

## Einführung in die Programmiersprachen C und C++

Prof. Dr. Ulf Rehmann, Fakultät für Mathematik

Übungsblatt 5 (4 Seiten)

### Zur Entspannung einige Spielprogramme:

#### Turm von Hanoi, Acht-Damen-Problem (sogar für n Damen), Rässelsprung

##### 5.1 Der Turm von Hanoi:

Ein Turm aus einer Anzahl von der Größe nach geordneten übereinandergelegten Scheiben (größte zuerst) ist von einer Position ("links") in eine andere Position ("rechts") zu versetzen. Dabei darf nur eine Hilfsposition ("mitte") zur Zwischenlagerung von Scheiben verwendet werden. Bedingung: Nie darf in einer der Positionen eine Scheibe auf eine kleinere gelegt werden. Für das klassische Standardproblem ist  $n = 10$ .

```
/* titel: turm.c, turm von hanoi */
#include <stdio.h>
typedef enum { links, mitte, rechts } turm;
main() {
    int scheibenzahl; void bezeuge();
    while (1) {
        printf("\nScheibenzahl (Abbruch mit 0): ");
        while ( scanf("%d", &scheibenzahl) == 0 ) {
            getchar();
            printf("\nFalsche Eingabe, bitte Zahl >= 0 eingeben: ");
        }
        if (scheibenzahl <= 0) break;
        printf ("%d %s\n",scheibenzahl,"Scheiben erfordern folgende Bewegungen:");
        bezeuge (scheibenzahl, links, mitte, rechts);
    }
}

void bezeuge (int anzahl, turm von, turm mittels, turm nach) {
    void bezeuge_scheibe();
    if ( anzahl == 1 )
        bezeuge_scheibe (von, nach);
    else {
        bezeuge ( anzahl - 1, von, nach, mittels);
        bezeuge_scheibe ( von, nach );
        bezeuge ( anzahl - 1, mittels, von, nach);
    }
}

void bezeuge_scheibe ( turm von, turm nach ) {
    void print_turm();
    printf ("Scheibe von "); print_turm (von);
    printf (" nach ");      print_turm (nach);
    printf ("\n");
}

void print_turm ( turm welcher ) {
    switch (welcher) {
        case links : printf ("LINKS "); break;
        case mitte : printf ("MITTE "); break;
        case rechts : printf ("RECHTS"); break;
    }
}
```

*Aufgabe 5.2: Modifizieren Sie das Programm, so dass alle möglichen Lösungen ausgegeben werden.*

*Aufgabe 5.3: Ändern Sie die Ausgabe so ab, dass ein Feld von Charakteren der Größe n dargestellt wird, wobei die Positionen der Damen durch ein '\*' gekennzeichnet wird.*

*Aufgabe 5.1: Schreiben Sie das Programm so um, dass die Anzahl der erforderlichen Bewegungen ausgegeben wird.*

##### 5.2 Das n-Damen-Problem:

n 'Damen' sind auf einem Schachbrett der Größe n mal n so aufzustellen, dass keine von einer anderen angegriffen ist im Sinne der Regeln der Damenzüge im Schachspiel.

Typische Ein- und Ausgabe:

