Function Examples

Announcements

Review

What Would Python Display?

What Would Python Display?

```
The print function returns None.  It also displays its arguments
(separated by spaces) when it is called.
```

What Would Python Display?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

What Would Python Display?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

This expression	Evaluates to	Interactive Output

What Would Python Display?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

This expression	Evaluates to	Interactive Output
5	5	

What Would Python Display?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

This expression	Evaluates to	Interactive Output
5	5	5

What Would Python Display?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

This expression	Evaluates to	Interactive Output
`5`	5	5
`print(5)`		

What Would Python Display?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

This expression	Evaluates to	Interactive Output
5	5	5
print(5)	None	

What Would Python Display?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

This expression	Evaluates to	Interactive Output
`5`	5	5
`print(5)`	None	5

What Would Python Display?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

This expression	Evaluates to	Interactive Output
`5`	5	5
`print(5)`	None	5
`print(print(5))`		

What Would Python Display?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

This expression	Evaluates to	Interactive Output
`5`	`5`	`5`
`print(5)`	`None`	`5`
`print(print(5))`	`None`	

What Would Python Display?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

This expression	Evaluates to	Interactive Output
`5`	`5`	`5`
`print(5)`	`None`	`5`
`print(print(5))`	`None`	`5` `None`

What Would Python Display?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

This expression	Evaluates to	Interactive Output
`5`	`5`	`5`
`print(5)`	`None`	`5`
`print(print(5))`	`None`	`5` `None`

None

What Would Python Display?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

This expression	Evaluates to	Interactive Output
`5`	`5`	`5`
`print(5)`	`None`	`5`
`print(print(5))`	`None`	`5` `None`

print(print(5)) → None

```
def delay(arg):
    print('delayed')
    def g():
        return arg
    return g
```

What Would Python Display?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

This expression	Evaluates to	Interactive Output
`5`	`5`	`5`
`print(5)`	`None`	`5`
`print(print(5))`	`None`	`5` `None`

`None`

```
def delay(arg):
    print('delayed')
    def g():
        return arg
    return g
```

Names in nested def statements can refer to their enclosing scope

What Would Python Display?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

A function that takes any argument and returns a function that returns that arg

```
def delay(arg):
    print('delayed')
    def g():
        return arg
    return g
```

Names in nested def statements can refer to their enclosing scope

This expression	Evaluates to	Interactive Output
`5`	`5`	`5`
`print(5)`	`None`	`5`
`print(print(5))`	`None`	`5` `None`

`print(print(5))` → `None`

What Would Python Display?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

A function that takes any argument and returns a function that returns that arg

```
def delay(arg):
    print('delayed')
    def g():
        return arg
    return g
```

Names in nested def statements can refer to their enclosing scope

This expression	Evaluates to	Interactive Output
`5`	`5`	`5`
`print(5)`	`None`	`5`
`print(print(5))`	`None`	`5` `None`
`delay(delay)()(6)()`		

`print(5)` → None

What Would Python Display?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

A function that takes any argument and returns a function that returns that arg

```
def delay(arg):
    print('delayed')
    def g():
        return arg
    return g
```

Names in nested def statements can refer to their enclosing scope

This expression	Evaluates to	Interactive Output
`5`	`5`	`5`
`print(5)`	`None`	`5`
`print(print(5))`	`None`	`5` `None`
`delay(delay)()(6)()`		

What Would Python Display?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

A function that takes any argument and returns a function that returns that arg

```
def delay(arg):
    print('delayed')
    def g():
        return arg
    return g
```

Names in nested def statements can refer to their enclosing scope

This expression	Evaluates to	Interactive Output
`5`	`5`	`5`
`print(5)`	`None`	`5`
`print(print(5))`	`None`	`5` `None`
`delay(delay)()(6)()`		

What Would Python Display?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

A function that takes any argument and returns a function that returns that arg

```
def delay(arg):
    print('delayed')
    def g():
        return arg
    return g
```

Names in nested def statements can refer to their enclosing scope

This expression	Evaluates to	Interactive Output
`5`	`5`	`5`
`print(5)`	`None`	`5`
`print(print(5))`	`None`	`5` `None`
`delay(delay)()(6)()`		

