zeigern `char *list[3]` repräsentiert

wird und der notwendige Speicherbereich für die Zeichenketten dynamisch genau passend reserviert wurde?

e) (*, 4 Minuten)
An welchen Stellen befinden sich in folgendem Programmcode Speicherlecks (es werden die Funktionen zur dynamischen Verwaltung von Matrizen aus der Vorlesung benutzt)? Schreiben Sie eine fehlerbereinigte Version.

```
int main() {
        int **A, **B;
        A = matrix_create(5, 5);
        if (!A)
                return 1;
        matrix_init(A, 5, 5);
        B = matrix_create(5, 5);
        if (!B)
                return 1;
        matrix_init(B, 5, 5);
        matrix_print(A, 5, 5);
        matrix_print(B, 5, 5);
        matrix_destroy(B, 5);
        return 0;
}
```

Aufgabe 38 ** *(Zeiger in Funktionen umbiegen, 13 Minuten)*

a) (**, 4 Minuten)
Betrachten Sie die folgende Funktion aus Kapitel 13 der Vorlesung:

```
char * split(char * w, char c) {
        int i = 0;
        while (w[i] != c && w[i] != '\0')
                ++i;
        if (w[i] == c) {
                w[i] = '\0';
                return &w[i + 1];
        }
        else
                return NULL;
}
```

Implementieren Sie eine abgeänderte Version dieser Funktion mit dem Prototypen
`int split(char *w, char c, char **rest)`, die

- `1` zurückgibt, falls `c` in `w` vorkommt,
- `0` zurückgibt, falls `c` in `w` nicht vorkommt,
- die Adresse des abgeteilten zweiten Teils von `w` nicht zurückgibt, sondern an der Stelle `*rest` speichert, falls für `rest` nicht `NULL` übergeben wurde.

b) (***, 4 Minuten)
Betrachten Sie die folgende Funktion aus Kapitel 14 der Vorlesung:

```
int **matrix_create(int ze, int sp)
{
        int **m, i, k;
        m = malloc(ze * sizeof(int*));
        if (!m)
                return NULL;
        for (i = 0; i < ze; ++i) {
                m[i] = malloc(sp * sizeof(int));
                if (!m[i]) {
                        for (k = 0; k < i; ++k)
                                free(m[k]);
                        free(m);
                        return NULL;
                }
        }
        return m;
}
```

Implementieren Sie eine abgeänderte Version dieser Funktion mit dem Prototypen
`int matrix_create(int ***m, int ze, int sp)`, die

- im Erfolgsfall `1` zurückgibt,
- im Fehlerfall `0` zurückgibt,
- die Adresse des dynamisch reserverierten Speicherbereichs nicht zurückgibt, sondern an der Stelle `*m` speichert.

c) (**, 5 Minuten)
Implementieren Sie ein compilierbares Hauptprogramm mit folgendem Verhalten:

- Es soll maximal 10 Zeichen vom Benutzer einlesen und in einer Zeichenkette abspeichern (längere Eingaben können ohne Fehlerbehandlung abgeschnitten werden).

- Es soll ein Anfangsstück der Eingabe mit Hilfe der Funktion `strtol` in einen `long`-Wert umwandeln, wobei dieses Anfangsstück als Hexadezimalzahl interpreptiert wird.
- Es soll den umgewandelten `long`-Wert und einen ggf. nicht umgewandelten Rest der Eingabe ausgeben.

Aufgabe 39 ** *(Mehrdimensionale Datenstrukturen verwalten, 17 Minuten)*

a) (***, 6 Minuten)
Implementieren Sie eine Funktion `int **matrix_cpy_deep(int **m, int ze, int sp)`, die eine **tiefe** Kopie einer Matrix `m` mit `ze` Zeilen und `sp` Spalten erstellt, im Erfolgsfall die Adresse dieser Kopie zurückgibt und sonst `NULL` zurückgibt.

b) (*, 4 Minuten)
Die Addition von zwei $n \times m$-Matrizen $A = (a_{i,j})_{1 \leq i \leq n, 1 \leq j \leq m}$ und $B = (b_{i,j})_{1 \leq i \leq n, 1 \leq j \leq m}$ ist wie folgt definiert:

$$A + B =: (c_{i,j})_{1 \leq i \leq n, 1 \leq j \leq m}$$

mit $c_{i,j} := a_{i,j} + b_{i,j}$. Implementieren Sie eine Funktion

```
void matrix_add(int a[][MAX_COL], int b[][MAX_COL], int n, int m),
```

die die $n \times m$-Matrizen `a` und `b` addiert und das Ergebnis in `a` speichert (`MAX_COL` ist eine symbolische Konstante für die maximale Anzahl der Spalten der verwendeten Matrizen).

c) (**, 3 Minuten)
Implementieren Sie eine Funktion `void stringlist_swap(char *list[], int i, int j)`, die die `i`-te Zeichenkette mit der `j`-ten Zeichenkette in `list` vertauscht.

d) (**, 4 Minuten)
Implementieren Sie eine Funktion `void matrix_init_mult(int *m, int n)`, die eine über den Einfachzeiger `m` verwaltete $n \times n$-Matrix $(m_{i,j})_{1 \leq i \leq n, 1 \leq j \leq n}$ mit folgenden Werten initialisiert:

$$m_{i,j} := i \cdot j$$

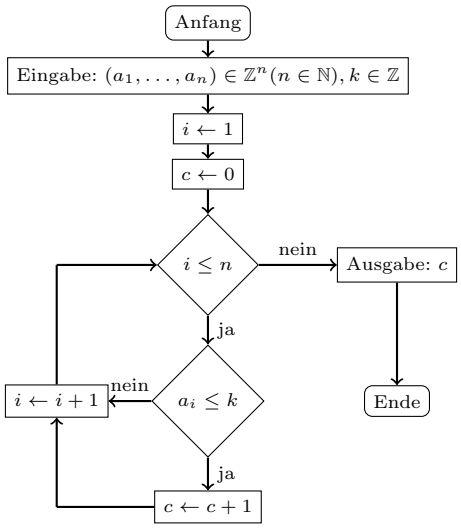
Aufgabe 40 * *(Algorithmus-Darstellungen, 17 Minuten)*

a) (5 Minuten)
Geben Sie den folgenden als Pseudocode gegebenen Algorithmus als Programmablaufplan an.

Algorithmus : EchteTeiler
Eingabe : $n \in \mathbb{N}, n > 0$
$i \leftarrow 2$;
$c \leftarrow 0$;
solange $i < n$ **tue**
  wenn ($n \mod i = 0$) **dann**
    $c \leftarrow c + 1$;
  $i \leftarrow i + 1$;
Ausgabe : c

b) (5 Minuten)
Eine quadratische Matrix $A = (a_{i,j})_{1 \leq i \leq n, 1 \leq j \leq n}$ mit reellwertigen Einträgen heißt *symmetrisch*, falls $a_{i,j} = a_{j,i}$ gilt für $1 \leq i \leq n$ und $1 \leq j \leq n$. Entwerfen Sie einen Algorithmus als Struktogramm, der für eine eingegebene Matrix mit reellwertigen Einträgen 1 zurückgibt, falls diese symmetrisch ist, und sonst 0 zurückgibt.

c) (5 Minuten)
Implementieren Sie den folgenden als Programmablaufplan gegebenen Algorithmus als C-Funktion.



d) (2 Minuten)
Sei A das Alphabet aller ASCII-Zeichen. Was leistet der folgende in Form eines Struktogramms gegebene Algorithmus?