```
        print("Break out of for loop")
        break
```

```
'Output': The variable val is: 0
     The variable val is: 1
     The variable val is: 2
     The variable val is: 3
     The variable val is: 4
     The variable val is: 5
     The variable val is: 6
     Break out of for loop
```

The continue statement skips the next iteration of the loop to which it belongs, ignoring any code after it.

```
a = 6
while a > 0:
    if a != 3:
        print("The variable a is:", a)
    # decrement a
    a = a - 1
    if a == 3:
        print("Skip the iteration when a is", a)
        continue
```

```
'Output': The variable a is: 6
     The variable a is: 5
     The variable a is: 4
     Skip the iteration when a is 3
     The variable a is: 2
     The variable a is: 1
```

Functions

A function is a code block that carries out a particular action (Figure 9-5). Functions are called by the programmer when needed by making a **function call**. Python comes pre-packaged with lots of useful functions to simplify programming. The programmer can also write custom functions.

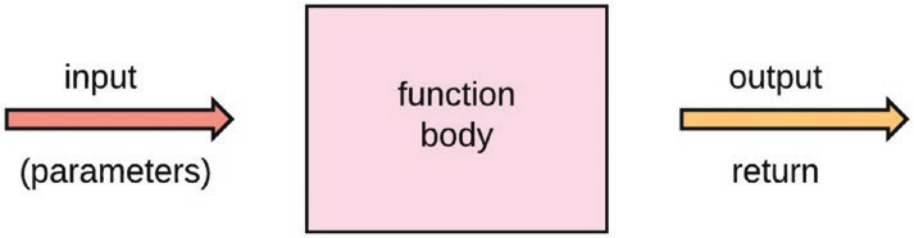

Figure 9-5. *Functions*

A function receives data into its parameter list during a function call. The inputed data is used to complete the function execution. At the end of its execution, a function always returns a result – this result could be 'None' or a specific data value.

Functions are treated as first-class objects in Python. That means a function can be passed as data into another function, the result of a function execution can also be a function, and a function can also be stored as a variable.

Functions are visualized as a black box that receives a set of objects as input, executes some code, and returns another set of objects as output.

User-Defined Functions

A function is defined using the def keyword. The syntax for creating a function is as follows:

```
def function-name(parameters):
    statement(s)
```

Let's create a simple function:

```
def squares(number):
    return number**2

squares(2)
'Output': 4
```

Here's another function example:

```
def _mean_(*number):
    avg = sum(number)/len(number)
    return avg

_mean_(1,2,3,4,5,6,7,8,9)
'Output': 5.0
```