What Would Python Display?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

A function that takes any argument and returns a function that returns that arg

```
def delay(arg):
    print('delayed')
    def g():
        return arg
    return g
```

Names in nested def statements can refer to their enclosing scope

This expression	Evaluates to	Interactive Output
`5`	`5`	`5`
`print(5)`	`None`	`5`
`print(print(5))`	`None`	`5` `None`
`delay(delay)()(6)()`		

What Would Python Display?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

A function that takes any argument and returns a function that returns that arg

```
def delay(arg):
    print('delayed')
    def g():
        return arg
    return g
```

Names in nested def statements can refer to their enclosing scope

This expression	Evaluates to	Interactive Output
`5`	`5`	`5`
`print(5)`	`None`	`5`
`print(print(5))`	`None`	`5` `None`
`delay(delay)()(6)()`		`delayed`

What Would Python Display?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

A function that takes any argument and returns a function that returns that arg

```
def delay(arg):
    print('delayed')
    def g():
        return arg
    return g
```

Names in nested def statements can refer to their enclosing scope

This expression	Evaluates to	Interactive Output
`5`	`5`	`5`
`print(5)`	`None`	`5`
`print(print(5))`	`None`	`5` `None`
`delay(delay)()(6)()`		`delayed` `delayed`

None

What Would Python Display?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

A function that takes any argument and returns a function that returns that arg

```
def delay(arg):
    print('delayed')
    def g():
        return arg
    return g
```

Names in nested def statements can refer to their enclosing scope

This expression	Evaluates to	Interactive Output
`5`	`5`	`5`
`print(5)`	`None`	`5`
`print(print(5))`	`None`	`5` `None`

`delay(delay)()(6)()`		`delayed`
		`delayed`
		`6`

None

What Would Python Display?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

A function that takes any argument and returns a function that returns that arg

```
def delay(arg):
    print('delayed')
    def g():
        return arg
    return g
```

Names in nested def statements can refer to their enclosing scope

This expression	Evaluates to	Interactive Output
`5`	`5`	`5`
`print(5)`	`None`	`5`
`print(print(5))`	`None`	`5` `None`

`delay(delay)()(6)()`	`6`	`delayed`
		`delayed`
		`6`

None

What Would Python Display?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

A function that takes any argument and returns a function that returns that arg

```
def delay(arg):
    print('delayed')
    def g():
        return arg
    return g
```

Names in nested def statements can refer to their enclosing scope

This expression	Evaluates to	Interactive Output
`5`	`5`	`5`
`print(5)`	`None`	`5`
`print(print(5))`	`None`	`5` `None`

`delay(delay)()(6)()`	`6`	`delayed`
		`delayed`
		`6`

`print(delay(print)()(4))`		

What Would Python Display?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

A function that takes any argument and returns a function that returns that arg

```
def delay(arg):
    print('delayed')
    def g():
        return arg
    return g
```

Names in nested def statements can refer to their enclosing scope

This expression	Evaluates to	Interactive Output
`5`	`5`	`5`
`print(5)`	`None`	`5`
`print(print(5))`	`None`	`5` `None`

`delay(delay)()(6)()`	`6`	`delayed`
		`delayed`
		`6`

`print(delay(print)()(4))`		`delayed`

None

What Would Python Display?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

A function that takes any argument and returns a function that returns that arg

```
def delay(arg):
    print('delayed')
    def g():
        return arg
    return g
```

Names in nested def statements can refer to their enclosing scope

This expression	Evaluates to	Interactive Output
`5`	`5`	`5`
`print(5)`	`None`	`5`
`print(print(5))`	`None`	`5` `None`

`delay(delay)()(6)()`	`6`	`delayed`
		`delayed`
		`6`

`print(delay(print)()(4))`		`delayed` `4`

None

What Would Python Display?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

A function that takes any argument and returns a function that returns that arg

```
def delay(arg):
    print('delayed')
    def g():
        return arg
    return g
```

Names in nested def statements can refer to their enclosing scope

This expression	Evaluates to	Interactive Output
`5`	`5`	`5`
`print(5)`	`None`	`5`
`print(print(5))`	`None`	`5` `None`

