Structure and Interpretation of Computer Programs with Python

-Presented By: Sean Li

Class Rules

1. Please have your camera on
2. Please stay mute unless you are told not to
3. If you have a question, please use the raise hand function
4. I will post all the resource on the website

https://seanliyucheng.github.io/intro-to-cs/

Let's introduce yourself to your new friends

Name

Age

School

programming background

Etc.

What is programming?

And where is it used?

What is Python?

And what can you do with it?

Use Cases

- Web Development
- Scientific Research
- Software Development
- Most companies use it more or less

Advantages

- Human readable
- Easy to deploy
- Robust Libraries
- Efficiency

and a lot more

What this course is about?

- This course is adapted from a Berkeley computer science course
- An introduction to programming
- Full understanding of Python fundamentals
- Combining multiple ideas in large projects
- How computers interpret programming languages
- THIS IS A CHALLENGING COURSE
- BUT YOU DEFINITELY WILL ACE IT AS LONG AS YOU KEEP TRYING

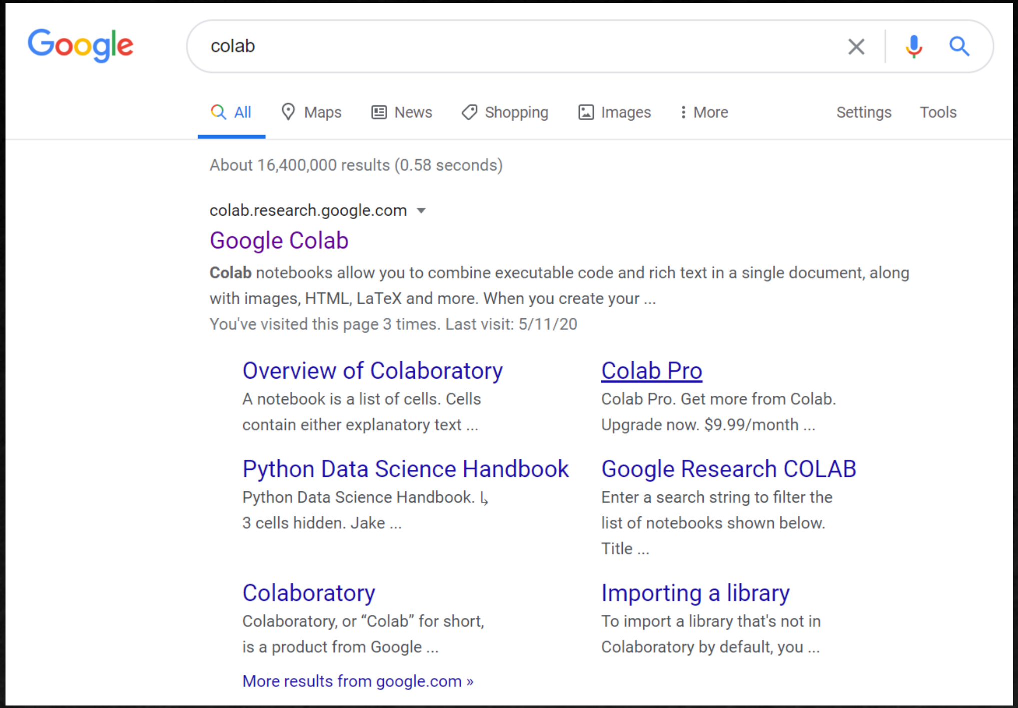
Syllabus

We use Colab to run our code online

https://colab.research.google.com/

1. Functions
2. Names
3. Iteration
4. Control/Loops
5. Mini-Project
6. List, Array
7. Debugging
8. Board Game Fundamentals
9. Object Oriented Programming
10. Project Composition
11. Project Implementation
12. Project Complete

Colab Setup


colab

All · Maps · News · Shopping · Images · More · Settings · Tools

About 16,400,000 results (0.58 seconds)

colab.research.google.com

Google Colab

Colab notebooks allow you to combine executable code and rich text in a single document, along with images, HTML, LaTeX and more. When you create your ...

You've visited this page 3 times. Last visit: 5/11/20

Overview of Colaboratory
A notebook is a list of cells. Cells contain either explanatory text ...

Colab Pro
Colab Pro. Get more from Colab. Upgrade now. $9.99/month ...

