COP 3223 Recitation: Strings

Problem: Secret Communications

Due to all of your various associations, you'd like to communicate with your roommate in confidence via email. You have a rather simple scheme to express strings of letters as numbers. You realize that since there are only 26 different letters, a string of letters can be viewed as a number in base 26! For example, consider the word

`COMPUTER`

You realize that we can just assign numeric values to the letters with A = 0, B = 1, C = 2, …, Z = 25 and then treat each letter location as the ones place, the 26 place, the 26^2 place and so forth. Thus, the value of `COMPUTER` would be:

$2x26^7 + 14x26^6 + 12x26^5 + 15x26^4 + 20x26^3 + 19x26^2 + 4x26^1 + 17x26^0$.

If someone knew this scheme, it's quite easy to uncover the underlying text, since no password is being used to hide the information. But, over the course of the semester you've realized that most of your fellow classmates aren't numerically inclined, so you are confident that this simple scheme will work.

You've decided to write two functions to aid in encoding and decoding your messages. One function takes in a string and its length, and returns a number (stored in a data type called `long long`) equivalent to the string using the system described above. The second function will take in a number and the length of the corresponding word and print out the corresponding string that is the result of decoding the first number.

You can test your functions by filling in the scaffold code which is set up to work in two modes:

1) Read an input file of words (in uppercase letters) and output to the screen the corresponding numbers (with the lengths of the converted words).

2) Read an input file of number pairs (number and length of word) and output to the screen the corresponding message.

To store the integers needed for the calculation, please use the type `long long`. This will ensure that the scheme is valid for strings of length 13 or fewer characters.

Also, a simple process to calculate the necessary sum is something called Horner's method. Instead of calculating any exponents, utilize the following idea for each letter in the string word:

```
long long value = 0;
int i;
for (i=0; i<length; i++)
    value = 26*value + (word[i] - 'A');
```

At the end of this code, value will equal the corresponding calculation shown above. Notice that when we subtract the Ascii value of an uppercase letter from the Ascii value of 'A', we naturally get the appropriate numerical value from 0 to 25 described above.

Finally, when attempting to reverse this process, note that we can repeatedly mod by 26 and divide by 26 to peel off each letter from the end, one by one. Rather than provide the code, a small example will be illustrated below:

Consider the number 1371, knowing the length of the word it encoded was 3:

1371 % 26 = 19 (T), 1371 / 26 = 52

52 % 26 = 0 (A), 52 / 26 = 2

2 % 26 = 2 (C) 2 / 26 = 0

The corresponding message was "CAT", reading the letters from the bottom to the top.

Note that to take an integer, x, in between 0 and 25, here is how you can store the corresponding uppercase letter in the character array word, index i:

```
word[i] = (char)(x + 'A');
```

Don't forget to null terminate your string after you create it. Here are the function prototypes for the two functions you'll have to complete for this assignment. Please write your code in the file, `secret-scaffold.c`.

```
long long convert(char word[], int length);

void printText(long long value, int length);
```

Note: if you run the correct program, choose (1) and encode the file msg.txt, your output should be similar to that shown in msg.out. If you run the correct program, choose (2) and decode the file msg.out, your output should be similar to that shown in msg.txt.