`delay(delay)()(6)()`	`6`	`delayed`
		`delayed`
		`6`

`print(delay(print)()(4))`		`delayed`
		`4`
		`None`

None

What Would Python Display?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

A function that takes any argument and returns a function that returns that arg

```
def delay(arg):
    print('delayed')
    def g():
        return arg
    return g
```

Names in nested def statements can refer to their enclosing scope

This expression	Evaluates to	Interactive Output
`5`	`5`	`5`
`print(5)`	`None`	`5`
`print(print(5))`	`None`	`5` `None`

`delay(delay)()(6)()`	`6`	`delayed`
		`delayed`
		`6`

`print(delay(print)()(4))`	`None`	`delayed`
		`4`
		`None`

None

What Would Python Print?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

This expression	Evaluates to	Interactive Output

What Would Python Print?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

This expression	Evaluates to	Interactive Output

```
def pirate(arggg):
    print('matey')
    def plunder(arggg):
        return arggg
    return plunder
```

What Would Python Print?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

```
def pirate(arggg):
    print('matey')
    def plunder(arggg):
        return arggg
    return plunder
```

This expression	Evaluates to	Interactive Output
`add(pirate(3)(square)(4), 1)`		

What Would Python Print?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

This expression	Evaluates to	Interactive Output
`add(pirate(3)(square)(4), 1)`		

```
def pirate(arggg):
    print('matey')
    def plunder(arggg):
        return arggg
    return plunder
```

A name evaluates to the value bound to that name in the earliest frame of the current environment in which that name is found.

What Would Python Print?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

A function that always returns the identity function

```
def pirate(arggg):
    print('matey')
    def plunder(arggg):
        return arggg
    return plunder
```

This expression	Evaluates to	Interactive Output
`add(pirate(3)(square)(4), 1)`		

A name evaluates to the value bound to that name in the earliest frame of the current environment in which that name is found.

What Would Python Print?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

A function that always returns the identity function

```
def pirate(arggg):
    print('matey')
    def plunder(arggg):
        return arggg
    return plunder
```

This expression	Evaluates to	Interactive Output
`add(pirate(3)(square)(4), 1)`		

A name evaluates to the value bound to that name in the earliest frame of the current environment in which that name is found.

What Would Python Print?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

A function that always returns the identity function

```
def pirate(arggg):
    print('matey')
    def plunder(arggg):
        return arggg
    return plunder
```

This expression	Evaluates to	Interactive Output
`add(pirate(3)(square)(4), 1)`		Matey

A name evaluates to the value bound to that name in the earliest frame of the current environment in which that name is found.

What Would Python Print?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)

def pirate(arggg):
    print('matey')
    def plunder(arggg):
        return arggg
    return plunder
```

A function that always returns the identity function

This expression	Evaluates to	Interactive Output
`add(pirate(3)(square)(4), 1)`		Matey

A name evaluates to the value bound to that name in the earliest frame of the current environment in which that name is found.

What Would Python Print?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

A function that always returns the identity function

```
def pirate(arggg):
    print('matey')
    def plunder(arggg):
        return arggg
    return plunder
```

This expression	Evaluates to	Interactive Output
`add(pirate(3)(square)(4), 1)`		Matey
`pirate(3)`	*func square(x)*	

A name evaluates to the value bound to that name in the earliest frame of the current environment in which that name is found.

What Would Python Print?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

A function that always returns the identity function

```
def pirate(arggg):
    print('matey')
    def plunder(arggg):
        return arggg
    return plunder
```

This expression	Evaluates to	Interactive Output
`add(pirate(3)(square)(4), 1)`		Matey
func square(x)		

A name evaluates to the value bound to that name in the earliest frame of the current environment in which that name is found.

What Would Python Print?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

A function that always returns the identity function

```
def pirate(arggg):
    print('matey')
    def plunder(arggg):
        return arggg
    return plunder
```

This expression	Evaluates to	Interactive Output
`add(pirate(3)(square)(4), 1)`		Matey
`pirate(3)(square)`	*func square(x)*	
`pirate(3)(square)(4)`	*16*	

A name evaluates to the value bound to that name in the earliest frame of the current environment in which that name is found.

What Would Python Print?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

A function that always returns the identity function