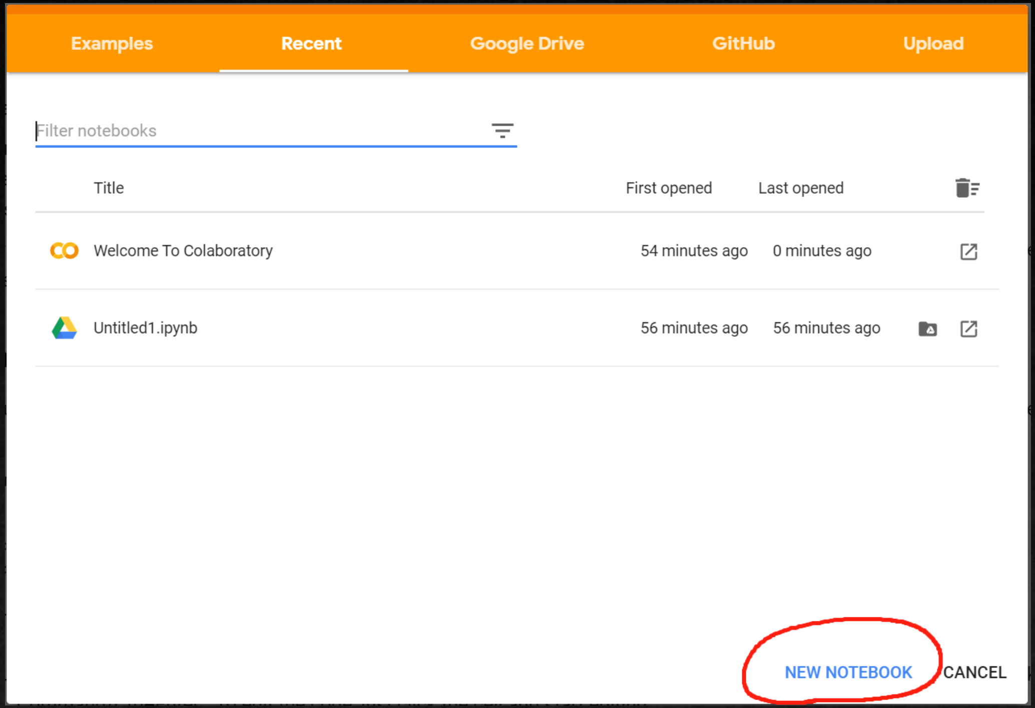
Python Data Science Handbook
Python Data Science Handbook. 3 cells hidden. Jake ...

Google Research COLAB
Enter a search string to filter the list of notebooks shown below. Title ...

Colaboratory
Colaboratory, or "Colab" for short, is a product from Google ...

Importing a library
To import a library that's not in Colaboratory by default, you ...

More results from google.com »



Title	First opened	Last opened
Welcome To Colaboratory	54 minutes ago	0 minutes ago
Untitled1.ipynb	56 minutes ago	56 minutes ago

Just like any other google stuff, it's linked directly to your google account and no installation is required.

Expressions

An expression describes a computation/evaluation and evaluates to a value

$3+5$

2^{10}

add(1, 2)

Exercise

Determine whether the following is an expression or not

2

3*5

F(x)

divide(1, 2)

Call Expression

All expressions can use function call notation

add(1, 2)

How is this evaluated?

Call Expression

$$\underbrace{\text{add}}_{\text{operator}} \ (\ \underbrace{1}_{\text{operand}} \ , \ \underbrace{2}_{\text{operand}} \ )$$

Operators and operands are also expressions so they evaluate to values.

Evaluation Process

1. Evaluate the operator and then the operand subexpressions
2. Apply the function that is the value of the operator to the arguments that are the values of the operands

But sometimes things get complicated…

Nested Expression Evaluation

```
mul(add(4, mul(4, 6)), add(3, 5))
```

Nested Expression Evaluation

```
mul ( add ( 4 , mul ( 4 , 6 )) , add ( 3 , 5 ))
```

Function: mul

Nested Expression Evaluation

mul (add (4 , mul (4 , 6)) , add (3 , 5))

Function: mul

add (4 , mul (4 , 6))

Nested Expression Evaluation

```
mul ( add ( 4 , mul ( 4 , 6 )) , add ( 3 , 5 ))
```

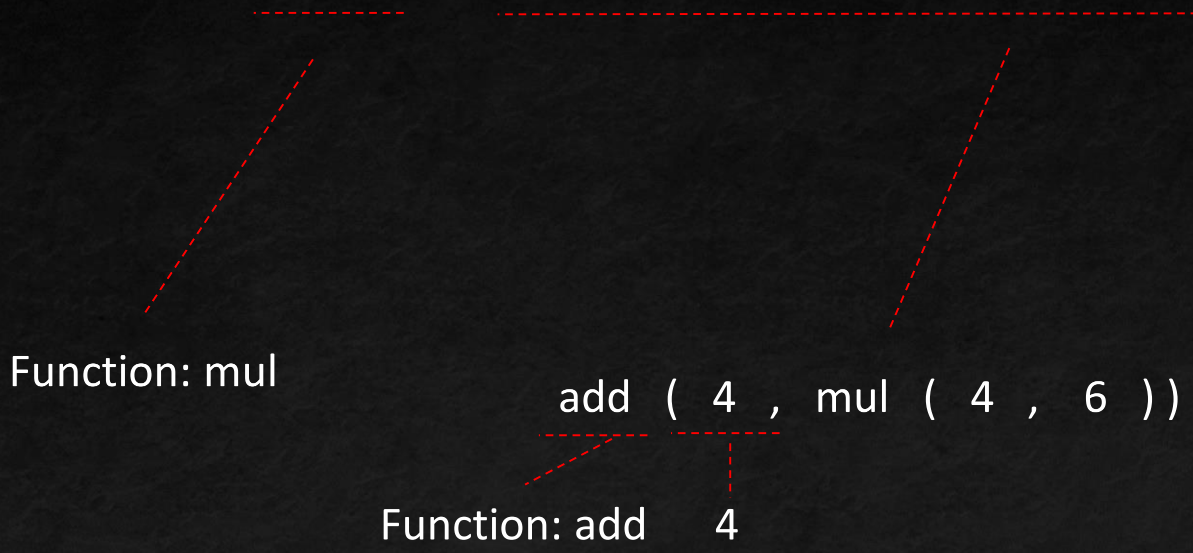
Function: mul

```
add ( 4 , mul ( 4 , 6 ))
```

