



Prifysgol Cymru
Y Drindod Dewi Sant
University of Wales
Trinity Saint David

ACCA5036

Scripting and Programming Concepts

Lecture 7: Functions in Python

Trawsnewid Addysg; Trawsnewid Bywydau
Transforming Education; Transforming Lives



www.uwtsd.ac.uk

Recap

- <https://app.sli.do/event/98tAqLh5Zdhtz6SWz6fvJC>





1. What will be the output of the following code?

```
my_list = [1, 2, 3, 4]  
my_list.append(5)  
print(my_list)
```



What data type does the following code represent?

```
my_data = (10, 20, 30)  
print(type(my_data))
```



What will be the output of the following dictionary operation?

```
my_dict = {"name": "Alice", "age": 25}  
print(my_dict["age"])
```



What will happen if we try to modify a tuple?

```
my_tuple = (1, 2, 3)  
my_tuple[0] = 10
```



Which of the following is the correct way to add an item to a set?

`my_set = {1, 2, 3}`



What will be the output of the following list slicing operation?

```
my_list = [10, 20, 30, 40, 50]  
print(my_list[1:4])
```



What will be the output of the following dictionary method?

```
my_dict = {"a": 1, "b": 2}  
print(my_dict.get("c", "Not Found"))
```



What will be the output of the following dictionary method?

```
my_dict = {"a": 1, "b": 2}  
print(my_dict.get("a", "Not Found"))
```



When is the submission due for your assessment?

- ① The Slido app must be installed on every computer you're presenting from

In-class Activity: Student Course Management System

This system allows users to manage **students and courses** while using different data structures:

- ✓ **List** → Stores student details.
- ✓ **Tuple** → Each student is stored as a tuple (ID, Name, Age).
- ✓ **Set** → Stores unique course names.
- ✓ **Dictionary** → Maps student IDs to enrolled courses.

<https://forms.office.com/e/xXjC3Hn91q>



In-class Activity: Student Course Management System

This system allows users to manage **students and courses** while using different data structures:

- ✓ **List** → Stores student details.
- ✓ **Tuple** → Each student is stored as a tuple (ID, Name, Age).
- ✓ **Set** → Stores unique course names.
- ✓ **Dictionary** → Maps student IDs to enrolled courses.

<https://forms.office.com/e/xXjC3Hn91q>



This week

Functions in Python

Implementing function examples

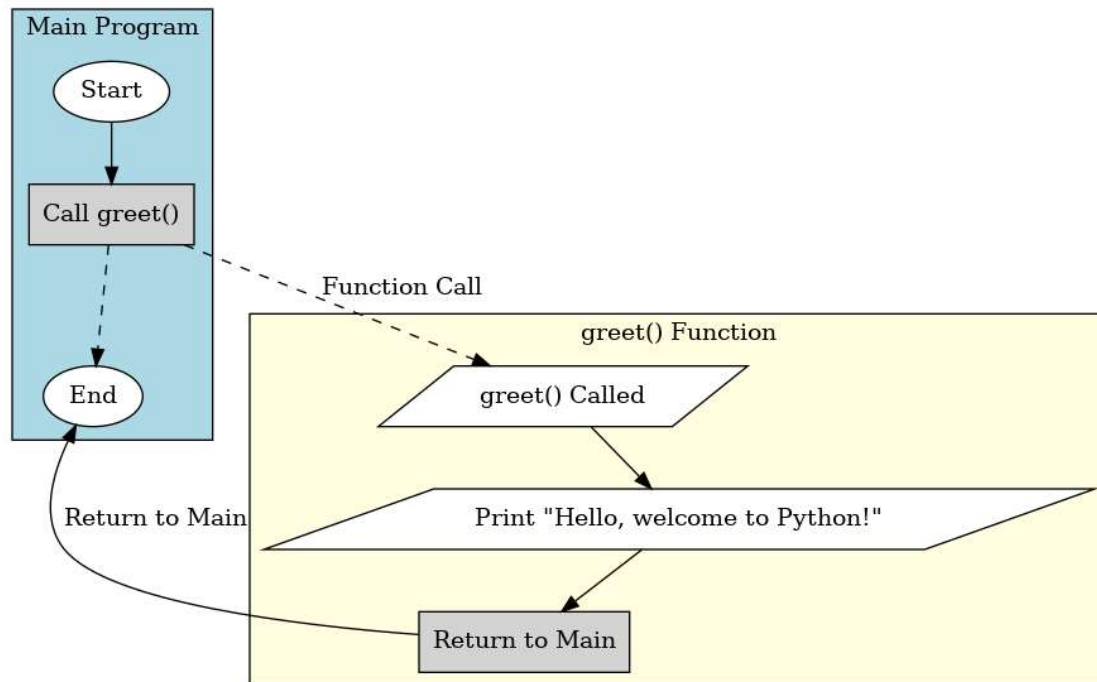
A Simple Function

```
def greet():  
    print("Hello, welcome to Python programming!")
```

