String Basics

Strings

1. Have the user input a string of characters. Count the number of characters entered, then display the first and last characters of the string.
2. Given 5 strings entered one-at-a-time by the user, determine the total number of letters entered, as well as the longest and shortest words.
3. Read a string of characters and a positive integer, *n*, from the user. Generate *n* random characters, each taken from the string.
4. *Card ID*: A playing card can be represented using two characters. The first character represents the *rank*, while the second represents the *suit*. The strings 3H, QD and TS represent the three of hearts, queen of diamonds, and ten of spades respectively. Given a two-character string, determine whether it is a valid card or not and, if it is, display its full name (e.g. “Three of hearts”).
5. For a user-entered string, display the decimal ASCII value for each character.
6. Given a string, determine whether it is a palindrome, i.e. reads the same forward or backward, like *racecar* or *toot*.
7. Determine the number of positive integers below 1,000,000 that are palindromes. Hint: try converting each integer to a string.
8. *ROT-13*: A simple encryption scheme is ROT-13, whereby each character is replaced by one that occurs 13 characters later in the alphabet, wrapping back to the beginning if necessary. For instance, A becomes N, B becomes O, …, and Z becomes M. The string PYTHON would be encoded as CLGUBA using this scheme. Write a program that uses the `chr` and `ord` functions to encode a string using ROT-13. Note: this program can also act as a ROT-13 *decoder*. Tbbq yhpx!