Function: add

Nested Expression Evaluation

mul (add (4 , mul (4 , 6)) , add (3 , 5))



Nested Expression Evaluation

```
mul ( add ( 4 , mul ( 4 , 6 )) , add ( 3 , 5 ))
```

Function: mul

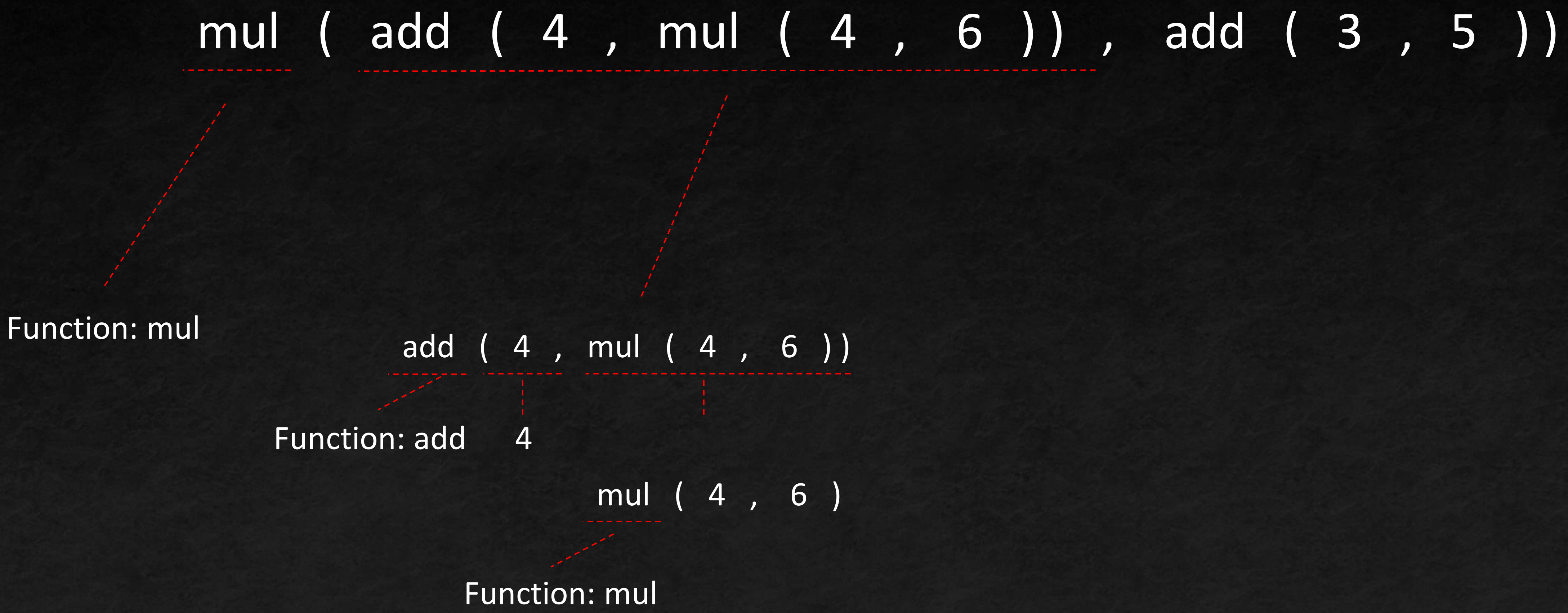
```
add ( 4 , mul ( 4 , 6 ))
```

Function: add

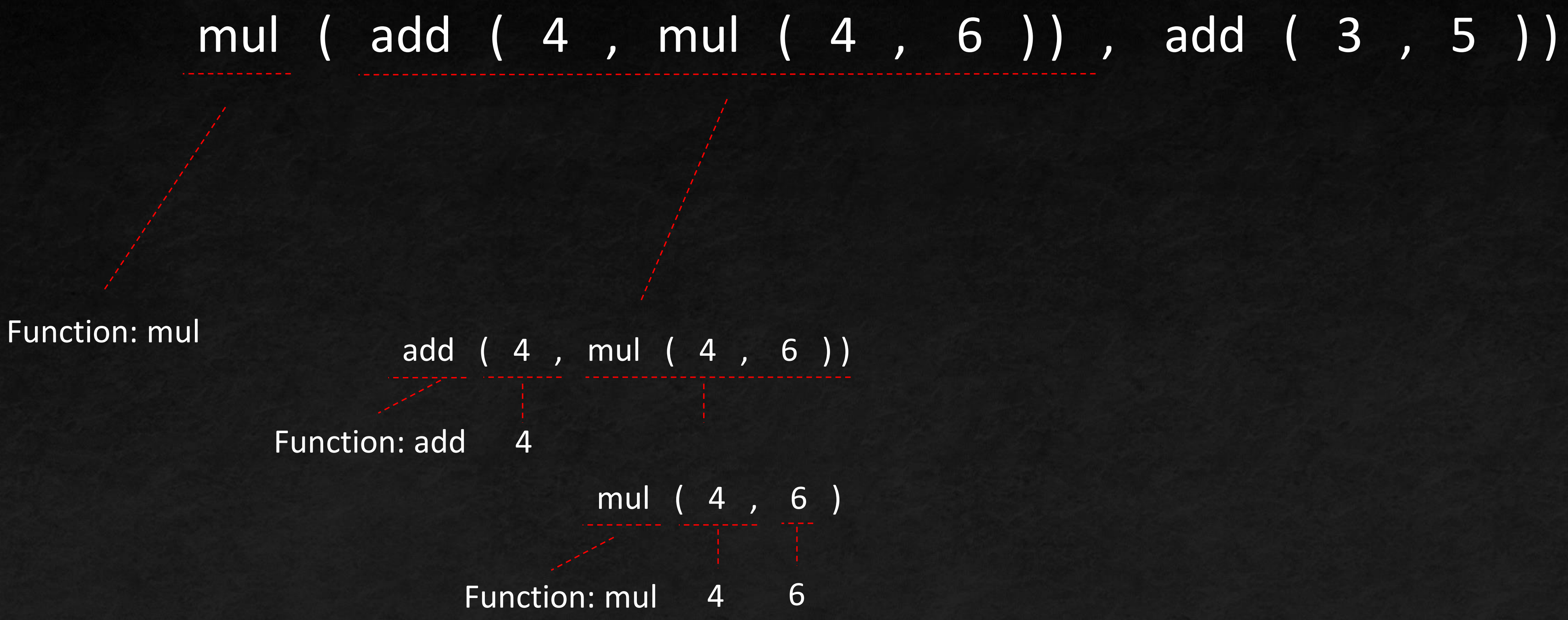
4

```
mul ( 4 , 6 )