Call the function

```
greet()
```

A Simple Function



```
def greet():  
    print("Hello, welcome to Python  
programming!")
```

```
# Call the function  
greet()
```

Reuse the block of code

```
def greet():  
    print("Hello, welcome to Python programming!")  
  
# Call the function  
greet()  
print("Python is not a snake")  
greet()
```

What is a Function?

- A function is a **block of reusable code** that performs a specific task.
- It allows you to avoid repeating the same code multiple times.
- Functions can take **inputs** (called **parameters** or **arguments**) and return an **output**

Why Use Functions?

- **Code Reusability:** Instead of writing the same code over and over, you can call a function whenever needed.
- **Organization:** Functions help break down a program into smaller, manageable pieces.
- **Debugging:** It's easier to debug when the code is divided into smaller parts (functions).

Basic Syntax of a Function

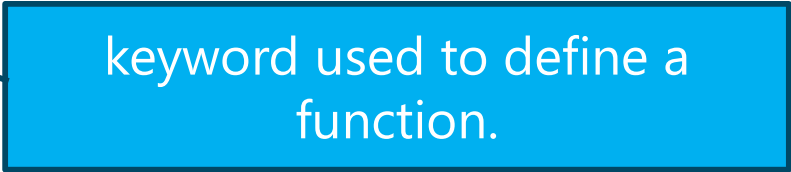
```
def function_name(parameters):  
    # Code block  
    return output
```

Basic Syntax of a Function

```
def function_name(parameters):
```

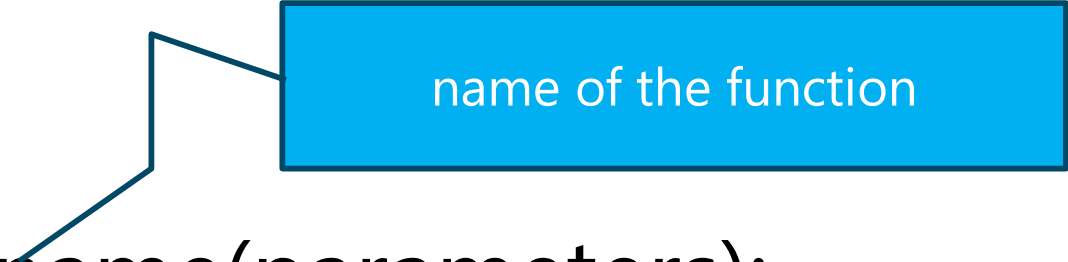
```
    # Code block
```

```
    return output
```



keyword used to define a
function.

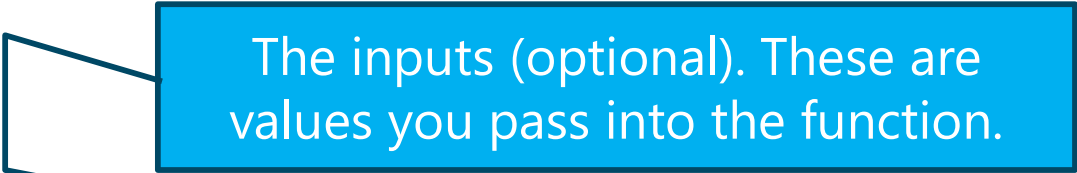
Basic Syntax of a Function



name of the function

```
def function_name(parameters):  
    # Code block  
    return output
```

Basic Syntax of a Function



The inputs (optional). These are values you pass into the function.

```
def function_name(parameters):  
    # Code block  
    return output
```

Basic Syntax of a Function

```
def function_name(parameters):  
    # Code block  
    return output
```



The output of the function (optional).
This is what the function sends back
after processing.

