Function Development in Python

Lesson Objectives

At the end of this lesson, you should be able to:

- Describe the concept of functions
- Explain the importance of functions
- Define functions in the Python programming language

Topic Outline

What is a Function?

Why do we Need Functions?

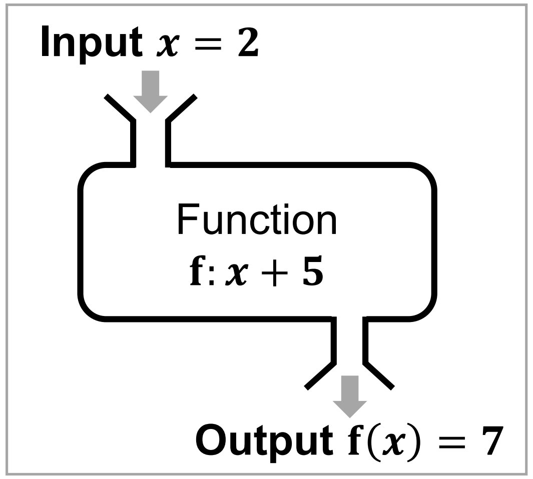
How to Define a Function in Python?

What is a Function?

FUNCTION

In Mathematics

performs some operation and returns **one** value/ thing



In Python

- represents a single operation to be performed
- takes zero or more arguments as input
- returns one value/ object as output

Python functions "encapsulate" the performance of its particular operation, so they can be used by others.

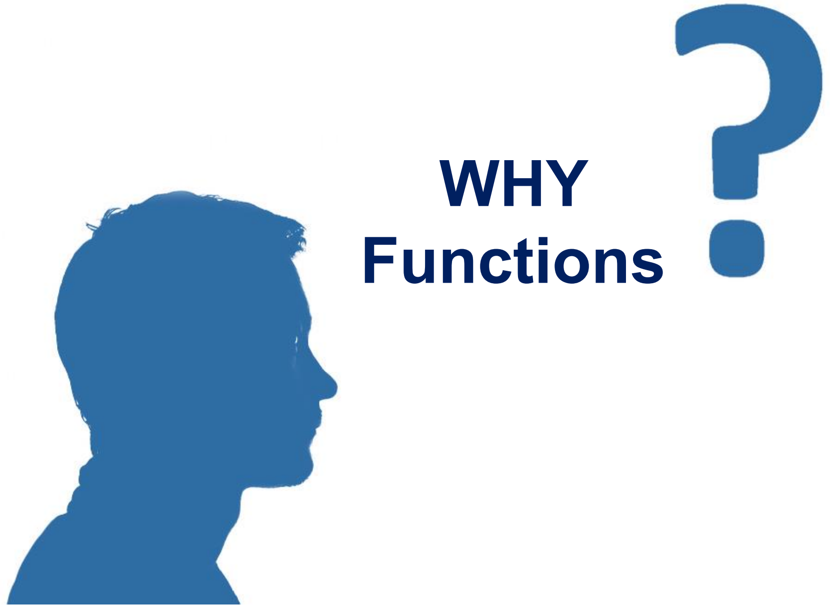
Functions in Python: Calculating sqrt

```
x = 10
precision = 0.001
low = 0
high = max(x, 1)
counter = 0
guess = (low + high) / 2.0
while abs (guess ** 2 - x) >
precision and counter <= 100:
    if(guess ** 2 < x):
        low = guess
    else:
        high = guess
    guess = (low + high) / 2.0
    counter += 1
assert counter <= 100, '100
iterations done and no good answer'
print('Num of iterations:',
counter, 'Estimate:', guess)
```

vs.

`sqrt(10)`

Importance of Functions



- Abstraction
- Divide-and-conquer problem solving
- Reuse
- Sharing
- Security
- Simplification and Readability

Mathematical Notation of Functions

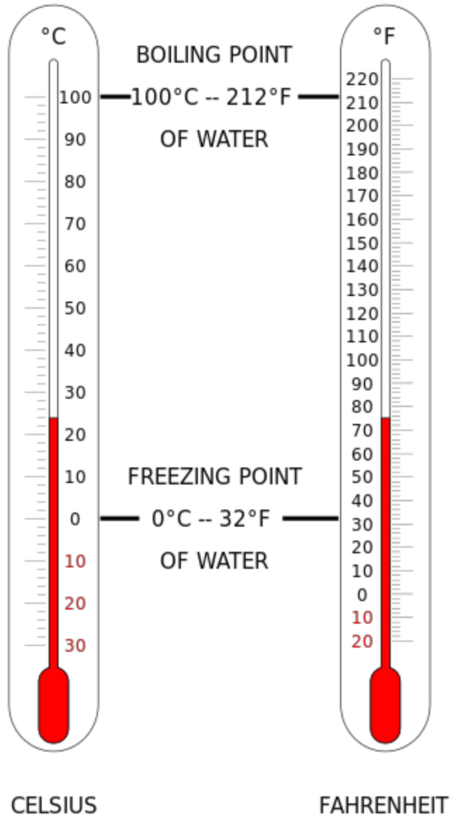
Consider a **function that converts temperatures in Celsius to Fahrenheit**:

Formula

```
F = C * 1.8 + 32.0
```

Functional Notation

`F = celsius2Fahrenheit(C)` where
`celsius2Fahrenheit(C) = C * 1.8 + 32.0`



Function Invocation in Python

Math

F = celsius2Fahrenheit(C)

Python

Terminology:
argument "C"

Invocation is the same:

```
F = celsius2Fahrenheit(C)
```

Function Definition in Python

Math

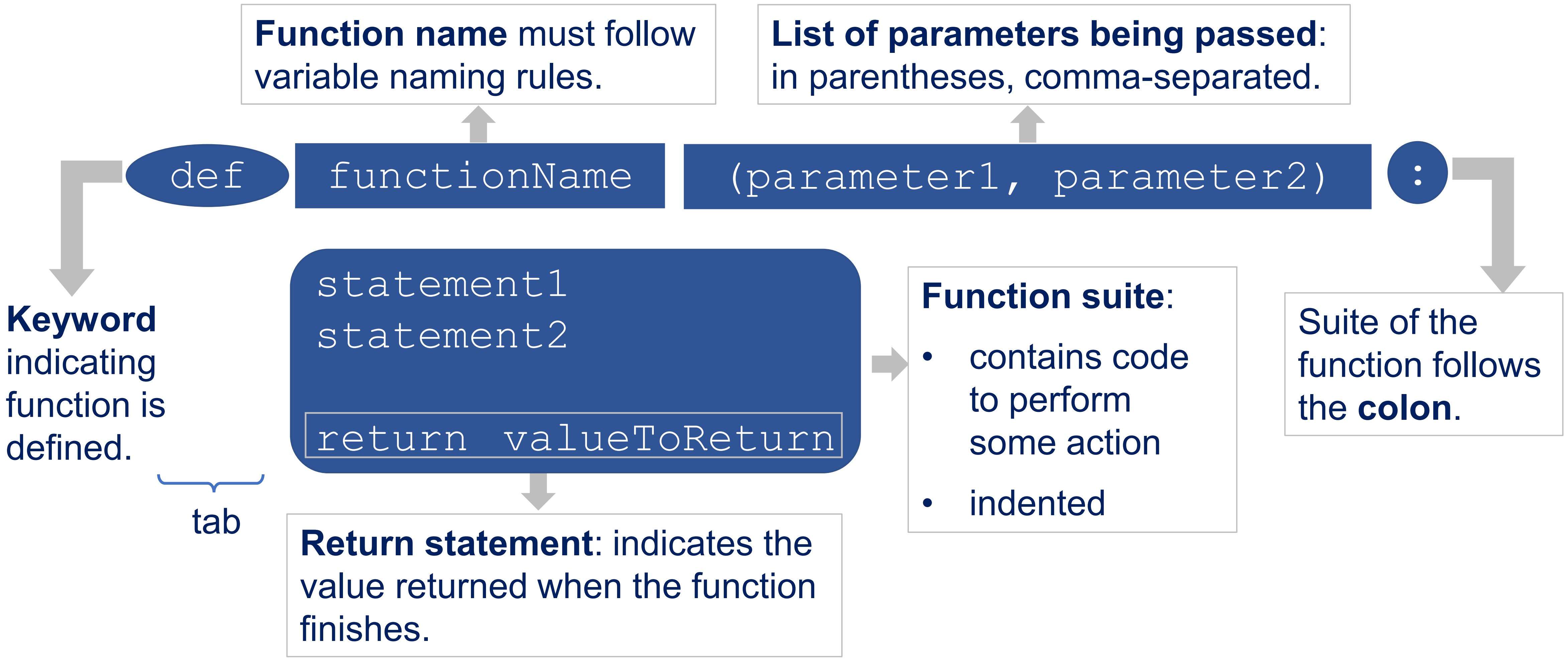
celsius2Fahrenheit(C) = C * 1.8 + 32.0

Python

Terminology:
parameter "C"

```
def celsius2Fahrenheit(C):
   return C * 1.8 + 32.0
```

Function Definition in Python



```
def functionName (parameter1, parameter2):
    statement1
    statement2

    return valueToReturn
```

Function name must follow variable naming rules.