```

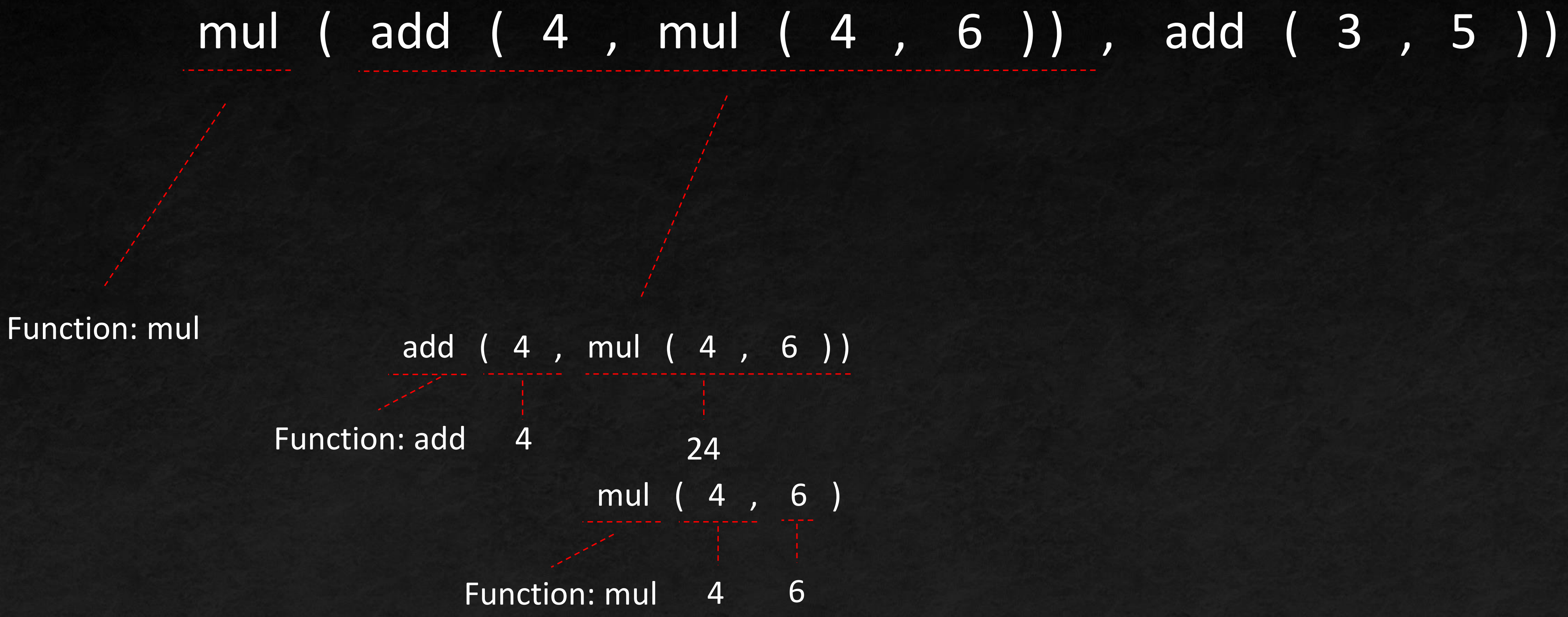
Nested Expression Evaluation



Nested Expression Evaluation

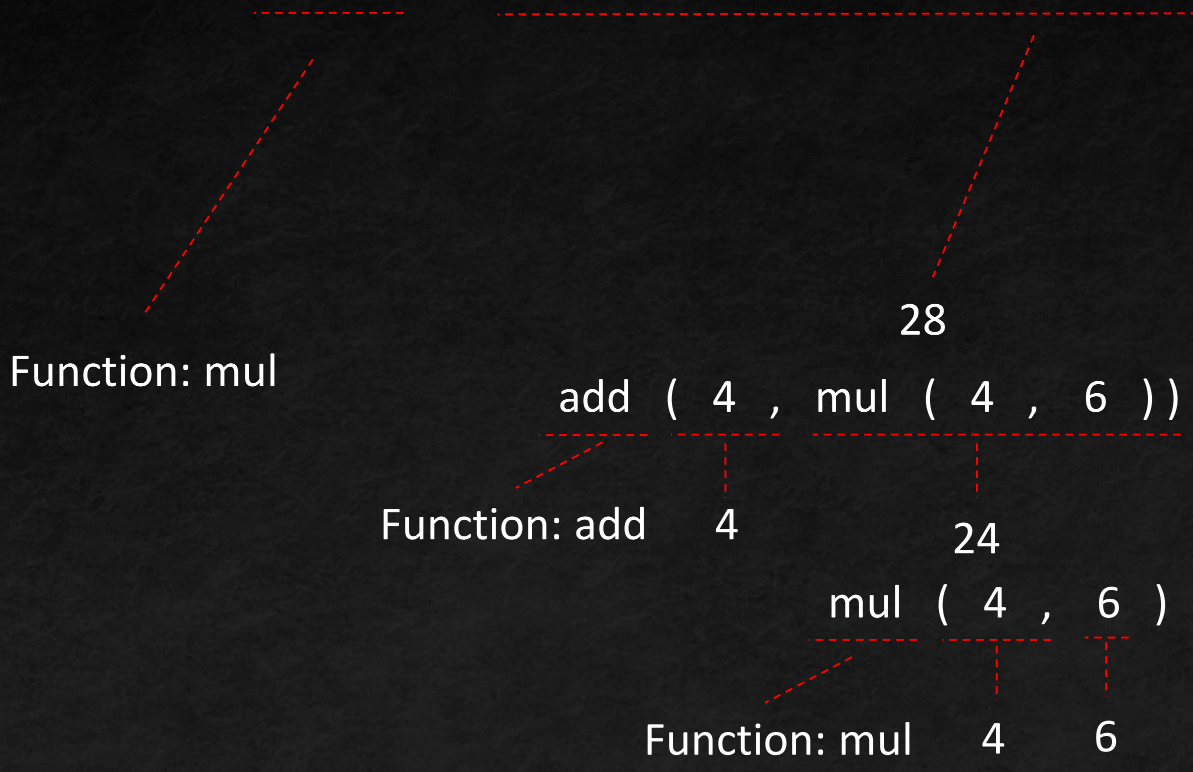


Nested Expression Evaluation

