Unit 1 - APCSA Programming Program
Password Enhancer Project

Problem Statement: Develop an algorithm for enhancing the strength of a password (represented as a `String` of characters) given specified best practices in creating strong passwords (explained below) and strategic use of methods of the `String` class. Then, implement that algorithm in a method within a Java program.

Overview:
Currently many websites provide suggestions to account holders about some widely-recognized best practices in creating a strong password. Two of these best practices are worth specific focus:

- Use a mix of letters, numbers, and symbols in your password.
- Don't use personal information or common words as passwords.

One strategy that many security-savvy web users implement is to replace specific letters in a password with a (somewhat) similar-looking character. For instance, the password "`spookyhalloween`" could be enhanced by replacing every "`o`" with "`0`" (zero), every "`a`" with "`@`", every "`e`" with "`3`", and every "`l`" with "`!`", so that the enhanced password would read as "`sp00kyh@!!0w33n`". (You might find it interesting to investigate the strength of each password, as well as your own passwords, at https://howsecureismypassword.net/.)

For this project, let's use this same strategy by replacing every vowel in a given password with another character. Specifically:

Vowel	Replacement Character
a	@
e	3
i	!
o	0 (zero)
u	^

NOTE: We are ONLY replacing lower case vowels. You may leave upper case vowels unchanged.

Program REQUIREMENTS:

1. The `main` method should prompt the user to input a password, call the helper method to enhance the password and print out the new improved password.
2. The `main` method should continue to prompt the user to input passwords until the user inputs "-999". (We will assume in the spirit of this project that the user has no interest in having this four-character password, which would take a computer 10 microseconds to crack.)
3. The code for strengthening the password MUST be done in a separate helper method other than `main`.
4. The helper method must be named `enhancePassword`.
5. The helper method takes a single parameter representing the old password.
6. The helper method must return the enhanced password (a `String`)
7. In designing your program, you are restricted to using only the methods of the `String` class that we have discussed in class (those found in the APCSA Java Subset)