List of parameters being passed: in parentheses, comma-separated.

Keyword indicating function is defined.

Suite of the function follows the **colon**.

Function suite:

- contains code to perform some action
- indented

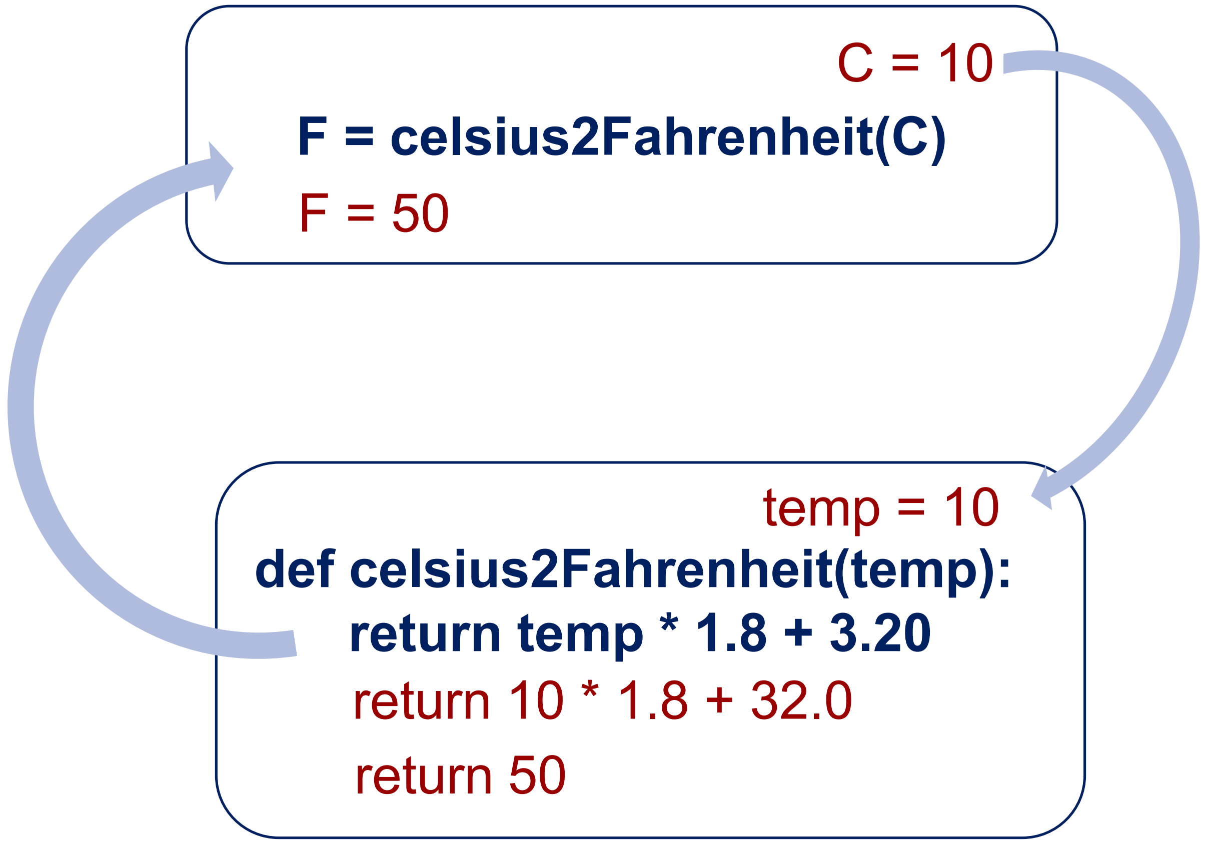
Return statement: indicates the value returned when the function finishes.

`return` Statement

- The `return` statement indicates the value that is returned by the function.
- The statement is optional (the function can return nothing).
- If there is no return, the function is often called a procedure.