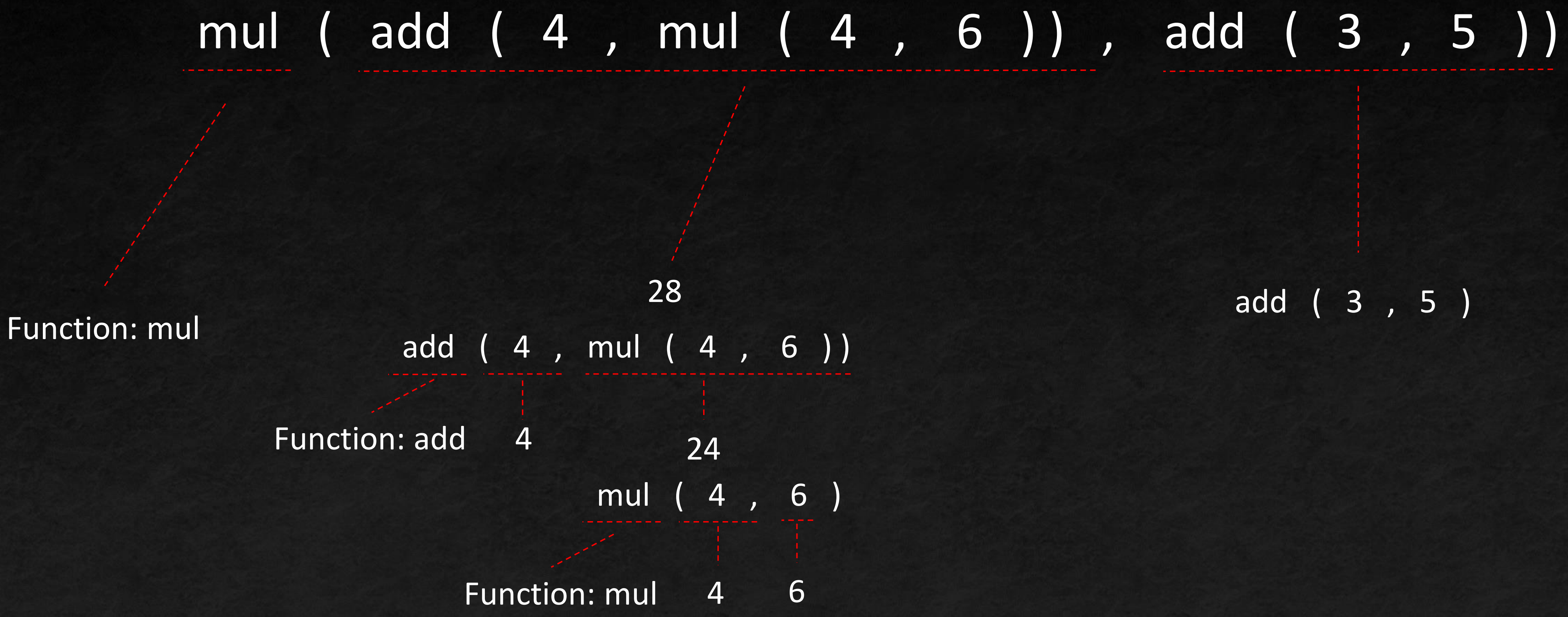


Nested Expression Evaluation

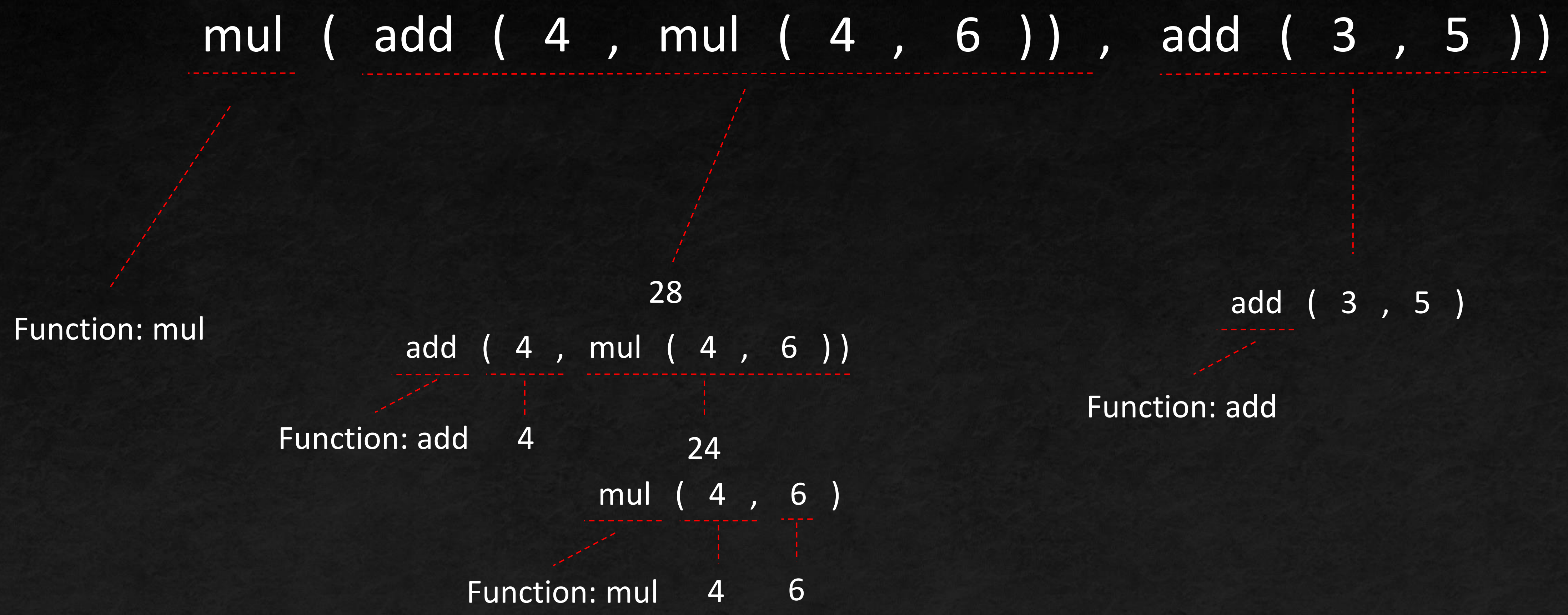
mul (add (4 , mul (4 , 6)) , add (3 , 5))



