CS101- Algorithms and Programming I
Lab 05

Lab Objectives: `static` methods.

For all labs in CS 101, your solutions must conform to the CS101 style guidelines (rules!)

Notes:

- You should not use Collections, regular expressions or non-CS101 tools in your solution.
- For each of the methods below, you should include a Javadoc comment. The comment should have the following format:

```
/** * This is a simple description of the method. . .
* @param name and description
* @return name and description
*/
```

The following will be in the script, `Lab06_yourname.java`:

PART A (Problem Description):
The soundex system encodes words into a letter followed by 3 numbers that describe (roughly) the sound of the word. Similar sounding words are encoded with the same 4-character codes. The (simplified) algorithm is as follows:

Assuming the word contains only alphabetic characters, the encoding steps are as follows:

1. The first letter of the Soundex code is the first letter of the string being encoded.
2. After the first letter in the string, do not encode **vowels** or the letters H, W and Y. These letters may affect the code by being present but are not encoded directly.
3. Assign a numeric digit between one and six to all letters after the first using the following map pings:
    - 1: B, F, P or V
    - 2: C, G, J, K, Q, S, X, Z
    - 3: D, T
    - 4: L
    - 5: M, N
    - 6: R
4. Where adjacent (side by side) digits are the same, remove all but one of those digits unless a vowel, H, W or Y was found between them in the original text. POP -> both Ps are encoded because there is a vowel between.
5. Force the code to be four characters in length by either padding with zeros or by truncation (dropping extra characters) where necessary.

PART B:

Your solution should make use of the following functions:

a) `isAlphaWord()` : takes a word as a parameter and returns true if all characters are alphabetic, false if not.

b) `getCode()` : takes a character as a parameter and returns the integer number representing the given character (see step c from part A). If the character is not one of the coded characters, return -1.

c) `buildCode()` : takes word as a parameter and returns a new word where each character has been changed to its corresponding code value. If no code value exists (vowels or hwy), concatenate the original character.

d) `removeAdjacentDuplicates()` : takes a String as a parameter, and for all (adjacent/side by side) repeating characters, keeps the first but removes the rest. For example, when passed the string '11233w3' returns '123w3'.

e) `removeLetters()` : takes 2 strings as parameters, the first is the string to update, and the second is a string containing the set of letters to remove. The function should return a new string where all letters that appear in the given set of letters have been removed.

f) `padCode()` : if the code passed as a parameter has fewer than 4 characters, pads the remaining spaces with zeros. If the code has more than 4 characters, returns the first 4 characters as a string.

g) `getSoundex()` : takes a word as a parameter and returns its soundex encoding according to the rules listed in part A. Return an empty string if there are any non-alpha characters in the word.

PART C:

Write a program that inputs a word from the user and outputs its soundex encoding (or error messages) where appropriate. You can validate your codes using the lookup tool found here.

Notes: the first letter may be o

Sample Run:

```
Enter a string: hi123
Characters must be alphabetic...

Enter a string: carrot
Soundex: C630

Enter a string: Caret
Soundex: C630

Enter a string: McLaren
Soundex: M246

Enter a string: MacLauren
Soundex: M246

Enter a string: hi
Soundex: H000

Enter a string: high
Soundex: H200

Enter a string: long
Soundex: L520

Enter a string: Llama
```

```
Soundex: L500


Enter a string: under
Soundex: U536

Enter a string: assist
Soundex: A223

Enter a string: exit
Soundex: E230

Goodbye!
```