Example: Calculate the area of a rectangle

Without using functions:

```
length = float(input("Enter the length: "))
```

```
width = float(input("Enter the width: "))
```

```
area = length * width
```

```
print("Area of the rectangle:", area)
```

Example: Calculate the area of a rectangle

With function:

```
def calculate_area(length, width):  
    return length * width
```

```
length = float(input("Enter the length: "))  
width = float(input("Enter the width: "))
```

```
area = calculate_area(length, width)  
print("Area of the rectangle:", area)
```

Time taken with & without functions

```
import time
```

```
start_time = time.time()
```

```
length = 100.5
```

```
width = 200.5
```

```
area = length * width
```

```
print("Area of the rectangle:", area)
```

```
end_time = time.time()
```

```
time_taken = end_time - start_time
```

```
print(f"Time taken: {time_taken} seconds")
```

Time taken with & without functions

```
import time
```

```
def calculate_area(length, width):  
    return length * width
```

```
start_time = time.time()
```

```
length = 100.5  
width = 200.5
```

```
area = calculate_area(length, width)
```

```
print("Area of the rectangle:", area)
```

```
end_time = time.time()
```

```
time_taken = end_time - start_time  
print(f"Time taken: {time_taken} seconds")
```

Time taken with & without functions

```
import time
```

```
start_time = time.perf_counter()
```

```
length = 100.5
```

```
width = 200.5
```

```
area = length * width
```

```
print("Area of the rectangle:", area)
```

```
end_time = time.perf_counter()
```

```
time_taken = end_time - start_time
```

```
print(f"Time taken: {time_taken} seconds")
```

Time taken with & without functions

```
import time
```

```
def calculate_area(length, width):  
    return length * width
```

```
start_time = time.perf_counter()
```

```
length = 100.5  
width = 200.5
```

```
area = calculate_area(length, width)
```

```
print("Area of the rectangle:", area)
```

```
end_time = time.perf_counter()  
time_taken = end_time - start_time  
print(f"Time taken: {time_taken} seconds")
```

Functions with Parameters

```
def greet_user(name):  
    print(f"Hello, {name}!")
```

```
greet_user("Alice")
```

Functions with Parameters

```
def greet_user(name):  
    print(f"Hello, {name}!")
```

```
greet_user("Alice")
```

Call function with
integer, float and
Boolean true or false,
will that work?

Functions with Parameters

```
def greet_user(name):  
    print(f"Hello, {name}!")
```

```
greet_user("Alice")  
greet_user((1,2,3))
```

Change the
parameter to list,
tuple, set
will that work?

Functions with Parameters

```
def greet_user(name):  
    if not isinstance(name, str):  
        print("Error: The input must be a string.")  
    return  
    print(f"Hello, {name}!")  
greet_user("Alice")  
greet_user((1,2,3))
```

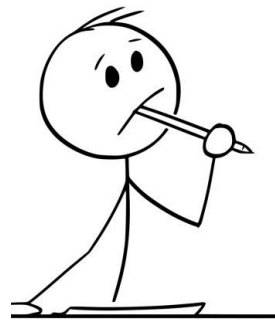
Functions with Parameters

```
def greet_user(name):  
    if not isinstance(name, int):  
        print("Error: The input must be an integer.")  
        return  
    print(f"Hello, {name}!")  
greet_user("Alice")  
greet_user((1,2,3))  
greet_user(123)
```

Using Arbitrary Arguments

```
def greet_user(name):  
    print(f"Hello, {name}!")
```

```
greet_user("Alice")
```



What should be
done to greet
Alice, Bob &
Charlie?

Using Arbitrary Arguments

```
def greet_users(*args):  
    for user in args:  
        print(f"Hello, {user}!")
```

```
greet_users("Alice", "Bob", "Charlie")
```



Using Lambda Function

```
add = lambda x, y: x + y
```

```
result = add(5, 3)
```

```
print(result)
```

Functions with Return Values

```
def greet_user(name):  
    print(f"Hello, {name}!")
```

```
def add(a, b):  
    return a + b
```

```
greet_user("Alice")  
result = add(3, 5)  
print(f"The sum is: {result}")
```

Functions with Return Values

```
def add(a, b):  
    return a+b
```

```
result = add(3, 5)  
print(f"The sum is: {result}")  
print("The sum is: ", add(4,5))
```

Using functions (including input function)

```
def get_input():  
    length = float(input("Enter the length: "))  
    width = float(input("Enter the width: "))  
    return length, width
```

```
def calculate_area(length, width):  
    return length * width
```

```
# Get user input  
length, width = get_input()
```

```
# Calculate and display the area  
area = calculate_area(length, width)  
print("Area of the rectangle:", area)
```

Using functions (including input function)

```
def get_input():  
    length = float(input("Enter the length: "))  
    width = float(input("Enter the width: "))  
    return length, width
```

```
def calculate_area(length, width):  
    return length * width
```

```
def display_area(area):  
    print("Area of the rectangle:", area)
```

```
# Main program  
length, width = get_input()  
area = calculate_area(length, width)  
display_area(area)
```

Using functions (allows users to input values n times and calculate the area for each rectangle)

Write a Python program that allows the user to calculate the area of multiple rectangles.