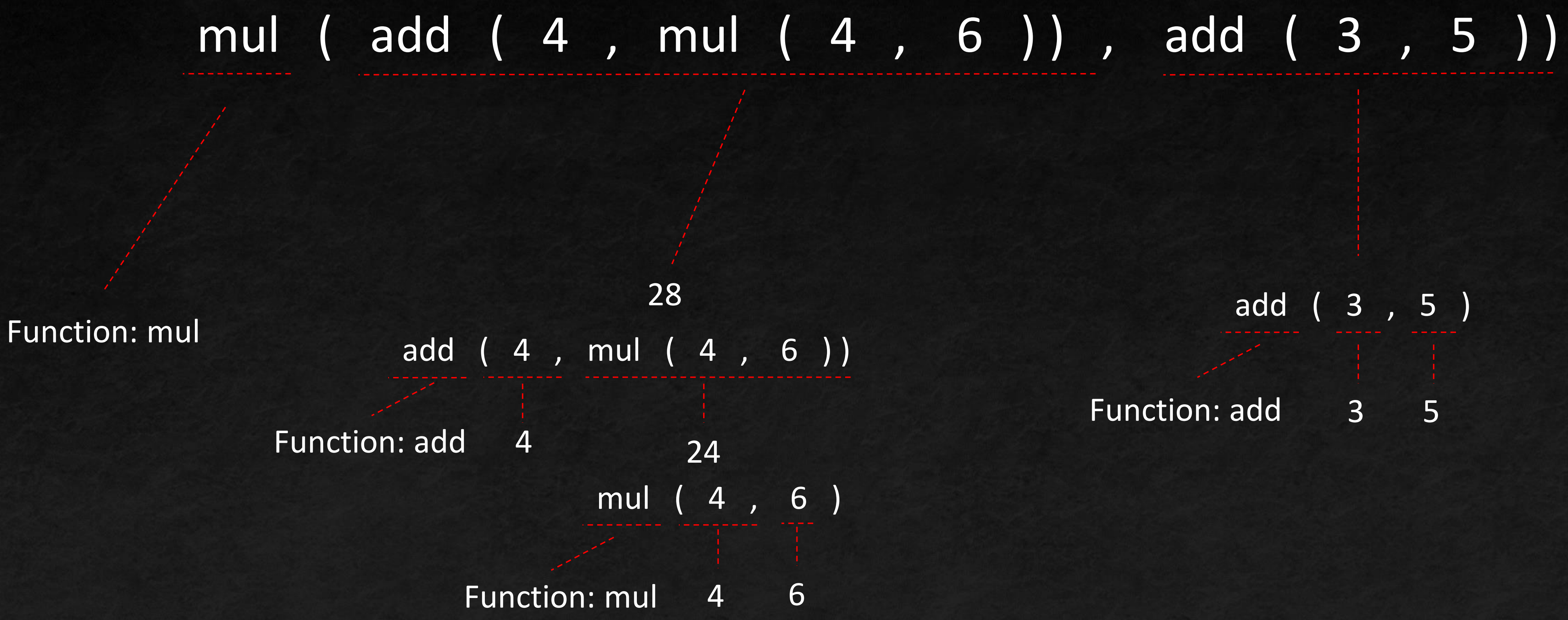
Nested Expression Evaluation


mul (add (4 , mul (4 , 6)) , add (3 , 5))
Function: mul
28
add (4 , mul (4 , 6))
add (3 , 5)
Function: add
4
24
mul (4 , 6)
Function: mul
4
6

