

CPE 325: Embedded Systems Laboratory

Laboratory Assignment #8

1. Assignment

[100 pts]

1. **[90 pts]**: Write and test a program in C that functions as defined below

<code>void UART_setup();</code>	Configures UCI to work in the UART mode at the baud rate of 57,600.
<code>void UART_Send_Character(char my_char);</code>	Sends a character via UART
<code>void UART_send_String(char* string);</code>	Sends a string via UART using <code>send_Character(char c)</code>
<code>void UART_get_word(char* buffer, int limit);</code>	Receives characters via UART until it finds the new line character or until the limit of characters is exceeded. Writes that string (excluding the new line character) to the buffer allocated outside of the function. Terminates the string with the null character.

Test your functions to make sure they work properly and none of them writes or reads non-allocated memory. Make sure that `UART_get_word()` inserts the null character at the end of your string, and does not exceed the limit, including the null character.

You will write a program that interacts with the user, providing them with strings to be typed in for practice.

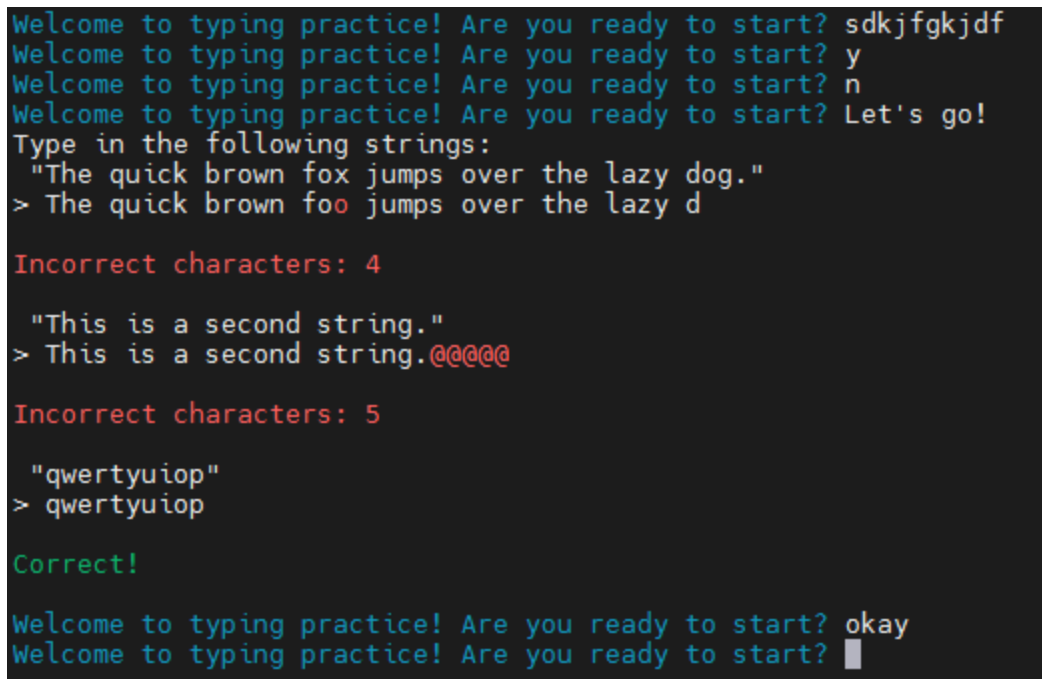
- a. Initially display the following message in HyperTerminal (You may use Putty, realterm, MobaXterm etc).

“Welcome to typing practice! Are you ready to start?”
- b. If the user types in the input **“Let’s go!”**, the program should display **“Type in the following strings:”** and then go to step c. If the user types in any other input, the program should repeat the initial prompt on a new line.
- c. The program should display a string to be typed in, and then allow the user to enter the string on a new line (make sure the user’s input lines up below the output prompt). Once the user presses Enter, the program should check the string to determine if it was entered correctly, or how many characters are incorrect. Inputs should be case-sensitive.
- d. The program should display a message with **“Correct!”** if entered correctly, or with the number of characters they entered incorrectly from the string.
- e. Steps c-d should be repeated for at least three different string prompts. These can be hardcoded in your program.

- f. Once the user has entered all strings, you should go back to displaying the initial greeting message on a new line. The program should be ready to use again without resetting the board (calling `main()` is not a valid solution for this).