```
def pirate(arggg):
    print('matey')
    def plunder(arggg):
        return arggg
    return plunder
```

This expression	Evaluates to	Interactive Output
`add(pirate(3)(square)(4), 1)`		Matey 17
`pirate(3)`	*func square(x)*	
`pirate(3)(square)(4)`	*16*	

A name evaluates to the value bound to that name in the earliest frame of the current environment in which that name is found.

What Would Python Print?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

A function that always returns the identity function

```
def pirate(arggg):
    print('matey')
    def plunder(arggg):
        return arggg
    return plunder
```

This expression	Evaluates to	Interactive Output
`add(pirate(3)(square)(4), 1)`	17	Matey 17

func square(x)

16

A name evaluates to the value bound to that name in the earliest frame of the current environment in which that name is found.

What Would Python Print?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

A function that always returns the identity function

```
def pirate(arggg):
    print('matey')
    def plunder(arggg):
        return arggg
    return plunder
```

This expression	Evaluates to	Interactive Output
`add(pirate(3)(square)(4), 1)` *func square(x)* *16*	17	Matey 17
`pirate(pirate(pirate))(5)(7)`		

A name evaluates to the value bound to that name in the earliest frame of the current environment in which that name is found.

What Would Python Print?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

A function that always returns the identity function

```
def pirate(arggg):
    print('matey')
    def plunder(arggg):
        return arggg
    return plunder
```

This expression	Evaluates to	Interactive Output
`add(pirate(3)(square)(4), 1)`	17	Matey 17
`pirate(pirate(pirate))(5)(7)`		

func square(x)

16

A name evaluates to the value bound to that name in the earliest frame of the current environment in which that name is found.

What Would Python Print?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

A function that always returns the identity function

```
def pirate(arggg):
    print('matey')
    def plunder(arggg):
        return arggg
    return plunder
```

This expression	Evaluates to	Interactive Output
`add(pirate(3)(square)(4), 1)`	17	Matey 17
func square(x)		
16		
`pirate(pirate(pirate))(5)(7)`		
Identity function		

A name evaluates to the value bound to that name in the earliest frame of the current environment in which that name is found.

What Would Python Print?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

A function that always returns the identity function

```
def pirate(arggg):
    print('matey')
    def plunder(arggg):
        return arggg
    return plunder
```

This expression	Evaluates to	Interactive Output
`add(pirate(3)(square)(4), 1)` *func square(x)* *16*	17	Matey 17
`pirate(pirate(pirate))(5)(7)` *Identity function*		Matey

A name evaluates to the value bound to that name in the earliest frame of the current environment in which that name is found.

What Would Python Print?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

A function that always returns the identity function

```
def pirate(arggg):
    print('matey')
    def plunder(arggg):
        return arggg
    return plunder
```

This expression	Evaluates to	Interactive Output
`add(pirate(3)(square)(4), 1)`	17	Matey 17
`pirate(pirate(pirate))(5)(7)`		Matey Matey

func square(x)

16

Identity function

A name evaluates to the value bound to that name in the earliest frame of the current environment in which that name is found.

What Would Python Print?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

A function that always returns the identity function

```
def pirate(arggg):
    print('matey')
    def plunder(arggg):
        return arggg
    return plunder
```

This expression	Evaluates to	Interactive Output
`add(pirate(3)(square)(4), 1)`	17	Matey 17
`pirate(pirate(pirate))(5)(7)`		Matey Matey

func square(x)

16

Identity function

A name evaluates to the value bound to that name in the earliest frame of the current environment in which that name is found.

What Would Python Print?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

A function that always returns the identity function

```
def pirate(arggg):
    print('matey')
    def plunder(arggg):
        return arggg
    return plunder
```

This expression	Evaluates to	Interactive Output
`add(pirate(3)(square)(4), 1)`	17	Matey 17
`pirate(pirate(pirate))(5)(7)`		Matey Matey

func square(x)

16

Identity function

5

A name evaluates to the value bound to that name in the earliest frame of the current environment in which that name is found.

What Would Python Print?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

A function that always returns the identity function

```
def pirate(arggg):
    print('matey')
    def plunder(arggg):
        return arggg
    return plunder
```

This expression	Evaluates to	Interactive Output
`add(pirate(3)(square)(4), 1)` *func square(x)* *16*	17	Matey 17
`pirate(pirate(pirate))(5)(7)` *Identity function* *5*		Matey Matey Error

A name evaluates to the value bound to that name in the earliest frame of the current environment in which that name is found.

What Would Python Print?

The print function returns None. It also displays its arguments (separated by spaces) when it is called.

