Functions

Introduction to Functions

What is a Function?

In programming, a **function** is a named block of code that performs a specific task or operation.

What is a Function?

In programming, a **function** is a named block of code that performs a specific task or operation.

Think of a function as a mini-program within your larger program. Just like you can break down a complex task into smaller subtasks, functions allow you to break down your code into smaller, manageable pieces.

Introduction to Functions

- **Functions** are the main mechanism for abstraction of statements and expressions in Lua.

Introduction to Functions

- **Functions** are the main mechanism for abstraction of statements and expressions in Lua.

- They can both carry out a specific task or compute and return values. In the first case, we use a function call as a statement; in the second case, we use it as an expression.

```
print(5*6, 6/5)
a = math.sin(2) + math.cos(10)
print(os.date())
```

Introduction to Functions

- In Lua, functions are considered "first-class citizens," meaning they can be assigned to variables, passed as arguments, stored in tables, and treated like any other data type.

Call Functions

- A list of arguments enclosed in parentheses denotes the call; if the call has no arguments, we still must write an empty list () to denote it. There is a special case to this rule: if the function has one single argument and that argument is either a literal string or a table constructor, then the parentheses are optional.

```
print "Lost Island" <--> print("Lost Island")
dofile 'hi.lua'       <--> dofile ('hi.lua')
```

Function Definition

- As we saw in other examples, a function definition in Lua has a conventional syntax, like here:

```
function calculateSum(sequence)
    local sum = 0
    for i = 1, #sequence do
        sum = sum + sequence[i]
    end
    return sum
end
```

Function Definition

- In Lua, it is possible to call a function with a different number of arguments compared to the number of parameters specified in the function definition. Lua handles this situation by adjusting the number of arguments to match the number of parameters. If there are more arguments than parameters, Lua simply discards the extra arguments. Conversely, if there are more parameters than arguments, Lua assigns the additional parameters the value nil.

Function Definition

- Consider we have the function:

```
function f (x, y) print(x, y) end
```