project[7]: pointers
Due Tuesday, 4/5/2016, 12:59:59pm

Project Goals

The goals of this project are to explore the use of pointers.

Important Notes:

1. **Comments:** Header comments are required on all files and recommended for the rest of the program. Points will be deducted if no header comments are included.

Problem 1

For this project you will write a program that reads in a text message and then prints the message in toggle case: lowercase letters switched to upper case and uppercase letters switched to lowercase. All other characters (digits or special characters should remain the same). The text message should be no longer than 30 characters.

Your program should define and use the following functions:

- `read_message`: the function should take as parameter a character array, which will be filled out while reading from the keyboard and a pointer to an integer that will store the size of the text message. You should use the `getchar` function to read the message one character at a time. This function should be the one that prints out the prompt for the user to enter the message.
- `toggle_case`: the function should take as parameter a pointer to a character in the message array, and update it accordingly (from lowercase to uppercase and reverse).

Additional constraints:
- your program should use pointers, not integers, to keep track of the positions in the message array.

Your program should function as follows:

```
Enter your string (maximum 30 characters): Today is March 29,
2016.
New string: tODAY IS mARCH 29, 2016.
```

Your program should be saved in a file called `toggle.c`.

Challenge: Write a function `count_cv` that counts how many consonants and vowels are in the given string and outputs that to the terminal. The function should take as parameter the character array and should use pointers, not integers, to keep track of the positions in the message array. Your challenge program should be saved in a file called `toggle_challenge.c`.

Problem 2

For this project you will write a program that reads in an integer representing a number of seconds and uses a function called `print_time`, which separates the number of seconds into the equivalent in hours, minutes and seconds.

The `print_time` function should take as parameters a long variable (containing the number of seconds entered by the user), and pointers to three integer variables representing the equivalent in hours, minutes and seconds. The function will use these pointers to ″return″ the hours, minutes, seconds equivalent.

Your program should ask the user to enter a number of seconds to be converted, then use the function to obtain the equivalent in hours, minutes and seconds and then print out these values on the screen.

Your program should function as follows:

```
Enter the number of seconds: 181

The equivalent is: 0 hour(s), 3 minute(s), 1 second(s).
```

Your program should be saved in a file called `convert_time.c`.

Challenge: make your previous program correctly print out hour versus hours, minute vs. minutes and second vs. seconds, depending on whether the amount is 1 (for hour, minute, second) or 0 or larger than 1 (for hours, minutes, seconds). Your challenge program should be saved in a file called `convert_time_challenge.c`.

Submission details

The project needs to be submitted by Tuesday, 4/5/2016, 12:59:59pm.

To submit your project, you will have to save your project files to an ECC machine using the Linux VM or the nomachine client:

- create a directory called “project7”
- save your *.c files in that directory
- save your description file into that directory
- DO THIS ONCE: Install the submission script ***(don’t type the ‘>’ symbols)***
  > cd ~
  > wget http://www.cse.unr.edu/~newellz2/submit
  > chmod +x ./submit
- TO Submit:
  > cd project7
  > ~/submit

The submission script copies all files in the current directory to our directory. You may submit as many times as you like before the deadline, we only keep the last submission.

Academic Honesty

Academic dishonesty is against university as well as the system community standards. Academic dishonesty includes, but is not limited to, the following:

Plagiarism: defined as submitting the language, ideas, thoughts or work of another as one's own; or assisting in the act of plagiarism by allowing one's work to be used in this fashion.

Cheating: defined as (1) obtaining or providing unauthorized information during an examination through verbal, visual or unauthorized use of books, notes, text and other materials; (2) obtaining or providing information concerning all or part of an examination prior to that examination; (3) taking an examination for another student, or arranging for another person to take an exam in one's place; (4) altering or changing test answers after submittal for grading, grades after grades have been awarded, or other academic records once these are official.

Cheating, plagiarism or otherwise obtaining grades under false pretenses" constitute academic dishonesty according to the code of this university. Academic dishonesty will not be tolerated and penalties can include canceling a student's enrollment without a grade, giving an F for the course, or for the assignment. For more details, see the University of Nevada, Reno General Catalog.