# Module 6.3 Function Scope: Coding Questions with Hints

## Question 1: Explain the difference between local and global variables with an example.

Hint: A local variable is defined inside a function and only accessible within it, whereas a global variable is accessible throughout the program.

## Question 2: How can you access a global variable inside a function?

Hint: Use the global keyword before the variable name to access a global variable inside a function.

## Question 3: Write a function that modifies a global variable.

Hint: First, declare the variable as global inside the function, then modify it.

## Question 4: What happens if you try to access a local variable outside its function?

Hint: A local variable cannot be accessed outside its function, resulting in a NameError.

## Question 5: Can a function access a variable defined in another function?

Hint: Generally, a function cannot access a variable defined in another function directly. You'd need to pass it as an argument or declare it global.

## Question 6: Create a function with a parameter that shadows a global variable. Explain the effect.

Hint: The parameter will take precedence over the global variable inside the function, 'shadowing' the global variable.

## Question 7: Discuss how to use a variable defined in one function within another function.

Hint: You can return the variable from the first function and pass it as an argument to the second.

## Question 8: Define a function that declares a new variable with the same name as a global variable. What is the scope of each?

Hint: The new variable will be local to the function, and the global variable will remain unaffected outside the function.

## Question 9: How can you modify a list defined outside a function within that function without using the global keyword?

Hint: Lists are mutable, so you can modify them directly by calling methods or changing items by index.

## Question 10: Explain the concept of closure in Python with an example of a nested function accessing a non-global variable from the enclosing function.

Hint: A closure allows the nested function to remember the state of its environment when called, thus accessing variables of the enclosing scope.