UFAZ - Bachelor of Computer Science

System Programming

PW01 : advanced C programming

For each exercice, we expect the student to write a program, compile it and run it without errors of several examples. Test sets and comments are as important as the code itself.

Exercice 1

Write a program which reads a string from the standard input *stdin* using the function `fgets`, copies it (using `strncpy` into another string, and then displays the new string (using the function `puts`) together with its length (using `strlen` and `printf`).

Exercice 2

We want to write a program which reads the name of a month (from the standard input), then prints its number (1 to 12) as well as its number of days. If the month is not valid, the program displays a error message.

To achieve that, we store the valid months as well as the corresponding numbers of days in a structure :

```
struct month {
    char name [9 + 1] ;/* name */
    int days ; /* number of days in the month */
};
```

We assume february always has 28 days.

Declare an (initialized) constant array of 12 structures to store the 12 possible months and write the expected program.

Exercice 3

Strings of characters and constants of type `char` may contain the following elements:

- regular characters, i.e. those which code is between 32 and 126;
- control characters (`\n`, `\r`, `\t`, and `\b`) ;
- non-printable characters such as (`\\`, `\` and `\"`) ;
- other characters whose numerical value is between 0 and 31 or greater that 127. In this case, we represent it using a backslash (`\`) followed by the octal code for this character.

Write a program which reads a string of characters with possibly some special characters and writes in another string the C representation of the initial string (for example, the character `\t` is replaced by 2 characters `\` et `t`).

Exercice 4

Write a C program which performs the converse of exercice 3. It transforms a string of characters, with some special characters as they are represented in C into a string which contains the translated characters (for example, the consecutive characters `\` and `t` are replaced by the character `\t`).