Function Definitions

Function Definition

Name of function in `pothole_case`

Zero or more parameters, comma-separated

```
def «function_name»(«parameters»):
    """docstring"""
    «body»
```

def keyword

One or more Python statements, indented 4 spaces

Computer Science
UNIVERSITY OF TORONTO

Function Definition

`def`: a keyword indicating a function definition

«function_name»

the function name, always in `pothole_case`

«parameters»

the parameter(s) of the function, 0 or more

a comma-separated list of variables whose values will be supplied (as arguments) when the function is called

`"""docstring"""`

an English explanation of what the function does, and some examples of it in use

«body»

one or more statements, often ending with a return statement

all body statements must be indented the same amount, usually 4 spaces

Function definition example

```
def add_three(n: int) -> int:
    """Return three more than n.

    >>> add_three(2)
    5
    >>> add_three(0)
    3
    """

    return n + 3
```

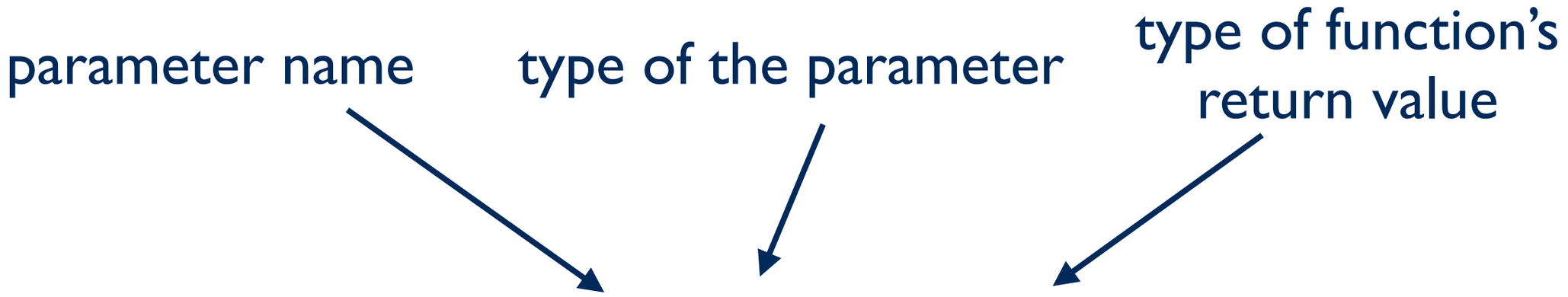
Function "header"

First line contains the **def** keyword, the name of the function, the arguments, and a "type contract"

```
def add_three(n: int) -> int:
    """Return three more than n.
    >>> add_three(2)
    5
    >>> add_three(0)
    3
    """

    return n + 3
```

Type Contract




```
def add_three(n: int) -> int:
    """Return three more than n.

    >>> add_three(2)
    5
    >>> add_three(0)
    3
    """

    return n + 3
```

Docstring

An English description of what the function **does**, but not how it does it.

```
def add_three(n: int) -> int:
    """Return three more than n.

    >>> add_three(2)
    5
    >>> add_three(0)
    3
    """
    return n + 3
```

Should start with 'Return' if function is returning something

Show two or three example calls to the function

Function Body

```
def add_three(n: int) -> int:
    """Return three more than n.

    >>> add_three(2)
    5
    >>> add_three(0)
    3
    """
    return n + 3
```

The python statements required to make the function do what the docstring says it does.

In this case, it's just one line, but can have more

Return statement

```
def add_three(n: int) -> int:
    """Return three more than n.

    >>> add_three(2)
    5
    >>> add_three(0)
    3
    """

    return n + 3
```

Indicates that this is the value the function evaluates to

Return Statement

Form:

```
return «expression»
```

How it's executed:

Evaluate the expression. This produces a value (which has a memory address).

Exit the function and produce that value to the caller.

More than one line in function body

```
def add_three(n: int) -> int:
    """Return three more than n.

    >>> add_three(2)
    5
    >>> add_three(0)
    3
    """

    result = n + 3
    return result
```

Can separate expression and return statement by using variables

Function Call

Name of function being called

Zero or more expressions, comma-separated

«*function_name*»(«*arguments*»)

Function Call

«*function_name*»(«*arguments*»)

The rules for executing a function call:

Evaluate the arguments. These produce values. Each value has a memory address.

Assign those values to the parameters. This stores the memory addresses of the values in the parameters.

Pause the current statement and execute the body of the function.