```
from operator import add, mul
def square(x):
    return mul(x, x)
```

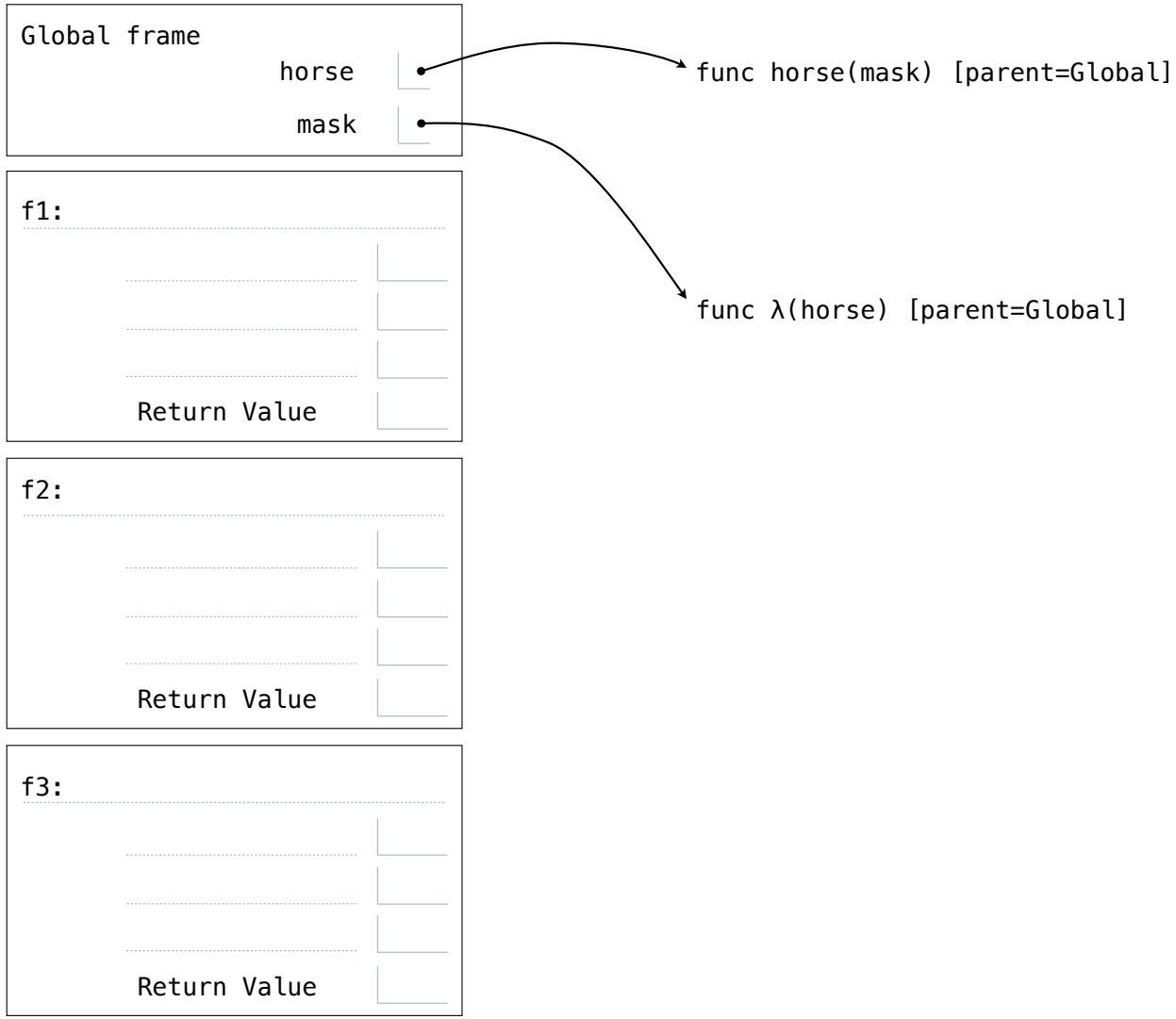
A function that always returns the identity function