Note: Your program must have at least three string inputs for typing. You should demonstrate all inputs during your demo.

The following screenshot demonstrates the functionality of the typing practice program:



```
Welcome to typing practice! Are you ready to start? sdkjfgkjdf
Welcome to typing practice! Are you ready to start? y
Welcome to typing practice! Are you ready to start? n
Welcome to typing practice! Are you ready to start? Let's go!
Type in the following strings:
  "The quick brown fox jumps over the lazy dog."
> The quick brown foo jumps over the lazy d

Incorrect characters: 4

  "This is a second string."
> This is a second string.@@@@@

Incorrect characters: 5

  "qwertyuiop"
> qwertyuiop

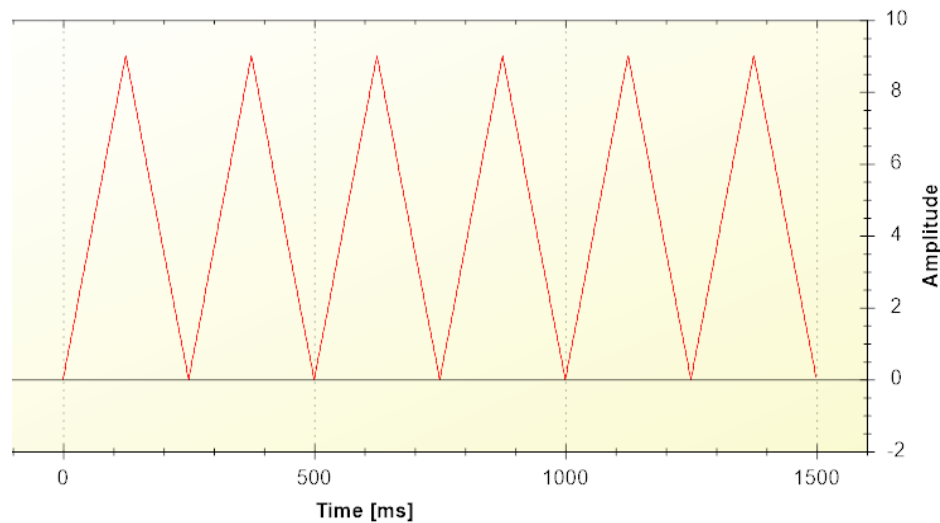
Correct!

Welcome to typing practice! Are you ready to start? okay
Welcome to typing practice! Are you ready to start? █
```

You do not need to include the quotation marks around the typing prompts, or the > indicating the start of user input; however, do make sure that the user input lines up with the prompt. Colors are only necessary for the bonus.

Hints:

- You can use `snprintf()` function to concatenate strings, and convert numbers to strings. Make sure you **do not overflow any buffers** when you use it.
 - Use `strcmp()` function to compare strings
 - Echo characters back if you want to see what you type.
 - Use `\r\n` sequence to properly start a new line.
2. **[10 pts]:** Write a C Program that generates a triangular wave and displays it on the UAH Serial app (using a 57,600 baud connection). The parameters for the triangular wave are as follows: (a) Amplitude 9 Units. (b) Frequency: 4 Hz. This signal will be transmitted to the serial app for a duration of 6 signal periods. Example output is shown below.



2. Bonus

[up to 15 pts]

1. For Part 1, an extra 5 points will be awarded if you print the text in colors. For example: The message "Correct!" can be printed in green text while the message "Incorrect characters" can be printed in red text. The greeting prompt can be printed in blue, etc. You are free to make your own choice but use of multiple colors will be given more value.
2. For Part 1, an extra 10 points will be awarded if you print the incorrectly typed characters in the string in red (see screenshot above for example). Your goal is to highlight the incorrect characters so they are easily visible.

3. Questions to be Addressed.

Please make sure that you have addressed following questions in your demonstration:

1. In a single demonstration, show the full operation of Q1.
2. Explain the timing calculation for the timeout feature of Q1 before results are shown.
3. Explain how the output was generated for Q2 for the specified duration.

4. Topics for Theory

1. Serial Communication and UART
2. UAH Serial App

5. Deliverables

1. Lab report which includes:
 - a. Flowchart for Q1.
 - b. Answers to any questions from the tutorial.
 - c. Screenshots for both Q1 and Q2
 - d. Any calculations required for timing, including the duration of the signal and its frequency in Q2.