Assignment Description  
1. Readme/Prompts

This program tests a recursive function that prints the elements of a sequence. It traces the argument on each recursive call and demonstrates the hidden costs of recursion.  
  
Q: Does this function work as expected? If so, how does it works, and describe any hidden costs in running it.  
  
A: The output is as expected yes, but with a significant caveat. Each time a recursive call is made, it creates a new slice of the sequence, which takes up extra memory. The more recursive calls there are, the more memory is needed to store those slices.  
  
Q: Is recursion actually a loop?:  
  
A: Recursion isn’t exactly the same as a loop, though both can be used to get similar results. With recursion, a function keeps calling itself to repeat a process, and a loop directly repeats a piece of code. Both tools can solve problems, but recursion tends to better for specific cases, like tree traversal, while loops are usually more efficient for straightforward repetitious tasks.  
  
Q: Do developers have to use recursion or is it just an option?:  
  
A: Recursion is definitely just an option. It’s not something you have to use. While recursion can be a better solution for specific problems. In many cases, you can achieve the same result using loops, which usually are more efficient in terms of memory and performance. Recursion can lead to issues like stack overflow if not utilized properly, especially with deep or infinite recursions (Thus the whole point of this assignment). So, while it’s a powerful tool to have in your toolkit, it’s important to determine whether recursion is the best fit for the problem at hand, or if a loop might be a simpler, more efficient solution.

2. Source Code of All Files

"""

Author: Paul Sommers

Date written: 11/18/2024

Assignment: Exercise 7-5 - Recursive Function Test

Short Desc: This program tests a recursive function that prints the elements of a sequence.

It traces the argument on each recursive call and demonstrates the hidden costs of recursion.

"""

# Recursive function to print all elements in a sequence

def printAll(seq):

if seq:

# Trace the current sequence and print the first element

print(seq, "->", seq[0])

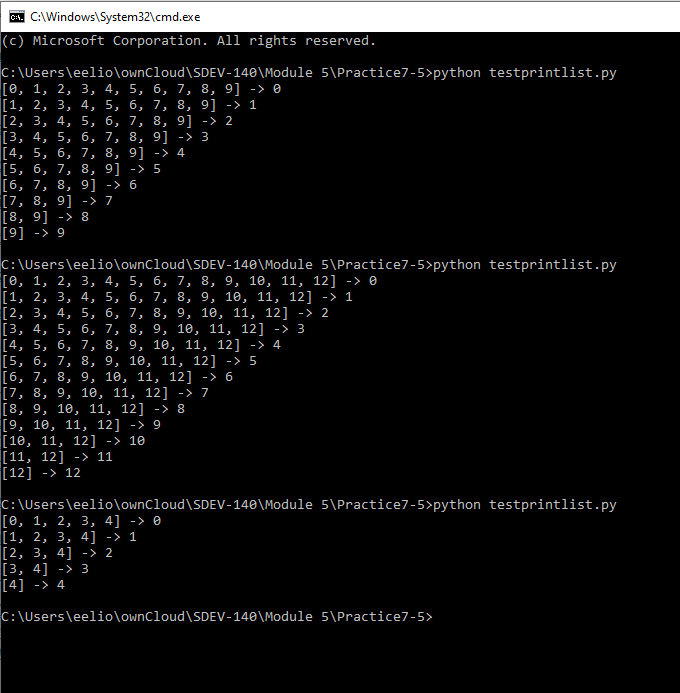
# Recursive call with the rest of the sequence (sliced from index 1)

printAll(seq[1:])

# Test the function

printAll([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])

3. Three Use Case Screen Shots



4. GitHub Url  
  
<https://github.com/PaulSommers/SDEV140-Practice7-5>