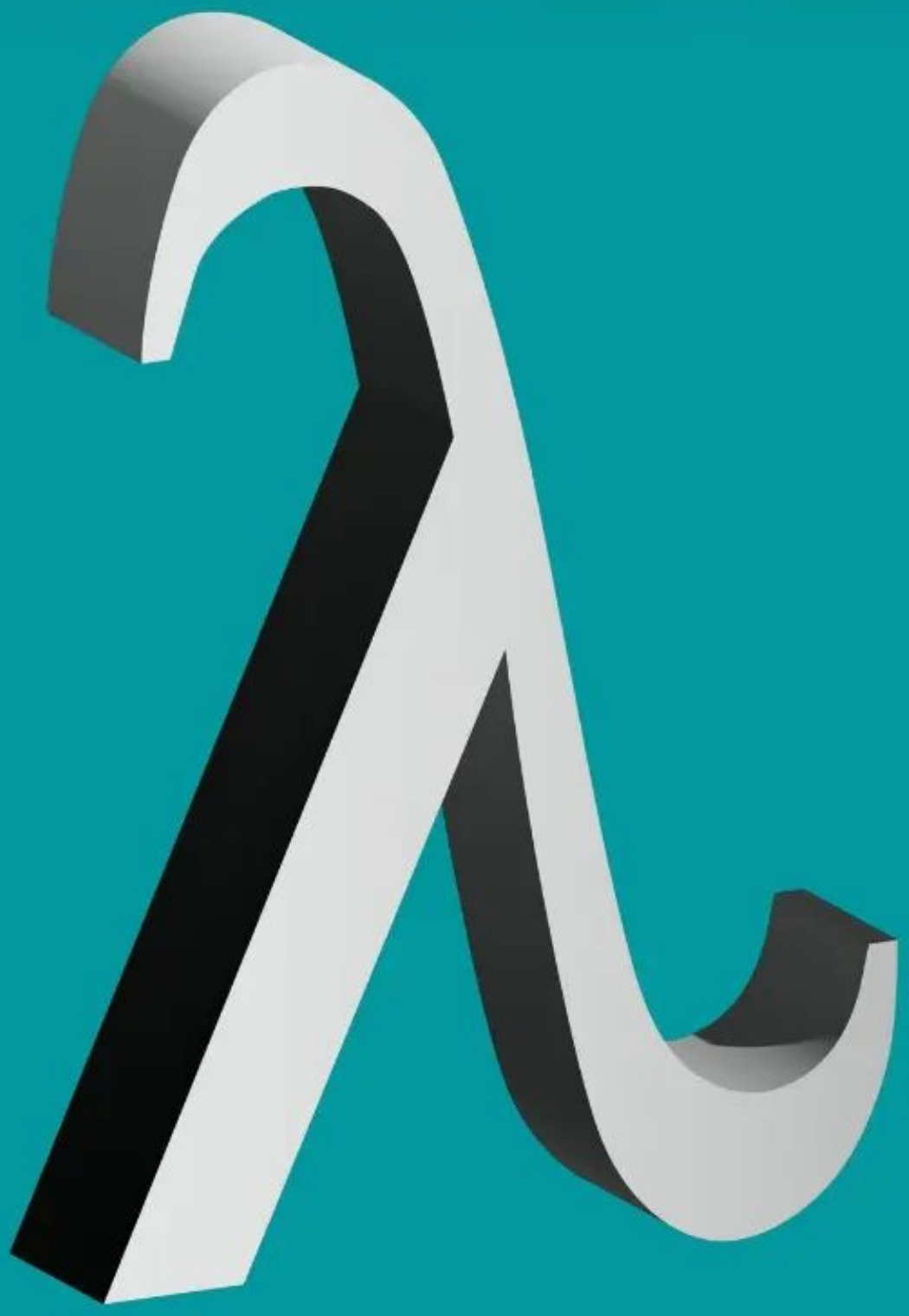
LAMBDA



```
lambda arguments : expression
```

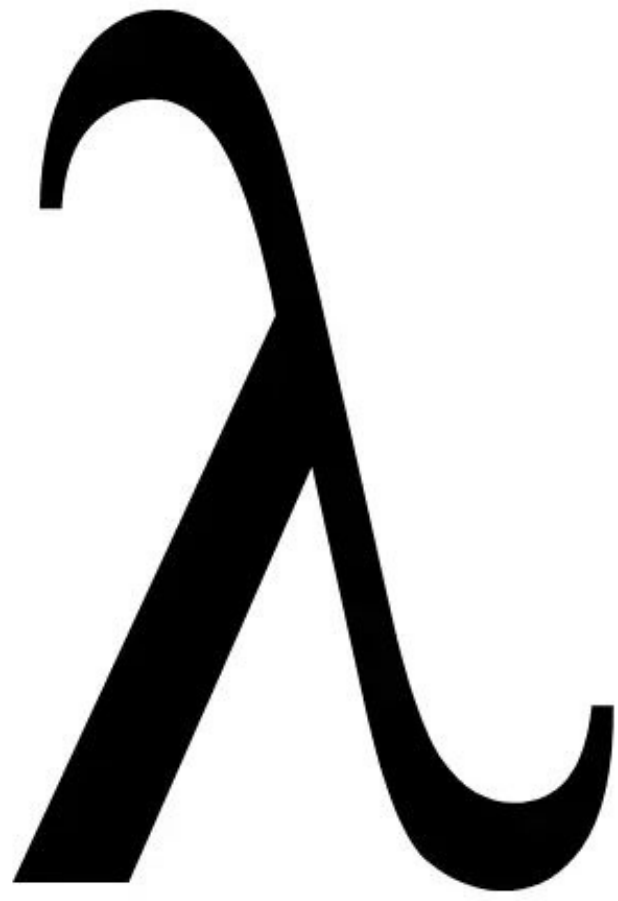
Lambda Function:

Lambda Functions are also called **Anonymous Functions**. An Anonymous Function is a function defined without a name.

As we know to define a normal function in python we need to use **def** keyword

But in this Case of anonymous functions,

We use the **lambda** keyword to define the functions.



Syntax

A lambda can have multiple arguments

The expression always returns an obiect

```
lambda arguments : expression
```

Every lambda begins with the "lambda" keyword

A colon precedes the expression

Keyword

argument

```
f= lambda a : a * a
```

function obiect that accepts and stores the result of the expression

one-line expression

Example:

```
sum = lambda x, y: x + y
print(sum(3, 4))
```

```
# Output: 7
```

Using **filter()** function to get all even numbers from a list:

```
numbers = [1, 2, 3, 4, 5,
6, 7, 8, 9, 10]
even_numbers = filter(lambda
x: x%2==0, numbers)
print(list(even_numbers))
```

```
Output: [2, 4, 6, 8, 10]
```

Using **map()** function to square all numbers in a list:

```
numbers = [1, 2, 3, 4, 5]

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

print(list(squared_numbers))
```

```
# Output: [1, 4, 9, 16, 25]
```

Using **reduce()** function to calculate the product of all elements in a list:

```
from functools import
reduce
numbers = [1, 2, 3, 4, 5]
product = reduce(lambda x,
y: x*y, numbers)
print(product)
```

```
Output: 120
```