```
def pirate(arggg):
    print('matey')
    def plunder(arggg):
        return arggg
    return plunder
```

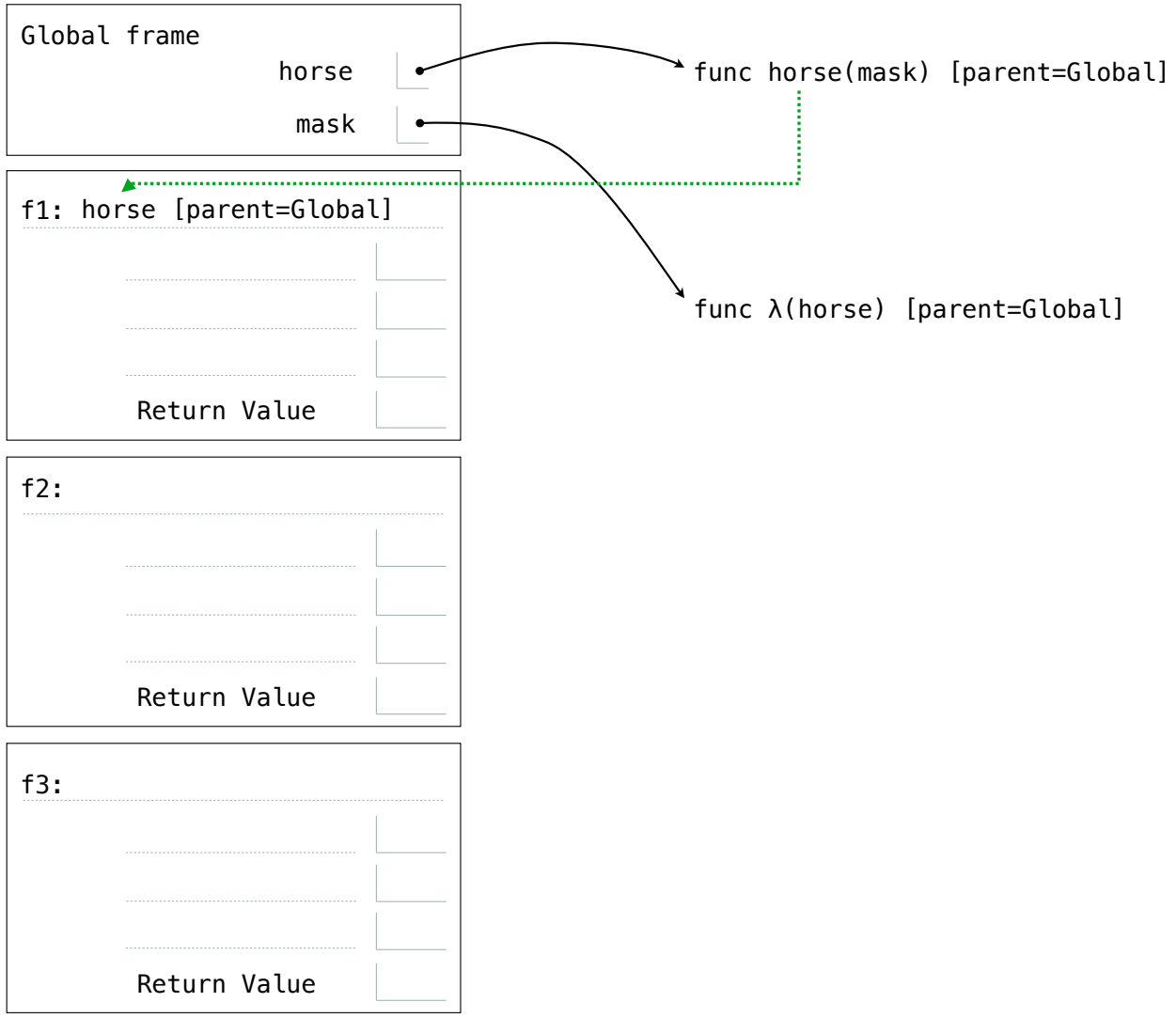
This expression	Evaluates to	Interactive Output
`add(pirate(3)(square)(4), 1)` *func square(x)* *16*	17	Matey 17
`pirate(pirate(pirate))(5)(7)` *Identity function* *5*	Error	Matey Matey Error

