Function Scope and Recursion

Part 1 Variable Scope

For the given code below, what are the outputs printed?

<table>
<tr><th>Code</th><th>Output</th></tr>
<tr><td><pre>x = 0
def foo_printx():
      print(x)
foo_printx()
print(x)</pre></td><td></td></tr>
<tr><td><pre>x = 0
y = 999
def foo_printx(y):
      print(y)
foo_printx(x)
print(x)</pre></td><td></td></tr>
<tr><td><pre>x = 0
def foo_printx():
      x = 999
      print(x)
foo_printx()
print(x)</pre></td><td></td></tr>
</table>

Part 2 Nested Functions

<table>
<tr><th>Expressions</th><th>Output</th></tr>
<tr><td><pre>x = 1
y = 2
def foo(y):
      def bar(x):
            return x+y
      return bar(y)
print(foo(x))</pre></td><td></td></tr>
<tr><td><pre>x = 1
y = 2
def foo(x):
      def bar(x):
            return x+y
      return bar(y)
print(foo(x))</pre></td><td></td></tr>
</table>

Part 3 Recursion

Recap from previous tutorial that we can create a customized burger. We have the price list for each ingredient reproduced below for convenience:

Ingredient	**Price**
'B' stands for a piece of bun	\$0.5
'C' stands for cheese	\$0.8
'P' stands for patty	\$1.5
'V' stands for veggies	\$0.7
'O' stands for onions	\$0.4
'M' stands for mushroom	\$0.9

You task last week was to write a function `burgerPrice(burger)` to take in a string of customized burger and return a total sum for the price. Your current task is to write the same function `burgerPrice(burger)` in recursion.

Part 4 Recursion vs Iteration

A. Sum Problem: Given a positive number n, the sum of all digits is obtained by adding the digit one-by-one. For example, the sum of 52634 is 5 + 2 + 6 + 3 + 4 = 20. Write a ***recursive*** and ***iterative*** function `sum(n)` to compute the sum of all the digits in n. You may assume that `n > 0`.

B. Factorial: Given a positive number n, the value of factorial of n (written as $n!$) is defined as $n! = n \times (n-1)!$. Additionally, the value of $0!$ is 1. Write a ***recursive*** and ***iterative*** function `fact(n)` that computes the value of `n!`.

Part 5 Challenge

A. Final Sum: Given a positive number n, the final sum is obtained by repeatedly computing the sum of n until the sum is a single digit. For example, `sum(52634) = 20`, which is not a single digit. We then continue with `sum(20) = 2`. Therefore, `final_sum(52634) = 2`. Write a ***recursive*** and ***iterative*** function `final_sum(n)` to compute the final sum of n.

B. Euler Constant: The value of e^x can be approximated using the formula $e^x = \frac{x^0}{0!} + \frac{x^1}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots$. Write a ***recursive*** and ***iterative*** function `find_e(x, n)` to find the approximation of e^x up to $n+1$ steps. In other words, the last value in the approximation will be $\frac{x^n}{n!}$.