Unit 10: String Idioms and C String Library

Although we have used `char` variables and arrays of `char` values in previous units, we still lack any convenient way to process a series of characters (a string in C terminology). Relevant for this unit is Book Chapter 13 "Strings", which covers both string constants (also called string literals in the C standard) and string variables, which can change during the execution of a program in detail. Relevant for this unit are the following Book Sections

- Section 13.1 explaines rules that govern string literals, including rules for embedding escape sequences in string literals and for breaking long string literals
- Section 13.2 shows how to declare string literals
- Section 13.3 describes ways to read and write strings
- Section 13.4 shows how to write string-handling functions that process strings
- Section 13.5 covers some of the string-handling functions of the C library
- Section 13.6 presents idioms that are often used when working with strings
- Section 13.7 describes how to set up arrays whose elements are pointers to strings of different lengths; and it also explains how supply command-line arguments to progams.

Project u10: One-Year Reminder

A one month reminder program is described in section 13.5 in the Book in detail, and provided as `remind.c`. Improve this provided remind.c program in the following ways:

- Have the program print the error message "The entered day is negative or larger than 31.", followed by a newline, and ignore a reminder if the corresponding day is negative or larger than 31. Hint: Use the continue statement, but be careful where to place it in your code.
- Have the program print a one-year reminder list. Allow the user to enter a month, a day, a 24-hour time, and a reminder. The printed reminder list should be sorted first by month, day, and time.
- Require the user to enter data by prompting "Enter reminder [month/day hours:minutes message]:", followed by a newline, and have the user enter all the data then.
- Month and day can be entered as either a one or two digit number, but must be printed with leading space to be aligned for two digits. Note that the month is not preceded by an extra space (as it is the case in remind.c (!)).
- The time is entered as 24-hour time, where hours and minutes are separated by ':'. The time must be printed with leading zeros to have hours and minutes aligned for two digits.

- The reminder message must be printed exactly as entered.
- The user can enter another reminder until either day or month is entered as 0. It is allowed to only enter 0 as day as well. This is followed by printing all the reminders in sorted order.

Hint: make sure you reserve enough (additional) memory for the characters representing month and time and the additional space characters. There is no extra sorting-step, the list of reminders is kept sorted by putting each new reminder at the right position in the array (see remind.c).

```
% u10_remindmod
Enter reminder [month/day hours:minutes message]:
4/18 9:00 Chinese Grand Prix 2010
Enter reminder [month/day hours:minutes message]:
5/16 14:00 Grand Prix de Monaco 2010
Enter reminder [month/day hours:minutes message]:
5/9 14:00 Gran Premio de Espana Telefonica 2010
Enter reminder [month/day hours:minutes message]:
0

One-Year Reminder
 4/18 09:00 Chinese Grand Prix 2010
 5/ 9 14:00 Gran Premio de Espana Telefonica 2010
 5/16 14:00 Grand Prix de Monaco 2010
%
```

Refer to the provided test-cases and compare the output of your program with the provided output of the test-cases. Your program should behave exactly like the test-cases before you hand it in. Hand in your program as `u10_remindmod.c`.