A name evaluates to the value bound to that name in the earliest frame of the current environment in which that name is found.

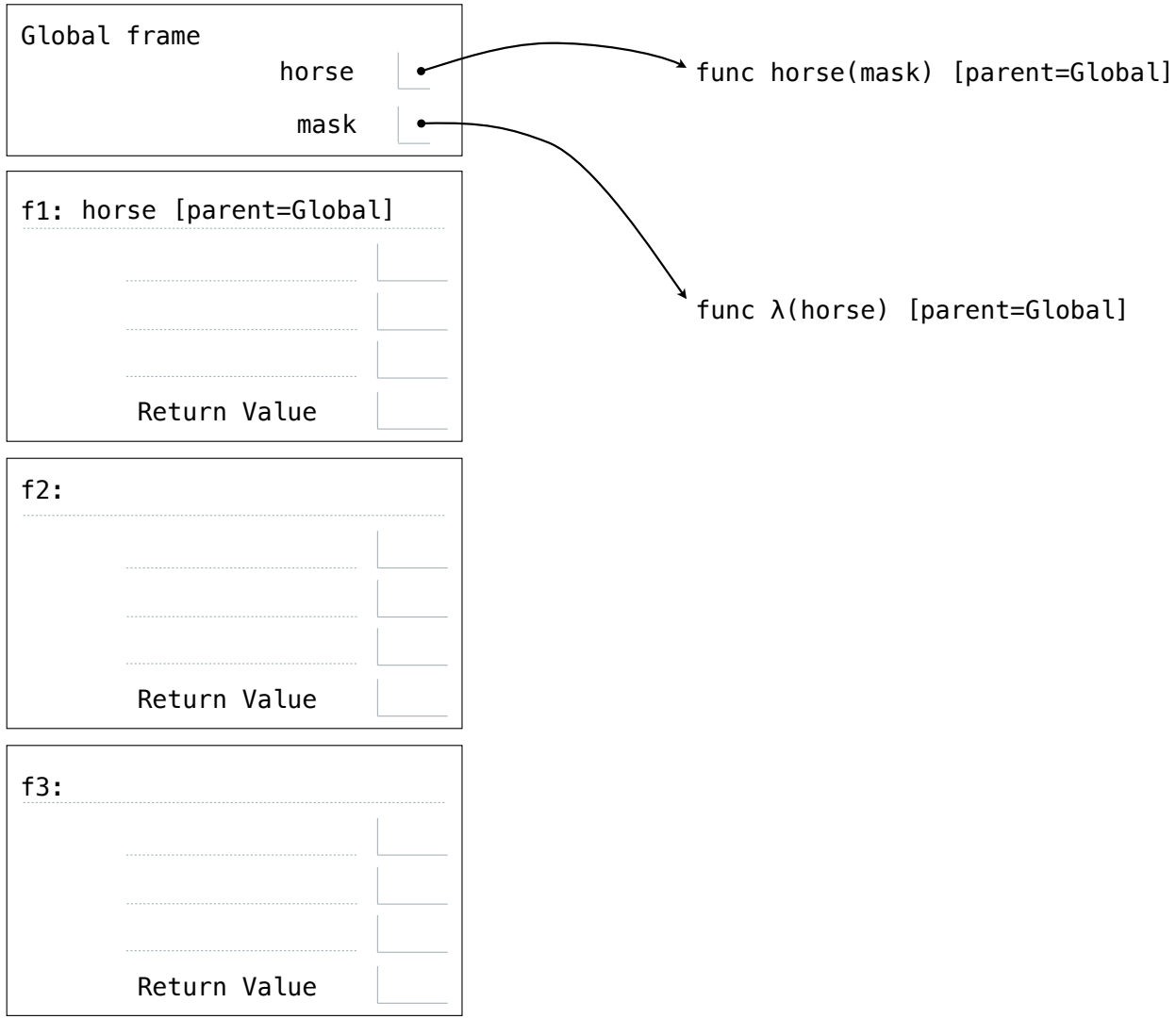
```
def horse(mask):
    horse = mask
    def mask(horse):
        return horse
    return horse(mask)

