

```

Mast-TJ00:
Tret Tourstatinnal(Precting));
rmed9sivestaliut:(roscfeccion):,
rUr.PSLA SidYut:(palotreccion):
mast #p0cs
Tree.fostle.rate:feelngiis)
ruckmsiy.bifint:(Peler_cartil));
presertef cppile:
(rosc faperyion.30MM,prolt)

with Beocce Cigesertice:
foce. Pastt matitul (procesible.y);

wint foe: Intmitage lyier Blnsle:
rosc cidk:);
Fast FoveCrontocation DinFkill;
foes:Tourting.Sp:Caplayints/festmlettire:

veportile ice.crinks;
Percrcityhe/chimpire:

focertis,listmic/Patinging:
foye foutitp1llg saple restthich).
(agaly.oplot:ts nopers:fooperate.autinoms.com: strtile.
reyels apost: listilp.

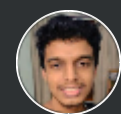
(ypryslet: Peest/Palfperslre:
inge. Fast.chat.loast,puchostiol.rrsi)sectioni

(Mescretp:Phalte.Spion:sftwion.restrats clobs.
(Anefement #:Eap/phook.cnest --1ml)
- Peplead
(AerckKoystinming/ectice);
(ua: foet ine.cplon:
(Mhestike:Ldetinimsaufectiblic.tyl);

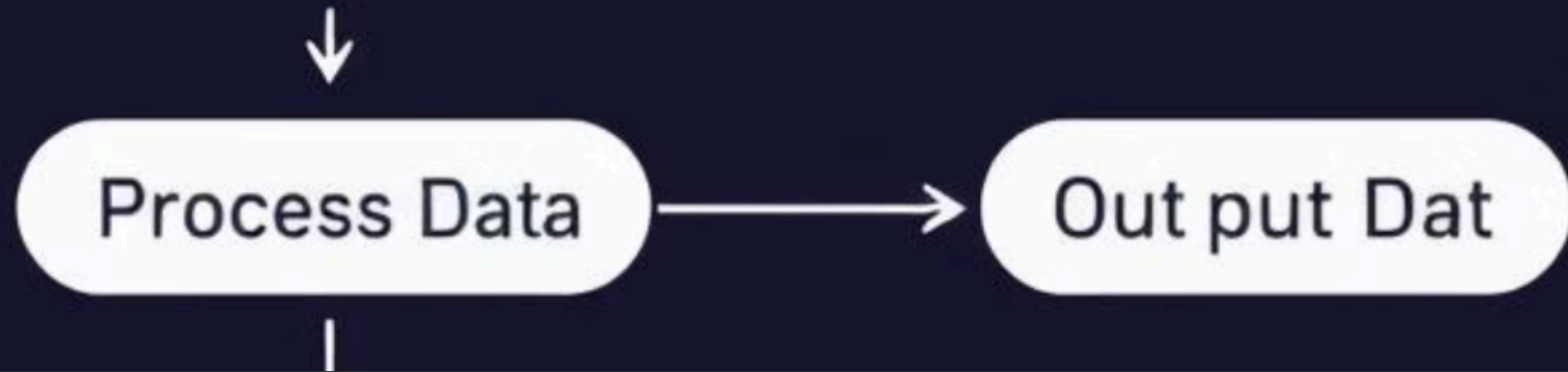
```

Python Functions and Lambda Expressions

Welcome to a deep dive into Python functions and lambda expressions! This presentation explores their functionalities, syntax, and real-world applications.



by Aravind K



Introduction to Functions

Functions are blocks of reusable code that perform specific tasks.

They promote code organization, avoid repetition, and improve readability.

1

Definition

Functions are defined using the 'def' keyword, followed by the function name and parameters.

2

Purpose

Functions break down complex tasks into smaller, manageable units.

3

Syntax

The basic syntax involves 'def', function name, parameters, and a code block.

Example of a Simple Function

Functions can perform operations like calculations, data processing, or any task you need to repeat.

They can return a result that can be used later in the program.

Function Definition

```
```python def greet(name):  
 return f"Hello, {name}!" ```
```

## Function Call

```
```python message =  
    greet("Aravind") print(message)  
```
```

## Output

```
``` Hello, Aravind! ```
```

function parameters

keyword

keyword

default

Function Parameters

Parameters allow functions to accept and process different inputs, making them more versatile.

| Parameter Type | Description | Example |
|-----------------------|---|---|
| Positional Parameters | Must be passed in the correct order. | <pre>python def
add(x, y): return x +
y</pre> |
| Keyword Parameters | Passed by explicitly stating the parameter name. | <pre>python def
add(x, y): return x +
y
add(y=5, x=3)</pre> |
| Default Parameters | Allow setting default values for optional parameters. | <pre>python def
greet(name="User")
: return f"Hello,
{name}!"</pre> |

Lambda Expressions

Lambda expressions are anonymous functions defined using the 'lambda' keyword.

They are used for short, throwaway functions without the need to formally define them.

1 Definition

A lambda function is defined in a single line using 'lambda', arguments, and an expression.

2 Purpose

Lambda functions provide a concise way to create simple functions without the need for a formal function definition.

3 Use Cases

They are commonly used with functions like 'map', 'filter', and 'reduce' to apply simple operations.

