Python for Beginners: Functions

Connect with me: Youtube | LinkedIn | WhatsApp Channel | Web | Facebook | Twitter

- Download PDF
- To access the updated lecture notes, please click on the following link: https://yasirbhutta.github.io/python/docs/functions.html

Functions

Syntax of function:

- Video: How to Add Two Numbers and Print the Result
- Video: How to Find the Maximum Value in a List of Numbers
- Video: How to Write a Python Function to Find the Length of a Word
- Video: Learn How to Use Default Parameters in Function Definition
- Video: Understanding *args in Functions - How to Add Any Number of Arguments with *args
- Video: How to Use *args in Python Functions
- Video: How to use **kwargs in Python

See also:

- Python Quiz - Functions

See also:

- Built-in Functions - Python 3.12.1 documentation

Anonymous (Lambda) Functions

- A lambda function in Python is a small, anonymous function that can be defined without name.
- Lamdba functions are used to write functions consisting of a single statement.
- A lambda function can take any number of arguments, but can only have one expression.
- It is created using the 'lambda' keyword, and it is often used as an argument to a higher-order function (a function that takes another function as an argument).

Syntax:

The syntax of a lambda is

```
lambda arguments:express
```

Example #1:

Following code is used to write the function to add 10 in given number.

```
def add_ten(x)
    return x + 10
```

above function can be written by the lamdba function in python.

```
add_ten = lamdba x: x + 10
print(add_ten(5) # 15
```

Example #2: multiple two numbers

use of lambda function to multiple two numbers

```
mul = lambda a, b : a * b
print(mul(2,4)) # 8
```

Example #3:

```
lambda x, y : x + y

_(6,8) # 14
```

Note: In the interactive interpreter, the single underscore**(_)** is bound to the last expression evalued.

Example #4: Immediately invoked function expression

```
(lambda x, y : x + y)(6,8) # 14
```

The lambda function above is defined and then immediately called with two arguments (6,8). it retuns the value **14**, which is the sum of the arguments.

Example #5:

```
def multiply(lambda(x,y):
    retun x*y

result = (lambda x,y : multiply(x,y))(5,3)
print(result) # Output: 15

mult = lambda x,y : multiply(x,y)
```

```
result = mult(6,2)
print(result) # Output: 12
```

- Video: How to: Use of Lambda function
- Video: How to: Use of Lambda function

Built-in Functions

Filter()

- In Python, the 'filter()' function is used to filter a sequence (e.g. list, tuple, etc.) by applying a certain test to each eleent of the sequence and returning only the elements that pass the test.
- The 'filter()' function takes two arguments: function and an iterable. The function is applied to each element of the iterable, and if the function retuns 'True' for that element, the element is included in the resulting iterable.

Example #5 use of lambda in filter() function

```
numbers = [1,2,3,4,5,6,7,8,9,10] # list

even_numbers= list(filter(lambda x : x % 2 == 0,numbers))

print(even_numbers)
```

Map()

- In Python, the 'map()' function is used to apply a certain function to each element of an iterable (e.g. list, tuple, etc.) and return an iterable containing the results.
- The 'map()' function takes two arguments: a function and an iterable. The function is applied to each element of the iterable, and the result of the function is included in the resulting iterable.

Example #6: use of lambda in map() function

```
numbers = [1,2,3,4,5,6,7,8,9,10]

squared_numbers = map(lambda x : x **2 ,numbers)

print(list(squared_numbers))
```

[need to add details]

See also:

- Video: How to: use a LAMBDA function as an argument in filter() and map()

Generators

- Video: How to Use Yield to Generate Values
- Video: Learn to Generate the Fibonacci Sequence in Python using Generators

True/False (Mark T for True and F for False)

Multiple Choice (Select the best answer)

Exercises

- Write a function `sum3(num1,num3,num3)` that takes three numbers as input and returns the sum.

- Write a function `SumNum(num1)` that takes a number as input and returns the sum of numbers from 1 to that number (num1).

- Write a function `sumSquares(x)` that takes a vector of numbers as input and returns the sum of their squares.

- Write a function `isEven(x)` that takes a number as input and returns true if it is even, and false otherwise.

- Write a program with three functions:

    1. **`isEven(n)`:** This function takes an integer `n` as input and returns `True` if `n` is even and `False` otherwise. You can use the modulo operator (`%`) to check for evenness.
    2. **`printTable(n)`:** This function takes an integer `n` as input and prints its multiplication table. The table should show the product of `n` with each number from 1 to 10, formatted like `n * i = n * i`, where `i` is the current number in the loop.
    3. **`main`:** The main program should:
        - Prompt the user to enter an integer.
        - Use the `isEven(n)` function to check if the entered number is even.
        - If the number is even, call the `printTable(n)` function to print its multiplication table.
        - If the number is odd, print a message indicating the number is odd and not eligible for printing a table.

Example output:

```
Enter an integer: 4
4 is even! Here's its multiplication table:
4 * 1 = 4
4 * 2 = 8
4 * 3 = 12
...
4 * 10 = 40
```

- Write a function `avgPositive(data)` that takes a list of numbers as input and returns the average of all positive numbers in the list.

Review Questions

References and Bibliography