-- assert and try/except

# Section/Exercise: assert

# Write a function, double_your_num, that asks the user
# to enter an integer between 1 and 10, and then
# prints the number * 2.
# Check that the input value is valid (type and range).
# Test your code with correct and incorrect input values.

def double_your_num():

#------------------------------------------

# Section/Exercise: try/except

# Given the following Python code
# to increment the first n integers
# in a list of integers by one:

def incr_li(li, n):
    '''(li: list of ints, n: int) -> None

    Increment the first n values in
    a list by 1 and print the new list.
    
    >>> incr_li([1, 2, 3, 4], 2)
    [2, 3, 3, 4]
    >>> incr_li([1, 2, 3, 4], 5)
    [2, 3, 4, 5]
    ...'''

    newli = li[:]  # slicing the entire list creates a copy
    for i in range(n):
        newli[i] += 1
    print(newli)

    return
# Revise the function to include
# try/except to catch the issue with
# the second example in the docstring.

#==========================================================
# Section/Exercise: Generate a ZeroDivisionError:
# Write Python code to catch a zero
# division error and print a msg:
# Denominator cannot be 0.

#==========================================================
# Section/Exercise: Generate a KeyError:
# Write Python code to catch a key
# error and print a message:
# <incorrect value> is not a key.

#==========================================================
# Section/Exercise: Generate a FileNotFoundError:
# Write Python code to catch a
# file not found error and print
# a message:
# Where is <filename>??

#==========================================================
def foo(x):
    """
    x = x * 2
    return x
    """

def bar(x):
    """
    foo(x); print(x)
    y = foo(x); print(x, y)
    x = foo(x); print(x, y)
    return
    """

>>> x = 4
>>> bar(x)
??
>>> x
??
>>> y
??

#==========================================================

def foo2(x):
    """
    x.append(99)
    return
    """

def bar2(x):
    """
    foo2(x); print(x)
    y = foo2(x); print(x, y)
    y = x
    foo2(x); print(x, y)
    return
    """

>>> x = [4]
>>> bar2(x)
??
>>> x
??
>>> y
??