

Lösung Übung 3

infT

1.

In einem Programm muss bei der Eingabe einer Zahl dafür gesorgt werden, dass nur Integer-Zahlen kleiner Null eingegeben werden können. Wie muss dieser Programmteil aussehen, damit die Forderung erfüllt wird? Benutzen Sie hierfür die `do`-Schleife.

L:

```
/* negInput.c */
#include <stdio.h>

int main() {
    int zahl;

    do {
        printf("Geben Sie eine Ganzzahl kleiner Null ein: ");
        scanf("%i", &zahl);
    } while (zahl >= 0);
}
```

2.

Gegeben ist das folgende Magische Quadrat von Dürer (es enthält u.a. alle ganzen Zahlen von 1 bis 16):

```
16  3  2 13
 5 10 11  8
 9  6  7 12
 4 15 14  1
```

- Speichern Sie das Quadrat in einem C-Programm in einem 2-dim. Array.
- Berechnen Sie die Summen aller Zeilen, Spalten und Hauptdiagonalen und geben Sie die Resultate am stdout aus.
- Schreiben Sie ein zweites Programm, das die Aufgabe b) löst, jetzt allerdings mit dem Quadrat in einem ein-dim. Array.

L:

```
/* duerer1.c */
#include <stdio.h>

#define DIM 4

int main() {
    int row, col;
    int sum1, sum2;
    int magicSquare[][4] = {
        {16, 3, 2, 13},
        { 5, 10, 11, 8},
        { 9, 6, 7, 12},
        { 4, 15, 14, 1},
    };
    int cols[] = {0, 0, 0, 0};

    printf("Summen der Zeilen: ");
    for (row = 0; row < DIM; ++row) {
        sum1 = 0;
        for (col = 0; col < DIM; ++col) {
            sum1 += magicSquare[row][col];

            cols[col] += magicSquare[row][col];
        }
        printf(" Zeile%i: %i ", row, sum1);
    }
}
```

```

printf("\n \n");

printf("Summen der Spalten: Spalte0: %i Spalte1: %i Spalte2: %i "
      "Spalte3: %i ",
      cols[0], cols[1], cols[2], cols[3]);
printf("\n \n");

printf("Summen der Diagonalen: ");
sum1 = sum2 = 0;
for (row = 0; row < DIM; ++row) {
    sum1 += magicSquare[row][row];
    sum2 += magicSquare[row][DIM - 1 - row];
}
printf("Diag1: %i Diag2: %i ", sum1, sum2);
}

/* duerer2.c */
#include <stdio.h>

#define DIM 4

int main() {
    int row, col;
    int sum1, sum2;
    int magicSquare[] = {
        16, 3, 2, 13,
        5, 10, 11, 8,
        9, 6, 7, 12,
        4, 15, 14, 1,
    };
    int cols[] = {0, 0, 0, 0};

    printf("Summen der Zeilen: ");
    for (row = 0; row < DIM; ++row) {
        sum1 = 0;
        for (col = 0; col < DIM; ++col) {
            sum1 += magicSquare[row * DIM + col];

            cols[col] += magicSquare[row * DIM + col];
        }
        printf(" Zeile%i: %i ", row, sum1);
    }
    printf("\n \n");

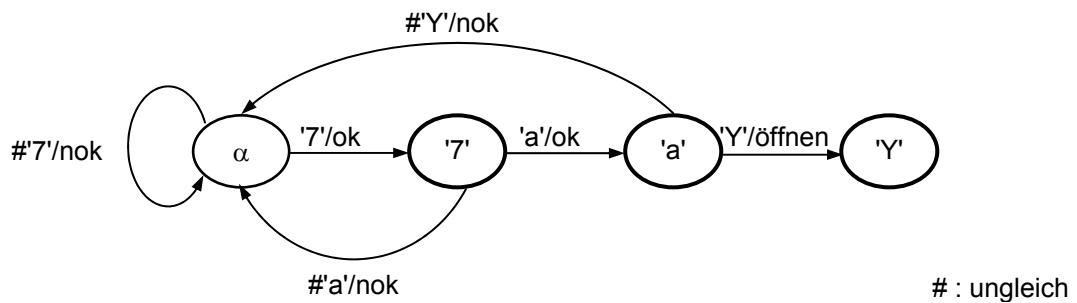
    printf("Summen der Spalten: Spalte0: %i Spalte1: %i Spalte2: %i "
          "Spalte3: %i ",
          cols[0], cols[1], cols[2], cols[3]);
    printf("\n \n");

    printf("Summen der Diagonalen: ");
    sum1 = sum2 = 0;
    for (row = 0; row < DIM; ++row) {
        sum1 += magicSquare[row * DIM + row];
        sum2 += magicSquare[row * DIM + (DIM - 1 - row)];
    }
    printf("Diag1: %i Diag2: %i ", sum1, sum2);
}

```

3.