Example of a Lambda Function

Lambda functions are a convenient way to create concise functions.

They are useful for tasks that require simple operations within a larger program.

Lambda Function

```
```python square = lambda x: x *  
x ```
```

## Function Call

```
```python result = square(5)  
print(result) ```
```

Output

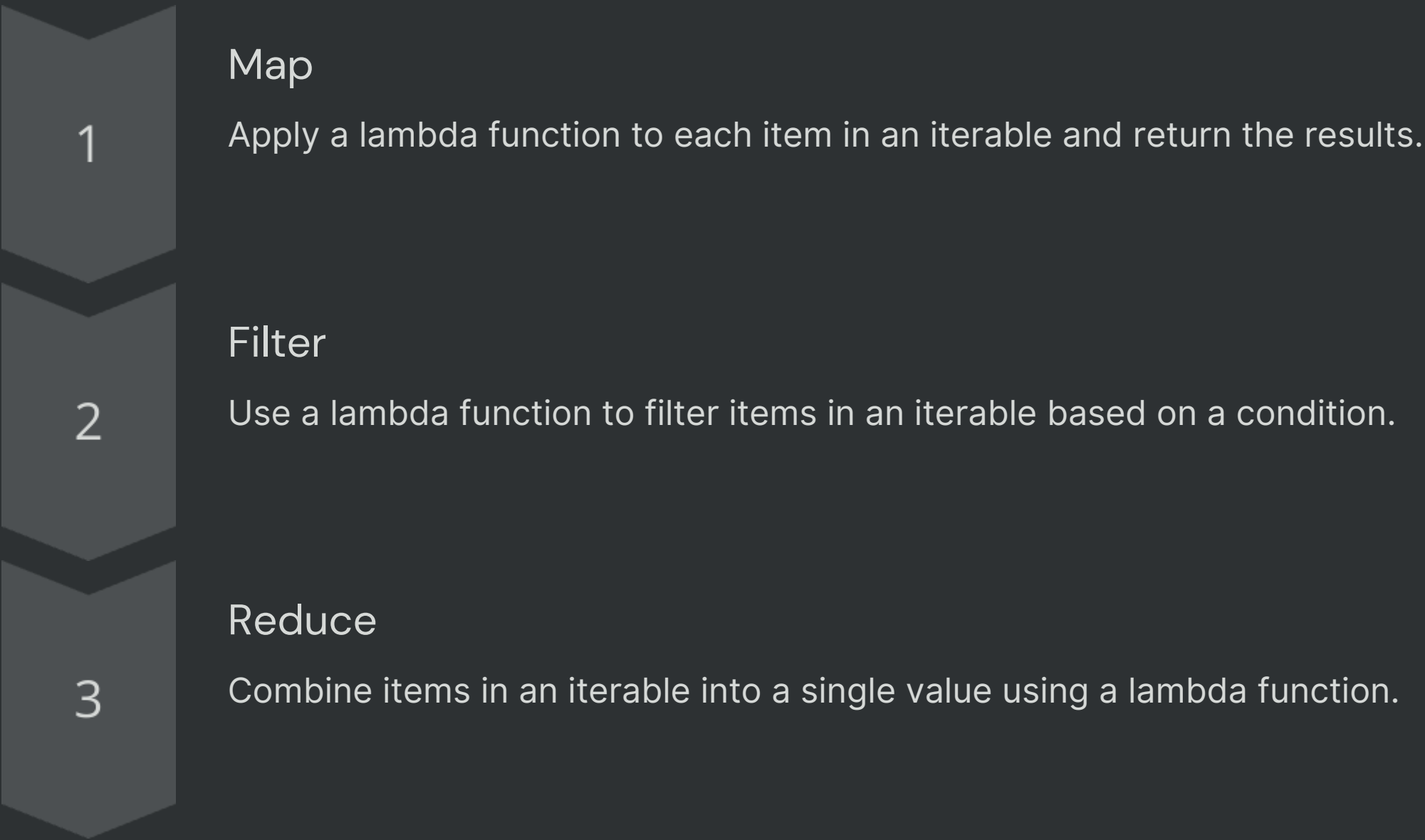
```
``` 25 ```
```



# Using Lambda with Map, Filter, and Reduce

'Map', 'filter', and 'reduce' are higher-order functions that operate on iterables.

Lambda functions are often used with these functions to apply operations efficiently.



## Map has

111.5, 1.101

111.5, 1.600

111.5, 1.69

111.5, 1.89

Lest

Lambda

1. 300

2. 300

1. 350

2. 600

3. 300

2. 680

35.60 :: 170244

35.65 :: 366754

35.60 :: 566395

35.65 :: 862546

33.55 :: 665750

55.65 :: 862614

```
lese tater
besete())
```

```
)
```

```
(lalmbbla_; ≈)
```

# Summary

Functions provide a structured way to organize and reuse code.

Lambda expressions offer a concise syntax for creating anonymous functions, particularly useful for tasks that require simple operations.

$f(x)$

## Functions

Promote code reuse and organization.

$\lambda$

## Lambda Functions

Provide a concise way to create anonymous functions.



## Map, Filter, Reduce

Higher-order functions that work efficiently with lambda expressions.