

Einführung in C++ – Übung 2

Testatgruppe A (Isaak)

Rasmus Diederichsen Simon Kern

22. Oktober 2014

Aufgabe 2.1

Listing 1: easter.c

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  /*
4   * Reads a year from stdin and calculates the date of easter for
      that year,
5   * which it prints on stdout. This is done until EOF is read.
6   * @return 0 on success
7   */
8  int main(void)
9  {
10     int year, golden_year, century,
11         leap_years_skipped, moon_coor_fact,
12         D, epact, day;
13     char * month = malloc(6 * sizeof(char));
14     /* User interaction */
15     /* printf("Please enter year: "); */
16
17     while (scanf("%d", &year) != EOF)
18     {
19
20         if (year < 0)
21         {
22             printf("Can't work with years B.C.\n");
23             continue;
24         }
25
26         golden_year = (year % 19) + 1;
27         century = (year / 100) + 1;
28         leap_years_skipped = (3 * century / 4) - 12;
29         moon_coor_fact = ((8 * century + 5) / 25) - 5;
30         D = (5 * year / 4) - leap_years_skipped - 10; /* what should
31             I call this? */
32         epact = (11 * golden_year + 20 + moon_coor_fact -
33             leap_years_skipped) % 30;
```

```

32         if ((epact == 25 && golden_year > 11) || epact == 24) epact
33             ++;
34         day = 44 - epact;
35         if (day < 21) day += 30;
36         day += 7 - ((D + day) % 7);
37         if (day > 31) {
38             month = "April";
39             day = day - 31;
40         }
41         else month = "March";
42         printf("%d_-%d_s_%d\n", year, month, day);
43     }
44     return 0;
45 }

```

Listing 2: makefile for easter.c

```

1  #
2  # Einfuehrung in die Programmiersprache C++
3  #
4  # Makefile fuer Aufgabe 2 (easter)
5  #
6
7  CC          = gcc
8  INFILE      = years.in
9  OUTFILE     = easter_dates.out
10 CORRECT_OUTFILE = correct_easter_dates.out
11
12
13 easter: easter.o
14     $(CC) easter.o -o easter
15
16 easter.o: easter.c
17     $(CC) -Wall -Wstrict-prototypes -ansi -pedantic -c easter.c
18
19 test:
20     ./easter < $(INFILE) > $(OUTFILE)
21     ./run_test $(OUTFILE) $(CORRECT_OUTFILE)
22
23 check:
24     ./c_style_check.py easter.c

```

Aufgabe 2.2

Listing 3: sudoku.c

```

1  #include <stdio.h>
2  #include "sudoku.h"
3
4
5  int is_valid(int z, int i, int j, int sudoku[9][9])
6  {
7      int x, y;
8      int subfield_x_ind = (i / 3) * 3; /* check the 3x3 field
9      containing (i,j) */
10     int subfield_y_ind = (j / 3) * 3;

```

```

10     if ((z < 1 || z > 9) || (sudoku[i][j] != 0)) return 0; /* if out
        of bounds or not free */
11     for (x = 0; x < 9; x++) /* check row and column intersecting at
        (i,j) */
12         if (sudoku[x][j] == z || sudoku[i][x] == z) return 0;
13     for (x = subfield_x_ind; x < subfield_x_ind + 3; x++){
14         for (y = subfield_y_ind; y < subfield_y_ind + 3; y++){
15             if(sudoku[x][y] == z) return 0;
16         }
17     }
18     return 1; /* if no probs found, success! */
19 }
20
21 int solve_sudoku(int i, int j, int sudoku[9][9])
22 {
23     int z; /* the number to try */
24     while (i < 9 && (sudoku[i][j] != 0))
25     { /* find next free field */
26         j++;
27         if (j >= 9)
28         {
29             j = 0;
30             i++;
31         }
32     }
33
34     if (i >= 9) return 1; /* no free fields? Already done. */
35     else
36     {
37         for (z = 1; z <= 9; z++) /* try all numbers */
38         {
39             if (is_valid(z, i, j, sudoku)) /* if assignment valid ...
                */
40             {
41                 sudoku[i][j] = z;
42                 if (!solve_sudoku(i, j, sudoku)) sudoku[i][j] = 0; /*
                    ... continue and try with next field */
43                 else return 1; /* wenn the last field is to be set and
                    it works, the function will jump here and bubble up
                    true */
44             }
45         }
46     }
47     return 0; /* if control jumps here, it means no number was valid
        , so return false */
48 }
49
50 void print_sudoku(int sudoku[9][9])
51 {
52     int i,j;
53     printf("+-----+-----+-----+\n");
54     for (i = 0; i < 9; i++) {
55         for (j = 0; j < 9; j++) {
56             if (j == 3 || j == 6)
57                 printf("|_d_",sudoku[i][j]);
58             else
59                 printf("%d_",sudoku[i][j]);

```

```

60     }
61     printf("\n");
62     if (i == 2 || i == 5)
63         printf("+-----+-----+\n");
64     }
65     printf("+-----+-----+\n");
66 }
67 int main(void)
68 {
69
70     if(solve_sudoku(0,0,field1)) print_sudoku(field1);
71     else printf("Unsolvable.\n");
72     return 0;
73 }

```

Listing 4: sudoku.h

```

1  int field1[][9] = {
2      {0, 0, 0, 0, 0, 8, 0, 3, 0},
3      {0, 3, 0, 5, 0, 0, 4, 7, 1},
4      {2, 0, 0, 1, 0, 0, 6, 9, 0},
5
6      {5, 0, 0, 0, 0, 2, 1, 0, 0},
7      {1, 2, 4, 0, 0, 0, 9, 6, 3},
8      {0, 0, 6, 4, 0, 0, 0, 0, 2},
9
10     {0, 8, 9, 0, 0, 5, 0, 0, 7},
11     {3, 5, 2, 0, 0, 9, 0, 4, 0},
12     {0, 1, 0, 3, 0, 0, 0, 0, 0}
13 };
14
15 /**
16  * Function to check whether an assignment for a given field is
17  * legal.
18  *
19  * @param z The number to be assigned.
20  * @param i The row number of the field to be assigned.
21  * @param j The column number of the field to be assigned.
22  * @param sudoku the Sudoku field against which to check.
23  * @return 1 on success, 0 on failure.
24  */
25 int is_valid(int z, int i, int j, int sudoku[9][9]);
26
27 /**
28  * Function to attempt solving a sudoku.
29  *
30  * @param i The row number at which to set the first number.
31  * @param j The column number at which to set the first number.
32  * @return 1 if a solution was found, 0 otherwise.
33  */
34 int solve_sudoku(int i, int j, int sudoku[9][9]);
35
36 /**
37  * Function to print a sudoku array on stdout.
38  *
39  * @param sudoku The puzzle to be printed.
40  */
41 void print_sudoku(int sudoku[9][9]);

```

Listing 5: makefile for sudoku.c

```
1 hello: sudoku
2     gcc -Wall -Wstrict-prototypes -ansi -pedantic sudoku.c -o
      sudoku
3 clean:
4     rm sudoku
5 test:
6     ./sudoku
```