Dynamics of Function Calls

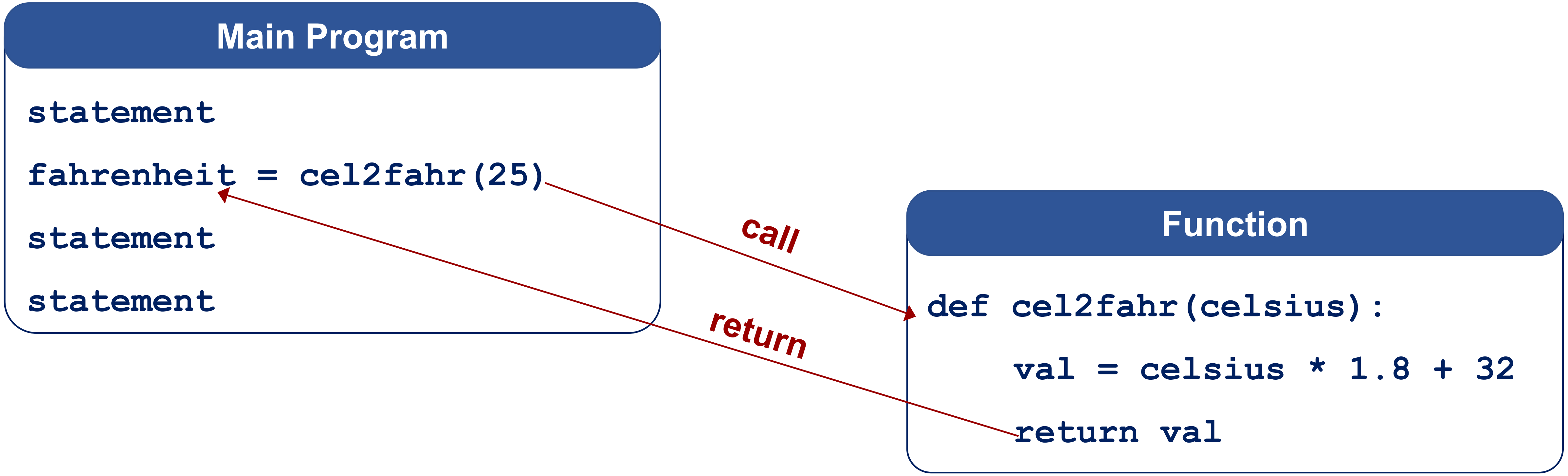
1. Function call copies argument C to parameter temp.
2. Control transfers to function “celsius2Fahrenheight”.
3. Expression in celsius2Fahrenheight is evaluated .
4. Value of expression is returned to invoker.



```
                              C = 10
F = celsius2Fahrenheit(C)
F = 50
```

```
                            temp = 10
def celsius2Fahrenheit(temp):
    return temp * 1.8 + 3.20
    return 10 * 1.8 + 32.0
    return 50
```

Dynamics of Function Calls



Principles of Writing a Function

Does one thing	Readable	Reusable	Complete	Not too Long
If it does too many things, it should be broken down into multiple functions (refactored).	If you write it, it should be readable. Give comments.	If it does one thing well, then when a similar situation (in another program) occurs, use it there as well.	A function should check for all the cases where it might be invoked. Check for potential errors.	Kind of synonymous with "**does one thing**". Use it as a measurement of doing too much.

A Function Example

A Function that Calculates the Length of an Input String

```
def str_length(a_str):
      count = 0

      for ch in a_str:
            count = count + 1;

      return count
```

```
str_length('abc')
```

↓

3

Procedures

- Functions **without return** statements are often called **procedures**.
- **Procedures** are used to perform some duty (print output, store a file, etc.).
- A **return** statement is not always required.

Multiple `return` Statements

- A function could have multiple `return` statements.

- The first executed return statement ends the function.

Caution

Multiple return statements might be confusing to the reader.

USE CAREFULLY!

Multiple `return` Statements

```python
def funcA (number):
        if number > 0:
                return "positive!"

        elif number < 0:
                return "negative!"

        else:
                return "zero!"
```

```
print(funcA(5))
   positive

print(funcA(-2))
   negative

print(funcA(0))
     zero
```

Functions Calling Functions

- Functions are made to solve a problem and can be called from other functions.
- Functions calling functions is the same as users calling functions.
  - There is no limit to the "depth" of multiple function calls.
  - Deep function calls could make following the flow of a program difficult.