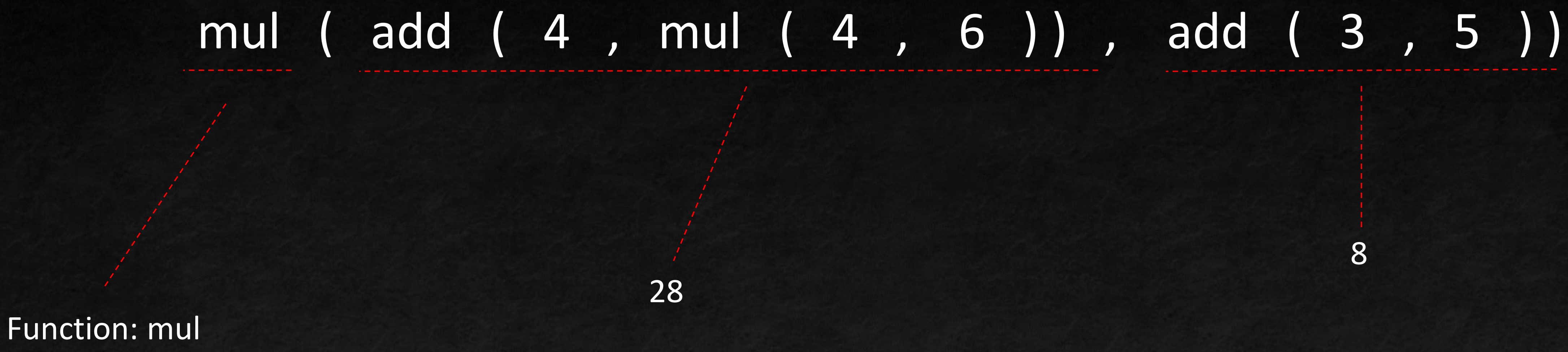
Nested Expression Evaluation



Nested Expression Evaluation



Nested Expression Evaluation



```
mul ( add ( 4 , mul ( 4 , 6 )) , add ( 3 , 5 ))
```

Nested Expression Evaluation

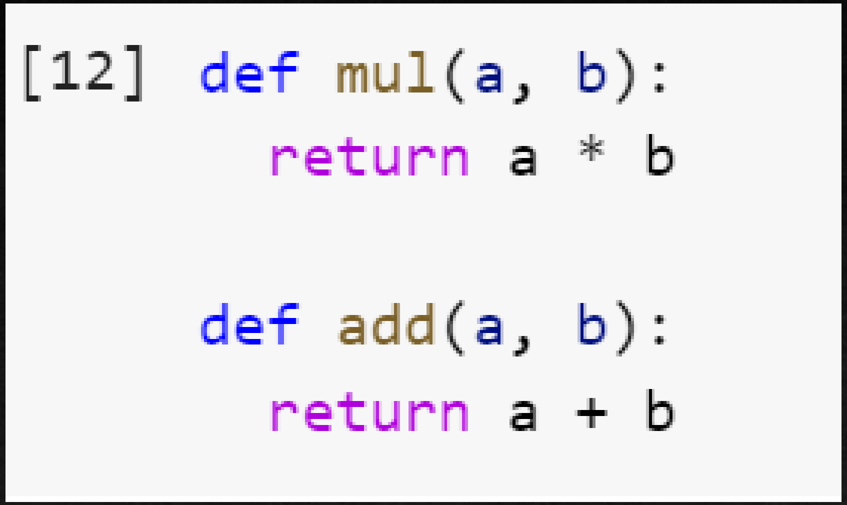
$$\underbrace{\texttt{mul}}_{\text{Function: mul}}\ (\ \underbrace{\texttt{add (4 , mul (4 , 6))}}_{96}\ ,\ \underbrace{\texttt{add (3 , 5)}}_{8}\ )$$

224

Exercise:

```
add (add  (  mul  (  1  ,  2 )  ,  6  )  ,  mul  (  2  ,  5  ))
```

Define a function:

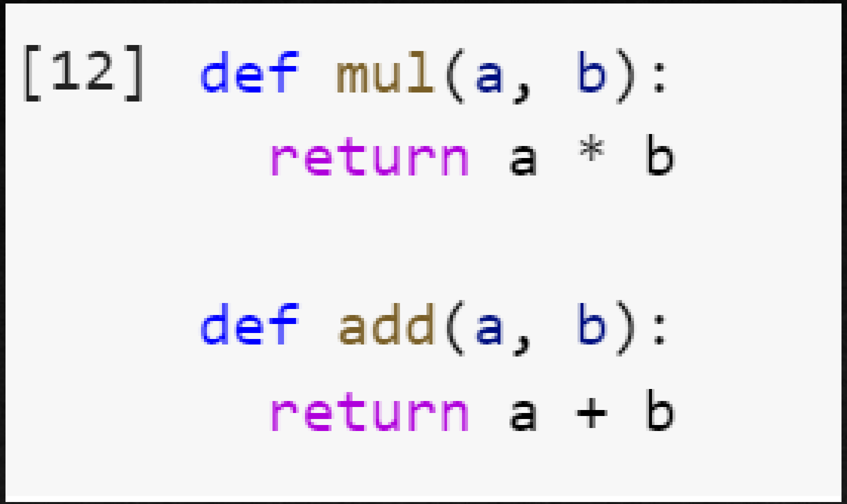


```
[12] def mul(a, b):
       return a * b

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

Do you notice a pattern?

Define a function:



```
[12] def mul(a, b):
       return a * b

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

Do you notice a pattern?

Define a function:

- All complete functions start with a keyword "def"
- Keyword "return" is optional. Whenever you need to return something, your last line should be [return + the stuff you want to return]
- If you don't return, it will return none automatically.

Exercise:

Try to define your own subtraction and division function