Gegeben ist das nachstehende Zustandsdiagramm für ein elektronisches Schloss. Schreiben Sie ein C-Programm, das Eingaben entgegen nimmt und das Öffnen des Schloss simuliert (Bei Öffnung entsprechende Ausgabe an den Bildschirm *und* Verlassen des Programms).



Code zum Öffnen: <ret> 7 <ret> a <ret> Y <ret>

L:

```

/* eschloss.c */
#include <stdio.h>

#define ALPHA 999 // Ausgangszustand
#define FIRST_CHAR '7' // Zustand und Chars
#define SECOND_CHAR 'a'
#define THIRD_CHAR 'Y'

int main() {
    int currState;
    char input[21];
    int num;

    printf("Elektronisches Schloss\n");

    currState = ALPHA;
    for (; ; ) {
        printf("> ");
        while (scanf("%20s", input) < 1) { }

        switch (input[0]) {
            case FIRST_CHAR:
                if (currState != ALPHA) currState = ALPHA;
                currState = FIRST_CHAR;
                break;
            case SECOND_CHAR:
                if (currState != FIRST_CHAR) currState = ALPHA;
                currState = SECOND_CHAR;
                break;
            case THIRD_CHAR:
                if (currState != SECOND_CHAR) currState = ALPHA;
                printf("Schloss offen!\n");
                return 0;
            default:
                currState = ALPHA;
                break;
        }
    }
}

```

4.
Welchen Fehler enthält der folgende Programmausschnitt?

```
...
char puffer[20];
strcpy(puffer, "Programmiersprache C");
...
```

L:
puffer ist um ein Zeichen zu kurz, da der Delimiter mit berücksichtigt werden muss.

5.
Testen Sie, ob das Zuweisen von Strukturvariablen das erhoffte Feld-weise Kopieren zeitigt. Wie sieht es aus, wenn verschachtelte Strukturen vorliegen oder eine Struktur ein Array enthält?

L:
Mit Beispiel aus dem Unterricht:

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

struct Assistant {
    char last[16];
    char first[11];
    int assists;
    int year;
};

struct HallOfFame {
    struct Assistant players[10];
    int nPlayers;
} hr, hr2;

void printHof(struct HallOfFame hof) {
    int i;
    for (i = 0; i < hof.nPlayers; ++i) {
        printf("%d: {%s, %s, %d}\n", hof.players[i].year, hof.players[i].last,
            hof.players[i].first, hof.players[i].assists);
    }
}

int main() {
    int i;
    hr.nPlayers = 0;

    puts("Insert first player");
    strcpy(hr.players[hr.nPlayers].last, "Ohm");
    strcpy(hr.players[hr.nPlayers].first, "Ted");
    hr.players[hr.nPlayers].assists = 72;
    hr.players[hr.nPlayers++].year = 1925;
    printHof(hr);

    puts("Insert next player");
    strcpy(hr.players[hr.nPlayers].last, "Enz");
    strcpy(hr.players[hr.nPlayers].first, "Mark");
    hr.players[hr.nPlayers].assists = 61;
    hr.players[hr.nPlayers++].year = 1961;
    printHof(hr);
}
```

```

puts("Deep copy struct");
printHof(hr);
hr2 = hr;                // kopiere via Zuweisung
printHof(hr2);

puts("Show that two different structs exist");
strcpy(hr.players[hr.nPlayers].last, "?");
strcpy(hr.players[hr.nPlayers].first, "??");
hr.players[hr.nPlayers].assists = -1;
hr.players[hr.nPlayers++].year = -2;
printHof(hr);

printHof(hr2);
}

```

Der gcc nimmt ein Deep-Copy vor, d.h. es handelt sich um zwei verschiedene Datenstrukturen mit gleichen Member-Werten, sogar wenn verschalchtet.

5.

Welche Ausgabe liefert das folgende Programm?

```

#include <stdio.h>

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

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

    printf("\n%s", kunde[0].name);
    printf("\n%c", kunde[2].name[0]);
    printf("\n%i", strlen(kunde[1].name));
    printf("\n%li", kunde[1].knr);
    printf("\n%li", kunde[3].knr);

    return 0;
}

```

L:

```

Meier
Z
7
82765
0 // static Variablen werden in das Data Segment geschrieben, das
// bei Programmbeginn mit Nullen initialisiert sein kann...

```