Programming in C II, Module II
Jacobs University Bremen
Dr. Jürgen Schönwälder

Course: JTSK-320112
February 10th, 2017

Assignment 2 - Pointers and Dynamically Allocated Multi-dimensional Arrays

- The problems of this assignment must be solved in C.
- Your programs should have the input and output formatting according to the testcases listed after the problems.
- Your programs should consider the grading rules posted on the course web page: `http://cnds.eecs.jacobs-university.de/courses/c2-2017/`

Problem 2.1 *Pointer arithmetic* (2 points)

Presence assignment, due by 18:30 h today

Write a program that counts the number of elements in an array until encountering the first negative value without the usage of any integer counter variables (except for the loop for reading the elements of the array), but with the usage of pointers and pointer arithmetic.
Your program should read an `int` for the length of the array and an array of floats (containing at least one negative value) from the standard input. The number of non-negative values before the first negative value should be outputted on the standard output.
You can assume that the input will be valid and the array will always contain at least one negative value.

Testcase 2.1: input

```
5
1.2
3.4
-5.86
4
5.45
```

Testcase 2.1: output

```
Before the first negative value: 2 elements
```

Problem 2.2 *Concatenating two strings* (2 points)

Presence assignment, due by 18:30 h today

Write a program to concatenate two strings (with the length of at most 80 characters) putting the result in a dynamically allocated array of chars with exact size.
Your program should read from the standard input two strings which may be statically allocated. The dynamically allocated string containing the concatenation result of the two strings should be printed on the standard output.
You can assume that the input will be valid.

Testcase 2.2: input

```
one
two
```

Testcase 2.2: output

```
Result of concatenation: onetwo
```

Problem 2.3 *Dynamically allocated matrix multiplication* (3 points)

Write a program that computes the multiplication of two dynamically allocated matrices. Your program should dynamically allocate the memory for the three matrices (two input matrices and the result matrix). You should write functions for reading a matrix from the standard input, printing a matrix to the standard output, and finally a function for computing the multiplication of two matrices. At the end, do not forget to deallocate the memory used by the three matrices. The product of two matrices A and B of dimensions $n \times m$ and $m \times p$ can be calculated as:

$$C_{ik} = \sum_{j=1}^{m} A_{ij} \cdot B_{jk}, \text{ with } i = 1, \ldots, n \text{ and } k = 1, \ldots, p.$$

Your program should read three integers (the dimensions n, m and p) and the elements of two integer matrices from the standard input. The result of the matrix multiplication should be printed on the standard output.
You can assume that the input will be valid.

Testcase 2.3: input

```
2
2
2
1
2
3
4
1
0
0
1
```

Testcase 2.3: output

```
Matrix A:
1 2
3 4
Matrix B:
1 0
0 1
The multiplication result AxB:
1 2
3 4
```

Problem 2.4 *Printing dynamically allocated 3D array sections* (4 points)

Write a program that dynamically allocates memory for a 3D array of integers and prints the 2D sections parallel to the "$x0y$ axis" (considering the array dimensions column-dimension, row-dimension and depth-dimension similar to the geometrical x, y and z dimensions) of a 3D array. Your program should read three integer values corresponding to the dimensions of a 3D array (in the order of rows, columns, depth) and should dynamically allocate the memory for this 3D array. You should write functions for reading the elements of the 3D array from standard input (first iterating through rows, then columns and then the depth) and finally a function for printing the 2D sections of the 3D array which are parallel to the "$x0y$ axis". Do not forget about the allocation and deallocation of the memory.
You can assume that the input will be valid.

Testcase 2.4: input

```
2
2
3
1
2
3
1
2
3
1
2
3
1
2
3
```

Testcase 2.4: output

```
Section 1:
1 1
1 1
Section 2:
2 2
2 2
Section 3:
3 3
3 3
```

How to submit your solutions

- Your source code should be properly indented and compile with `gcc` without any warnings. You should use `gcc -Wall -Werror -o program program.c`. Insert suitable comments (not on every line . . .) to explain what your program does.
- Please name the programs according to the suggested filenames (they should match the description of the problem) in Grader. Otherwise you might have problems with the inclusion of header files. Each program **must** include a comment on the top like the following:

```
/*
   JTSK-320112
   a2_p1.c
   Firstname Lastname
   myemail@jacobs-university.de
*/
```

- You have to submit your solutions via *Grader* at
  `https://grader.eecs.jacobs-university.de`.
  If there are problems (but **only** then) you can submit the programs by sending mail to `j.schoenwaelder@jacobs-university.de` **with a subject line that begins with JTSK-320112**. **It is important that you do begin your subject with the coursenumber, otherwise I might have problems to identify your submission.**
- Please note, that after the deadline it will not be possible to submit any solutions. It is useless to send late solutions by mail, because they will not be accepted.

This assignment is due by Wednesday, February 15th, 10:00 h.