

# Modules and Functions

In this chapter, we will talk about modules and functions.

## WE'LL COVER THE FOLLOWING



- Modules
  - Module Creation
  - Use a Module
- Functions
  - Non-Parameterized Function
  - Parameterized function
  - Parameterized function with a return statement

## Modules #

A function is a block of code that is used to perform a single action. A module is a Python file containing a set of functions and variables of all types (arrays, dictionaries, objects, etc.) that you want to use in your application.

## Module Creation #

To create a module, create a python file with a `.py` extension.

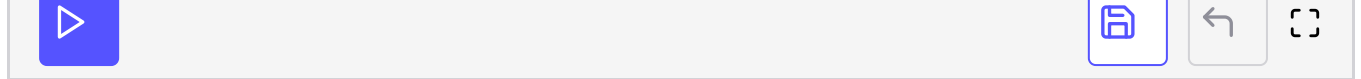
## Use a Module #

Modules created with a `.py` extension can be used with an `import` statement.

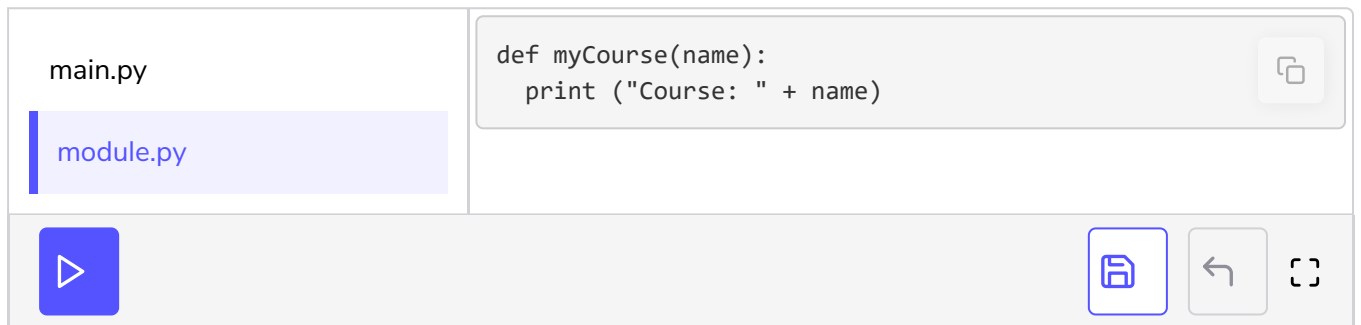
Start up your Python REPL, and let's use the “math” module that provides access to mathematical functions:

```
import math
print math.cos(0.0)
print math.radians(275)
```





Now, let's create a module with a function and use the import statement to call the function:



## Functions #

A function is a block of code that contains a sequence of instructions that are executed when the function is invoked. Data passed in the functions are known as function parameters.

### Non-Parameterized Function #

A function that does not contain parameters is a non-parameterized function.

The following defines the “do\_hello” function that prints two messages when invoked:



**Note:** Make sure that you insert a tab before both print expressions in the previous function. Tabs and spaces in Python are relevant and define that a block of code is somewhat dependent on a previous instruction. For instance, the print expressions are “inside” the “do\_hello” function and, therefore, must have a tab.

### Parameterized function #

Functions that can receive parameters are parameterized functions.

The following Python code uses an “add\_one()” function that receives a parameter ‘val’,and it prints the incremented value inside the function:

```
def add_one(val):  
    print "Function got value",val+1  
    return  
add_one(1)
```



## Parameterized function with a return statement #

Functions can also receive parameters and return values (using the “return” keyword).

The following Python code uses an “add\_one()” function that receives a parameter ‘val’; it prints the value passed in the function, and returns the incremented value:

```
def add_one(val):  
    print "Function got value", val  
    return val + 1  
value = add_one(1)  
print value
```



Now that the concept of modules and functions in python is clear, let’s check your knowledge in the upcoming exercises before moving on to the ‘Recursion’ lesson.