Problems on application of String class

1. Write an application involving switch-case-default that performs one of the following actions:
(i) reads a line of text and outputs the text twice—once in all uppercase letters and once in all lowercase letters;
(ii) reads a line of text and a search character and uses String method indexOf to determine the number of occurrences of the character in the text;
(iii) reads a line of text, tokenizes the line using space characters as delimiters and outputs only those words beginning with the letter "b";
(iv) reads a line of text, tokenizes it using space characters as delimiters and outputs only those words ending with the letters "ED".
(v) reads a line of text, tokenizes the line with String method split and outputs the tokens in reverse order. Use space characters as delimiters.

2. Write an application that inputs an integer code for a character and displays the corresponding character. Modify this application so that it generates all possible three-digit codes in the range from 000 to 255 and attempts to print the corresponding characters.

3. Write an application that uses random-number generation to create sentences. Use four arrays of strings called article, noun, verb and preposition. Create a sentence by selecting a word at random from each array in the following order: article, noun, verb, preposition, article and noun. As each word is picked, concatenate it to the previous words in the sentence. The words should be separated by spaces. When the final sentence is output, it should start with a capital letter and end with a period. The application should generate and display 20 sentences.
The article array should contain the articles "the", "a", "one", "some" and "any"; the noun array should contain the nouns "boy", "girl", "dog", "town" and "car"; the verb array should contain the verbs "drove", "jumped", "ran", "walked" and "skipped"; the preposition array should contain the prepositions "to", "from", "over", "under" and "on".

4. Write an application that encodes English-language phrases into pig Latin. Pig Latin is a form of coded language. There are many different ways to form pig Latin phrases. For simplicity, use the following algorithm:

To form a pig Latin phrase from an English-language phrase, tokenize the phrase into words with String method split. To translate each English word into a pig Latin word, place the first letter of the English word at the end of the word and add the letters “ay.” Thus, the word “jump” becomes “umpjay,” the word “the” becomes “hetay,” and the word “computer” becomes “omputercay.”
Blanks between words remain as blanks. Assume the following: The English phrase consists of words separated by blanks, there are no punctuation marks and all words have two or more letters.
Method printLatinWord should display each word. Each token is passed to method printLatinWord to print the pig Latin word. Enable the user to input the sentence. Keep a running display of all the converted sentences in a text area.

5. Write an application that inputs a telephone number as a string in the form (555) 555-5555. The application should String method split to extract the area code as a token, the first three digits of the phone number as a token and the last four digits of the phone number as a token. The seven digits of the phone number should be concatenated into one string. Both the area code and the phone number should be printed. Remember that you’ll have to change delimiter characters during the tokenization process.