Assignment#1 Warm-up and string methods

1. Write a program that takes your full name as input and displays the abbreviations of the first and middle names but not the last name which is displayed as it is. For example, if your name is Robert Brett Roser, then the output should be R.B.Roser.

   Hint: use the `.split()` method!

2. You are asked to ensure that the first and last names of people begin with a capital letter in their passports.
   For example**, alison heck** should be capitalized correctly as **Alison Heck**.
   Given a full name, your task is to *capitalize* the name appropriately.

3. Given 2 strings (input by a user), return a *new* string made of the first, middle and last character of *each* input string.

   Example: "jeffery" and "baseballs" output: "jbfbys"

   Note: instruct the user that each word must have 3 or more characters and have an odd number of characters

4. Given two strings from a user, S1 and S2, create a new string that is a mix of both. The new string made of the first character of S1, then the last character of S2, then second character of S1, then the second to last character of S2, and so on. Any leftover chars go at the end of the result.

   Example and Expected Outcome:
   "Pynative", "Website" = PeytniastbievWe

5. Given an input string, count occurrences of all characters within a string
   For Example:
   "pynativepynvepynative"= {'p': 3, 'y': 3, 'n': 3, 'a': 2, 't': 2, 'i': 2, 'v': 3, 'e': 3}

6. Given a list, slice it into a 3 equal chunks and over each list. Inform the user that the length of items in your list must be evenly divisible by 3

   For Example: sampleList = [11, 45, 8, 23, 14, 12, 78, 45, 89]

   Output: sec1=[11,45,23], sec2=[23,14,12], sec3=[78,45,89]

7. From **MIT Python Open Course** Second Problem Set:

   Create the classic word game ***Hangman***.

   - The computer must select a word at random from the list of available words you have created.

   - The game is interactive; the user inputs their guess and the computer either:

   a. reveals the letter if it exists in the secret word
   b. penalize the user and updates the number of guesses.

   - The game ends when either the user guesses the secret word, or the user runs out of guesses.

- Your program must include at least two (user created) functions:

  For example: `player guess(), guess_check(), is_game_over()`

If you have already created a Hang man game as a past project then try the following…or ***do both!*** (fun):

8.

Americans spell differently from Canadians. Americans write "neighbor" and "color" while Canadians write "neighbour" and "colour".

Write a program to help Americans translate to Canadian.

Your program should interact with the user in the following way. The user should type a word (not to exceed 64 letters) and if the word appears to use American spelling, the program should echo the Canadian spelling for the same word. If the word does not appear to use American spelling, it should be output without change. When the user types "quit!" the program should terminate.

The rules for detecting American spelling are quite naive: If the word has more than four letters and has a suffix consisting of a consonant followed by "or", you may assume it is an American spelling, and that the equivalent Canadian spelling replaces the "or" by "our". Note : you should treat the letter "y" as a vowel.

Keyboard input and screen output is expected.

Sample session. *User input in italics.*

```
Enter words to be translated:
color
colour
for
for
taylor
taylour
quit!
```