Functions Calling Functions: Example

```
funcA('abc')
   positive
```

```
def str_length(a_str):
    count = 0

    for ch in a_str:
        count = count + 1;

    return count
```

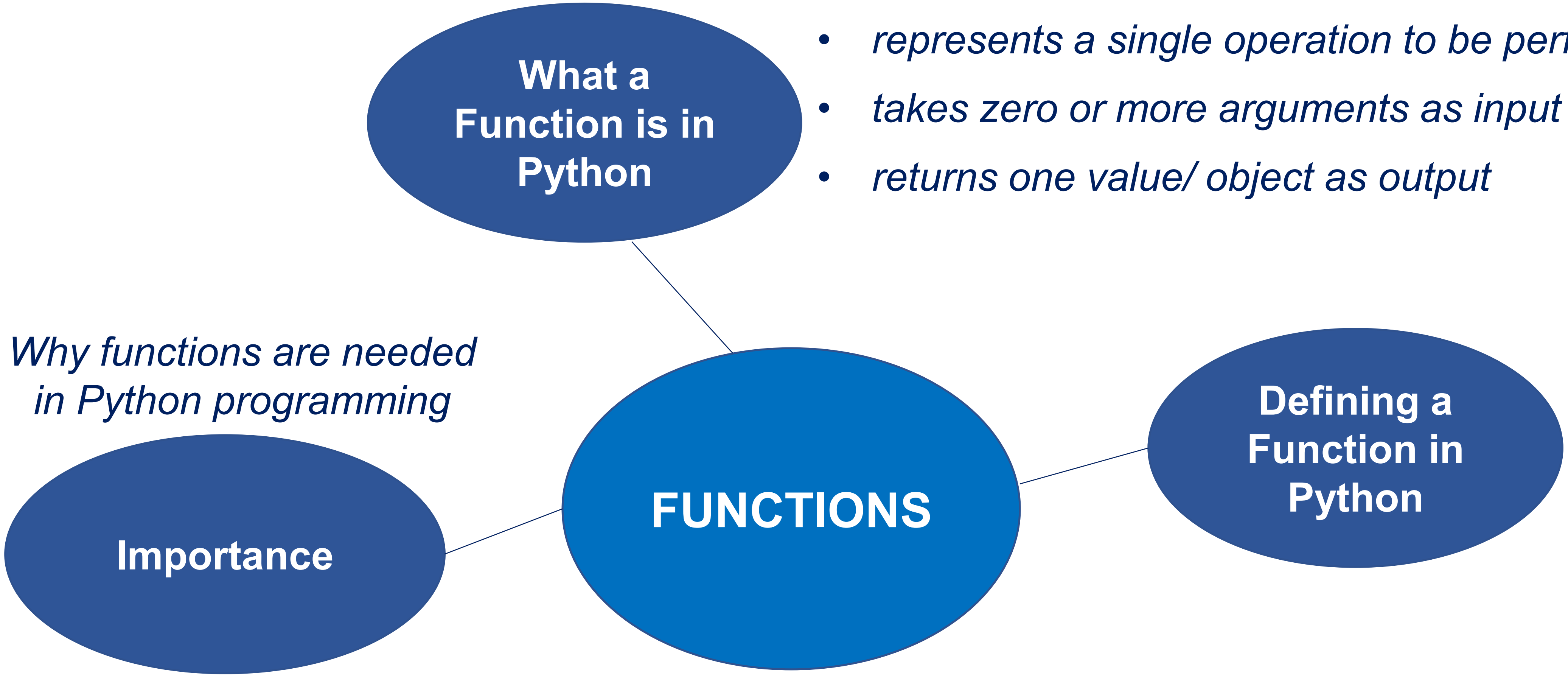
```
def funcA (text):
    length = str_length(text)

    if length > 0:
        return "positive!"

    elif length < 0:
        return "negative!"

    else:
        return "zero!"
```

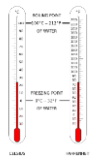
Summary



- *represents a single operation to be performed*
- *takes zero or more arguments as input*
- *returns one value/ object as output*

Why functions are needed in Python programming

References for Images

No.	Slide No.	Image	Reference
1	7		Tumisu (n.d.). Ask [Online Image]. Retrieved May 15, 2018 from https://pixabay.com/en/question-why-question-mark-ask-1038491/.
2	8		By User:Gringer - n /a, Public Domain, retrieved May 15, 2018 from https://commons.wikimedia.org/w/index.php?curid=10842578.
3	9, 10		By User:Bobarino - Made by following Information.png, CC BY-SA 3.0, retrieved May 15, 2018 from https://en.wikipedia.org/w/index.php?curid=9180601.
4	16, 19, 21		Python Logo [Online Image]. Retrieved April 24, 2018 from https://pixabay.com/en/language-logo-python-2024210/.
5	17		Prosmile (n.d.). Gear [Online Image]. Retrieved May 15, 2018 from https://pixabay.com/en/gear-icon-service-configuration-1674891/.

References for Images

No.	Slide No.	Image	Reference
6	18		Caution [Online Image]. Retrieved May 15, 2018 from https://pixabay.com/en/caution-hazard-warning-alert-152926/.