The program should:

1. Ask the user how many rectangles they want to calculate.
2. For each rectangle, take the length and width as input.
3. Use functions to:
 - Get user input (`get_input()`)
 - Calculate the area (`calculate_area()`)
 - Display the result (`display_area()`)
4. Repeat the process for the specified number of rectangles and display each area separately.

Using functions (allows users to input values **n** times and calculate the area for each rectangle)

```
def get_input():  
    """Function to get length and width from the user."""  
    length = float(input("Enter the length: "))  
    width = float(input("Enter the width: "))  
    return length, width  
  
def calculate_area(length, width):  
    """Function to calculate the area of a rectangle."""  
    return length * width  
  
def display_area(area, count):  
    """Function to display the area of a rectangle."""  
    print(f"Rectangle {count}: Area = {area}")  
  
# Main program  
n = int(input("How many rectangles do you want to calculate? "))  
  
for i in range(1, n + 1):  
    print(f"\nRectangle {i}:")  
    length, width = get_input()  
    area = calculate_area(length, width)  
    display_area(area, i)  
  
print("\nAll calculations are done!")
```

Knowledge Check - Functions

```
def my_function(x, y):  
    return x + y
```

```
result = my_function(5, 7)  
print(result)
```

What will be the output of the above code?

- a) 12
- b) 57
- c) 5 + 7
- d) Error

Knowledge Check - Functions

```
def greet(name):  
    return "Hello, " + name + "!"
```

```
message = greet("Alice")  
print(message)
```

What will be the output?

- a) Hello, Alice!
- b) Hello, name!
- c) Alice
- d) Error

Knowledge Check - Functions

```
def modify_list(numbers):  
    numbers[0] = 100  
    return numbers
```

```
my_list = [1, 2, 3]  
print(modify_list(my_list))
```

What will be the output of the code?

- a) [100, 2, 3]
- b) [1, 2, 3]
- c) [1, 2, 3, 100]
- d) Error

Knowledge Check - Functions

```
def update_dict(my_dict):  
    my_dict["age"] = 30  
    return my_dict
```

```
person = {"name": "John", "age": 25}  
print(update_dict(person))
```

What will be the output?

- a) {'name': 'John', 'age': 25}
- b) {'name': 'John', 'age': 30}
- c) {'name': 'John', 'age': '30'}
- d) Error

Knowledge Check - Functions

```
def power(base, exponent=2):  
    return base ** exponent
```

```
print(power(3))
```

What will be the output of the code?

- a) 9
- b) 6
- c) 3
- d) Error

Knowledge Check - Functions

```
def add(a, b):  
    return a + b
```

```
def subtract(a, b):  
    return a - b
```

```
x = 5  
y = 3  
print(add(x, subtract(x, y)))  
What will be the output?
```

- a) 3
- b) 5
- c) 8
- d) 7

Knowledge Check - Functions

```
def append_element(my_list):  
    my_list.append(4)  
    return my_list
```

```
my_list = [1, 2, 3]  
print(append_element(my_list))
```

What will be the output?

- a) [1, 2, 3]
- b) [1, 2, 3, 4]
- c) [4, 1, 2, 3]
- d) Error

Knowledge Check - Functions

```
def test_scope():  
    a = 5  
    return a
```

```
print(a)
```

What will be the output of the code?

- a) 5
- b) Error: 'a' is not defined
- c) None
- d) Undefined

Knowledge Check - Functions

```
def test_scope():  
    a = 5  
    return a
```

```
print(a)
```

```
def test_scope(a):  
    print(a)  
    return a  
  
test_scope(2)
```

What will be the output of the code?

- a) 5
- b) Error: 'a' is not defined
- c) None
- d) Undefined

Knowledge Check - Functions

```
def multiply(a, b=2):  
    return a * b
```

```
print(multiply(5))  
print(multiply(5, 4))
```

What will be the output?

- a) 10, 20
- b) 10, 8
- c) 5, 8
- d) Error

Knowledge Check - Functions

```
def swap(a, b):  
    return b, a
```

```
x, y = swap(3, 5)  
print(x, y)
```

What will be the output?

- a) 5, 3
- b) 3, 5
- c) Error: too many values to unpack
- d) None

Knowledge Check - Functions

```
def update_dict(d):  
    d['age'] = 40  
    return d
```

```
my_dict = {'name': 'Alice', 'age': 30}  
update_dict(my_dict)  
print(my_dict)
```

What will be the output?

