CMSC246 Systems Programming - Lab 2

C data types and limits. atoi, palindromes.

1. Write a C program to figure out how many bytes are used by the following types on your computer/compiler:

   - `short`
   - `int`
   - `long`
   - `char`
   - `float`
   - `double`

   Confirm your answer by looking at the contents of file: `/usr/include/limits.h`

2. Write a for loop to push to the limit of the values `int` variables can take. From (1) you determined the number of bits an `int` variable takes. Computers use 2's complement representation. Recall in 2's complement representation, the largest positive integer that can be stored in x bits is the value 2^{x-1}-1 and the smallest value is -2^{x-1}.

3. Let's assume the range is for an int is [LOW .. HIGH].

   ```
   #define HIGH xxxx
   int start = HIGH-5, end = HIGH;
   for (int i = start; i < end; i++)
      printf("i = %d\n", i);
   ```

   Fill in the value of `HIGH` for "xxxx" that you obtained in step 2. Run the program.

   Describe the output…

   Next, change the loop as shown below (we have changed the < to <=):

   ```
   for (int i = start; i <= end; i++)
   ```

   Run the program again. What happens?

   Make sure you understand what happens and why.

4. Become more adept with Emacs. Try some of the things on the quick reference sheet.

5. Write a C function to convert user input to an integer without using scanf or any function that converts strings to integers. You function should:
    a. accept using input using the getchar() function
    b. build an integer until receiving a non-digit input character. For example, given the input 132e the function would return 132
6. Create / customize you .bashrc file.
    a. Add "." to your PATH
    b. create an alias to do something at least semi-useful
    `alias coffee=‘echo "I need caffeine" ‘`
    is NOT useful
7. Write a palindrome recognizer. The program should:
    a. use getchar for reading input from a user
        i. read characters until getting a CRLF
    b. Determine if the string entered is a palindrome case insensitively (without simply down casing the input)
    c. print out something like "erE is a palindrome"
    d. Return to a.
    e. The program only ends when killed by CTRL-c