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

Course: JTSK-320111
September 21st, 2017

Assignment 3 - Conditions and Loops

- 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 3.1 *Infinite loop by bad coding* (1 point)

d **Presence assignment, due by 18:30 h today**
The program below prints

```
n is 2
n is 2
...
```

until you stop the execution by pressing `Ctrl-C`. Fix the program such that it prints 2, 3, 4 and 5 as values for `n`.

```
#include <stdio.h>
int main() {
    int n = 2;
    while (n < 6)
        printf("n is %d\n", n);
        n++;
    printf("That's it.\n");
    return 0;
}
```

Problem 3.2 *Divisible by 4 and 7?* (1 point)

Presence assignment, due by 18:30 h today
Write a program, where you can enter an integer from the keyboard. Determine whether the number is divisible by both 4 and 7. Depending on the outcome print on the screen
"`The number is divisible by 4 and 7`" or
"`The number is not divisible by both 4 and 7`".
You can safely assume that the input will be valid.

Problem 3.3 *Categorization of characters* (1 point)

Presence assignment, due by 18:30 h today
Write a program where you can enter a character from the keyboard. Then determine whether the character is a lowercase alphabetic character or not and print a corresponding message on the screen.
You can safely assume that the input will be valid.

Problem 3.4 *Seconds and minutes I* (2 points)

Write a program where you can enter an integer `n` from the keyboard. Then a "conversion table" for 1 to `n` minutes should be printed on the screen as in the example below. Make sure that the integer value you entered for `n` is valid (positive and non-zero). If an invalid integer `n` was entered then repeat the entering until a valid value will be entered.
You must use a `while` loop to print the conversion table. The output has to be precisely formatted as in the example.

```
1 minute is 60 seconds
2 minutes are 120 seconds
3 minutes are 180 seconds
...
```

Problem 3.5 *Seconds and minutes II* (1 point)

In your solution for **Problem 3.4** replace the `while` loop for generating the output by a `for` loop such that the program will have the same functionality.

Problem 3.6 *Writing characters I* (1 point)

Write a program where you first enter a character `ch` and then an integer `n` from the keyboard. Print the character `ch` on the screen `n` times separated by space. Make sure that `n` will have a valid integer value (i.e., cannot be negative). In the invalid case repeat entering `n` until a valid value will be entered.
You can safely assume that the input for `ch` and `n` will be valid regarding the their types.

Problem 3.7 *Writing characters II* (1 point)

Write a program where you first enter an uppercase character `ch` and then an integer `n` from the keyboard. Print the characters corresponding to `ch`, `ch+1`, ..., `ch+n` on the screen separated by newline.
You can safely assume that `ch` is an uppercase character and `n` is greater than 0 and less or equal than 32.

Bonus Problem 3.8 *Writing characters III* (1 point)

Add checks to your solution for **Problem 3.7** such that the program prints corresponding messages and stops the execution (by `return 1` in the `main` function) if `ch` is not an uppercase alphabetic character or if `n` is greater than 32 or less or equal than 0.

Problem 3.9 *Computing the average of grades* (2 points)

Write a program where you first enter from the keyboard a character `c` followed by an integer `n`, and `n` float values representing grades. Use an array for storing the grades. You can assume that not more than 50 grades will be entered. Your program should compute and print the following on the screen: if `c` is 's' then the sum of the grades, if `c` is 'p' then the product of the grades and if another character was entered then the arithmetic mean (or average) of all grades.
You must use a `switch` instruction in your solution.
You can safely assume that the input will be valid.

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
   a3_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.he@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 Tuesday, September 26th, 10:00 h.