04 - Functions

**Function calls**

```python
>>> type(42)
<class 'int'>
```

```python
>>> int('32')
32
```

```python
>>> int(3.999)
3
```

```python
>>> int(-2.3)
-2
```

```python
>>> float(32)
32.0
```

```python
>>> str(45)
'45'
```

**Math module**

First you must import it.

```python
>>> import math
```

This creates a `math` module object that you can then use.

```python
>>> math
<module 'math' (built-in)>
```

```python
>>> radians = 0.7
>>> height = math.sin(radians)
```

**Function definition**

```python
def func_name(parameter_list):
    statements
    <empty line>
function header
    start of body (indented 4 spc)
    end of body
```
```python
>>> def print_lyrics():
...    print("I'm a lumberjack, and I'm okay.")
...    print("I sleep all night and I work all day.")
...
This definition creates a function object.

>>> print(print_lyrics)
<function print_lyrics at 0x11053fb80>

>>> type(print_lyrics)
<class 'function'>

>>> print_lyrics()
I'm a lumberjack, and I'm okay.
I sleep all night, and I work all day.

You can now call the function from inside another function if so desired.

```python
>>> def repeat_lyrics():
...    print_lyrics()
...    print_lyrics()
...

Ex1: lumberjack.py
Use VSC to type the following script.
Use the IT (^`) to run it:

$ python lumberjack.py
```
```
Parameter and Arguments

```python
>>> def print_twice(value):
...   print(value)
...   print(value)
...   print(value)
   >>> print_twice('Hello world."
```

value is the parameter and msg and 'Hello world' are both arguments.

Arguments are copied into their respective parameters.

Keyword arguments

Keyword arguments allow for the easy identification of each parameter by name. You use an argument keyword to make it clear which parameter is used in the call.

```python
>>> print_twice(value = 'Hello world')
```

Return value

Every function returns a value, its type depended on the value returned in a return statement. If the function does not have a return statement, it returns None.

```python
>>> return_value = print_twice('What?"
What?
What?

>>> print (return_value)
None
```