- a) {'name': 'Alice', 'age': 30}
- b) {'name': 'Alice', 'age': 40}
- c) Error: cannot modify dictionary inside function
- d) None

Knowledge Check - Functions

```
def update_dict(d):  
    d['age'] = 40  
    return d
```

```
my_dict = {'name': 'Alice', 'age': 30}  
update_dict(my_dict)  
print(my_dict)
```

What will be the output?

- a) {'name': 'Alice', 'age': 30}
- b) {'name': 'Alice', 'age': 40}
- c) Error: cannot modify dictionary inside function
- d) None

Knowledge Check - Functions

```
def factorial(n):  
    if n == 0:  
        return 1  
    else:  
        return n * factorial(n-1)  
  
print(factorial(5))
```

What will be the output of the above code?

- a) 120
- b) 24
- c) 5
- d) 1

Knowledge Check - Functions

```
def factorial(n):
```

```
    if n == 0:
```

```
        return 1
```

```
    else:
```

```
        return n * factorial(n-1)
```

A **recursive function** is a function that **calls itself**

```
print(factorial(5))
```

What will be the output of the above code?

- a) 120
- b) 24
- c) 5
- d) 1

Knowledge Check - Functions

```
def sum_numbers(n):  
    if n == 0:  
        return 0  
    else:  
        return n + sum_numbers(n - 1)  
  
print(sum_numbers(4))
```

What will be the output of the code?

- a) 10
- b) 4
- c) 12
- d) 24

Knowledge Check - Functions

```
def count_down(n):  
    if n <= 0:  
        print("Done!")  
    else:  
        print(n)  
        count_down(n-1)
```

count_down(3)

What will be the output?

- a) 3 2 1 Done!
- b) Done! 3 2 1
- c) 3 Done!
- d) Error

Knowledge Check - Functions

```
def count_down(n):  
    if n <= 0:  
        print("Done!")  
    else:  
        print(n)  
        count_down(n-1)
```

count_down(True)

What will be the output?

- a) 3 2 1 Done!
- b) 1 Done!
- c) True Done!
- d) Error

Knowledge Check - Functions

```
def fibonacci(n):  
    if n <= 1:  
        return n  
    else:  
        return fibonacci(n-1) + fibonacci(n-2)  
  
print(fibonacci(5))
```

What will be the output?

- a) 5
- b) 8
- c) 3
- d) 10

Knowledge Check - Functions

```
def infi_recursion():  
    return infinite_recursion()
```

```
print(infi_recursion())
```

What will be the output of the above code?

- a) Infinite recursion and eventually a RecursionError
- b) 1
- c) None
- d) Error due to missing base case

Knowledge Check - Functions

```
def infi_recursion():  
    return infinite_recursion()
```

```
print(infi_recursion())
```

What will be the output of the above code?

- a) Infinite recursion and eventually a RecursionError
- b) 1
- c) None
- d) Error due to missing base case

Knowledge Check - Functions

```
def reverse_string(s):  
    if len(s) == 0:  
        return s  
    else:  
        return reverse_string(s[1:]) + s[0]  
  
print(reverse_string("hello"))
```

What will be the output of the above code?

- a) "olleh"
- b) "hello"
- c) "h"
- d) Error

Knowledge Check - Functions

```
def countdown(n):  
    if n == 0:  
        return "Start!"  
    else:  
        return countdown(n-1)  
  
print(countdown(3))
```

What will be the output of the code?

- a) "Start!"
- b) 3 2 1 Start!
- c) Error
- d) None

Knowledge Check

What happens if you forget the deadline for the assessment, which is **25th April 2025**?

a) The universe will collapse, and your assignment will be sent to the moon. 

b) You will be greeted by a team of robotic professors who give out endless homework. 

c) You'll be required to write an essay titled "Why I Forgot the Deadline and How I Will Fix It." 

d) You'll lose your grade and will forever be haunted by the ghost of missed deadlines. 

Functions : In-class activity

Student Grade Calculator

<https://forms.office.com/r/uD9zgurru1>



Thank you

Gracias

Merci

Danke

Grazie

Obrigado / Obrigada

Dank je / Dank u

Спасибо (Spasibo)

谢谢 (Xièxiè)

ありがとう (Arigatou)

감사합니다 (Gamsahamnida)

धन्यवाद (Dhanyavaad)

شكرا (Shukran)

Teşekkürler

Ευχαριστώ (Efharisto)

תודה (Toda)

ขอบคุณ (Khxbkhun)

Asante

Salamat