Welche Zahl ( < 20 ) ? 8

1 5 8 6 3 7 2 4

Dies bedeutet: Die Dame in der ersten Spalte steht in Zeile 1, die in der zweiten Spalte steht in Zeile 5 usw. Die Funktion `int versuche()` implementiert einen "Backtracking"-Algorithmus.

```
/* titel: 8.c 8 Damen (und mehr) */
#include <stdio.h>
#define ZZ 20 /* Zahl der Zeilen und Spalten */
int max, DAME[ ZZ ], ZEILE[ ZZ ], DIA0[ 2*ZZ-1 ], DIA1[ 2*ZZ-1 ];
main() {
    int i;
    printf("Welche Zahl ( < %d, Abbruch mit 0 ) ? ", ZZ);
    while( scanf("%d", &max)==0 ) {}
    getchar();
    printf("\nFalsche Eingabe, bitte Zahl eingeben: ", ZZ);
}
if (max <= 0) exit(0);
if (max > ZZ) max = ZZ-1;
else if (max < 1) max = 1;

if ( versuche(0) )
    for (i=0; i < max; i++)
        printf("%d ", DAME[i]);
    printf("\n");

int versuche(int i){
    int j, q;
    for (j = q = 0; q == 0 && j < max; j++) {
        if ( ZEILE[j] == 0 && DIA0[i+j] == 0 && DIA1[i-j+max-1] == 0 ) {
            DAME[i] = j+1;
            ZEILE[j] = DIA0[i+j] = DIA1[i-j+max-1] = 1;
            if (i == max - 1)
                q = 1;
            else
                if ( (q = versuche(i+1)) == 0)
                    ZEILE[j] = DIA0[i+j] = DIA1[i-j+max-1] = 0;
        }
    }
    return(q);
}
```

## 5.3 Rösselsprung:

Das folgende Programm berechnet eine Folge von Springer-Zügen im Sinne des Schachspiels, die ein ganzes Schachfeld der Größe  $ZZ \times ZZ$  bedeckt dertart, dass jedes Feld genau einmal erreicht wird.

```

/* titel: springer.c, Roeselsprung-Programm */
#include <stdio.h>
#define ZZ 8

int z;

int FELD[ ZZ ][ ZZ ];
int a[] = { 2, 1, -1, -2, -2, -1, 1, 2 }; /* Feld
int b[] = { 1, 2, 2, 1, -1, -2, -2, -1 }; /* Koordinaten der
int xx, yy; /* Springerzuege */

main() {
    z = 4;
    while ( 1 ) {
        printf("Zeilen: ");
        while ( scanf("%d", &z) != 1 ) {
            getchar();
        }
        printf("\nFehlerhafte Eingabe, bitte Zahl < %d eingeben: ", ZZ);
    }
    if ( z <= 1 || z >= ZZ) exit(0);
    { int i, j;
      for (i=0; i<z; i++)
        for (j=0; j<z; j++)
          FELD[i][j] = 0;
    }

    printf("Anfangswerte xx yy <= %d: ", z);
    while (scanf("%d %d", &xx, &yy) != 2) {
        getchar();
    }
    printf("\nFehlerhafte Eingabe, bitte zwei Zahlen <= %d eingeben: ", z);
    }
    if ( versuche( xx-1, yy-1 ) )
        drucke();
    else printf("\nkeine Loesung\n");
}

versuche( int x, int y) {
    int k, q, u, v;
    for (k = q = 0; q == 0 && k < 8; k++) {
        u = x+a[k]; v = y+b[k];
        if ( 0 <= u && u < z && 0 <= v && v < z && FELD[u][v] == 0 ){
            FELD[u][v] = FELD[x][y] + 1;
            if ( FELD[u][v] == z*z )
                q = 1;
            else
                if ( (q = versuche( u, v ) == 0)
                    FELD[u][v] = 0;
        }
    }
    return(q);
}

```

```

drucke() {
    int i, j;
    printf("\n");
    for (i = 0; i < z; i++) {
        for (j = 0; j < z; j++)
            printf("%3d", FELD[i][j]);
        printf("\n");
    }
}

```

Typische Ein- und Ausgabe:

```

Zeilen: 6
Anfangswerte xx yy = 1 1

1 16 7 26 11 14
34 25 12 15 6 27
17 2 33 8 13 10
32 35 24 21 28 5
23 18 3 30 9 20
36 31 22 19 4 29

```

*Aufgabe 5.4: Modifizieren Sie das Programm dertart, dass ein "zyklischer" Rösselsprung gefunden wird, so dass also das erste Feld durch einen Springerzug vom letzten Feld aus erreicht wird.*

*Aufgabe 5.5: Schreiben Sie eine Version des Programms, die alle zyklischen Lösungen findet.*