Programming in C I, Group B
Jacobs University Bremen
Dr. Kinga Lipskoch
Instructor: Xu He

Course: JTSK-320111
September 22nd, 2017

Assignment 4 - Functions, Arrays and Strings

- The problems of this assignment must be solved in C.
- The TAs are grading solutions to the problems according to the following criteria: http://minds.jacobs-university.de/sites/default/files/uploads/teaching/CProgrammingFall17/Grading-Criteria-c.pdf

Problem 4.1 *Wrong length* (1 point)

Presence assignment, due by 18:30 h today

The program below should compute and print the length of a string. Why does it always print the length of 0 independently from the input string? Fix the program such that it prints the correct length. Formulate your answer to the question and explain your fix within a comment.

```
#include <stdio.h>
int length(char str[]) {
    int idx;
    /* Loop until end of string */
    for (idx = 0; str[idx] != '\0'; ++idx)
        /* do nothing */
    return idx;
}

int main() {
    char line[100];
    while(1) {
        printf("Enter string:\n");
        fgets(line, sizeof(line), stdin);
        printf("Length (including newline) is: %d\n", length(line));
    }
    return 0;
}
```

Problem 4.2 *Millimeters and meters* (1 point)

Presence assignment, due by 18:30 h today

Write a program that converts a value that is entered from the keyboard in millimeters to meters. For solving this problem write and call a function `double convert(int mm)` that does the actual conversion. Print the result of the conversion from the `main` function.
You can safely assume that the input will be valid.

Problem 4.3 *Ounces and pounds* (1 point)

Presence assignment, due by 18:30 h today

Write a program that converts the US units of mass (ounces and pounds) to kilograms. First read the weight of an object expressed by two values for pounds and ounces from the keyboard and then convert the units of mass using the function below as one value corresponding to the converted sum of the two previous values (written by you as well)

```
double to_kilogram(int pound, int ounce);
```

that does the actual conversion. Print the result on the screen from the `main` function. Note that: $1\ pound = 453.6\ g, 1\ ounce = 28.350\ g$

You can safely assume that the input will be valid.

Problem 4.4 *Printing a frame* (1 point)

Write a program which reads two integers `n` and `m`, and a character `c` from the keyboard. This program should define and call a function with the prototype:

```
void print_frame(int n, int m, char c);
```

which prints a frame of size `n`x`m` consisting of the character `c` and space as in the following testcase.

You can safely assume that the input will be valid.

Testcase 4.4: input

```
4
7
$
```

Testcase 4.4: output

```
$$$$$$$
$     $
$     $
$$$$$$$
```

Problem 4.5 *Computing sum and average* (1 point)

Write a program where you can enter up to 8 integers from the keyboard. If the number entered is equal to -99, stop reading numbers from the keyboard and compute the sum and average of all values (excluding -99) using two functions (one for the sum and one for the average). You have to write these functions as well. Print your results on the screen from the `main` function.

You can safely assume that the input will be valid.

Make sure you consider all the cases: less than 8 numbers or exactly 8 numbers might be entered. After all the numbers have been entered you need to make sure that the sum and average are computed.

Problem 4.6 *Determine position of minimum value* (1 point)

Write a program which reads an integer value `n` followed by `n` other integer values as elements of an array with not more than 50 elements. Write also a function with the prototype:

```
int posmin(int v[], int n);
```

which determines and returns the position of the minimum value of `v`. Print the elements of the array separated by space and then on a new line the position of the minimum value on the screen from the `main` function.

You can safely assume that the input will be valid.

Problem 4.7 *Swap two values* (1 point)

Write a program which reads two double values from the keyboard. Then write three functions. The first function should return the sum of the two double values and should have the prototype:

```
double sum(double a, double b);
```

The second function should "return" by address the sum of the two double values and should have the prototype:

```
void sumbyref(double a, double b, double *s);
```

The third function should swap the two double values and "return" their modification by address. It should have the prototype:

```
void swapbyref(double *a, double *b);
```

Show that the calls of the first two functions have the same effect. Also show what is the effect of calling `swapbyref`.

You can safely assume that the input will be valid.

Problem 4.8 *Working with strings* (2 points)

Write a program where you can enter two strings `s1` and `s2` from the keyboard. You can assume that the input strings will not be longer than 100 characters. The program should do the following:

1. Print on the screen the correct lengths of both strings,
2. Print on the screen the concatenation of `s1` with `s2`,
3. Declare a third string, copy `s1` into it and print the third string on the screen,
4. Compare the two strings `s1` and `s2` and print a corresponding message on the screen,

5. Read a character `c` from the keyboard and search for `c` in `s1`. The position of the first occurrence of `c` within `s1` should be printed on the screen. If the character is not contained in the string then print a corresponding message on the screen.

For solving this problem use the string functions from `string.h`.
Learn how to use them with the help of the man pages or online documentation.

Bonus Problem 4.9 *Detect all occurrences of a character* (1 point)

Modify your solution for **Problem 4.8** such that all occurrences of the character `c` can be detected in `s1`. All positions of occurrence should be printed on the screen.

How to submit your solutions

- Your source code should be properly indented and compile with `gcc` without any warnings (You can use `gcc -Wall -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.
  Each program **must** include a comment on the top like the following:

```
/*
   JTSK-320111
   a4_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 `x.hed@jacobs-university.de` **with a subject line that begins with JTSK-320111**.
  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, September 27th, 10:00 h.