Project 3, Program Design

1. Write a C program that will calculate the voltage for an experiment that will be carried out repeated using an ac voltage signal. The signal is to have one of three values, depending on the time since the beginning of the experiment:

   For time t <1, volts(t) = 0.5 x sin (2t).
   For time 1.0 <= t <= 10.0, volts(t) = sin(t).
   For time t > 10.0, volts(t) = sin(10.0).

   1) Name your program *voltage.c.*
   2) The program will calculate the voltage starting at time 0 until the time reaches 12. Increment the time by 0.5 for each *calculation* of the voltage.
   3) Use a for loop.
   4) Use the math library function **sin(x)** to compute sin(x). You will need to include the system header file math.h.
   5) On Unix you will need **–lm** in your command line to tell the Linker to search the math library when you compile.

      ```
      gcc -Wall -lm voltage.c
      ```

   6) The formatted output should look like the following. The time and voltage should display two digits after the decimal point.

time (sec)	voltage
0.00	0.00
0.50	0.42
1.00	0.84
1.50	1.00
2.00	0.91
2.50	0.60
3.00	0.14
3.50	-0.35
4.00	-0.76
4.50	-0.98
5.00	-0.96
5.50	-0.71
6.00	-0.28
6.50	0.22
7.00	0.66
7.50	0.94
8.00	0.99
8.50	0.80
9.00	0.41
9.50	-0.08
10.00	-0.54

10.50	-0.54
11.00	-0.54
11.50	-0.54
12.00	-0.54

2. Write a program that takes a message entered by the user, remove all punctuation (nonletters) except white spaces, bring all letters to uppercase, and then print the message. For example, with the input

   I came, I saw, I conquered!

   the output text should be

   I CAME I SAW I CONQUERED

   1) Name your program `remove.c`.
   2) Assume the input only contains punctuation and letters and does not contain digits.
   3) The user input ends with the user pressing the enter key (a new line character).
   4) You can use character handling functions such as toupper and isalpha. Don't forget to include ctype.h if you use any character handling functions.

Before you submit

1. Compile both programs with –Wall. –Wall shows the warnings by the compiler. Be sure it compiles on ***circe*** with no errors and no warnings.

   gcc -Wall -lm voltage.c

   gcc –Wall remove.c

2. Be sure your Unix source file is read & write protected. Change Unix file permission on Unix:

   chmod 600 voltage.c

   chmod 600 remove.c

3. Test the second program with the shell script try_remove on Unix:

   chmod +x try_remove

   ./try_remove

4. Submit voltage.c and remove.c on Canvas.

Grading

Total points: 100 (50 point each problem)

1. A program that does not compile will result in a zero.
2. Runtime error and compilation warning 5%
3. Commenting and style 15%
4. Functionality 80%

Programming Style Guidelines

The major purpose of programming style guidelines is to make programs easy to read and understand. Good programming style helps make it possible for a person knowledgeable in the application area to quickly read a program and understand how it works.

1. Your program should begin with a comment that briefly summarizes what it does. This comment should also include your **name**.
2. In most cases, a function should have a brief comment above its definition describing what it does. Other than that, comments should be written only *needed* in order for a reader to understand what is happening.
3. Variable names and function names should be sufficiently descriptive that a knowledgeable reader can easily understand what the variable means and what the function does. If this is not possible, comments should be added to make the meaning clear.
4. Use consistent indentation to emphasize block structure.
5. Full line comments inside function bodies should conform to the indentation of the code where they appear.
6. Macro definitions (#define) should be used for defining symbolic names for numeric constants. For example: **#define PI 3.141592**
7. Use names of moderate length for variables. Most names should be between 2 and 12 letters long.
8. Use underscores to make compound names easier to read: **tot_vol** or **total_volumn** is clearer than totalvolumn.