mask = lambda horse: horse(2)

horse(mask)
```



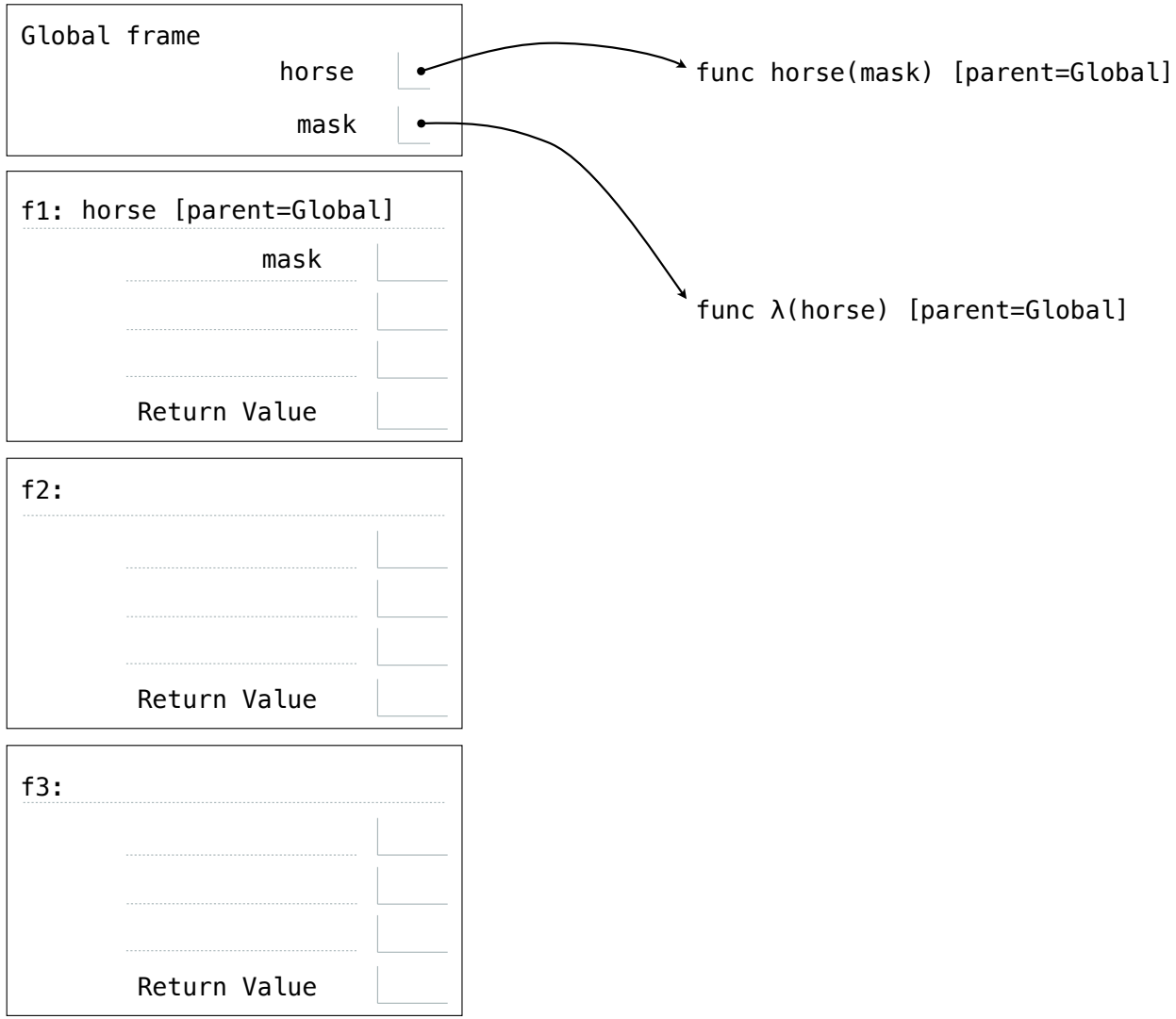
```
def horse(mask):
    horse = mask
    def mask(horse):
        return horse
    return horse(mask)

mask = lambda horse: horse(2)

horse(mask)
```



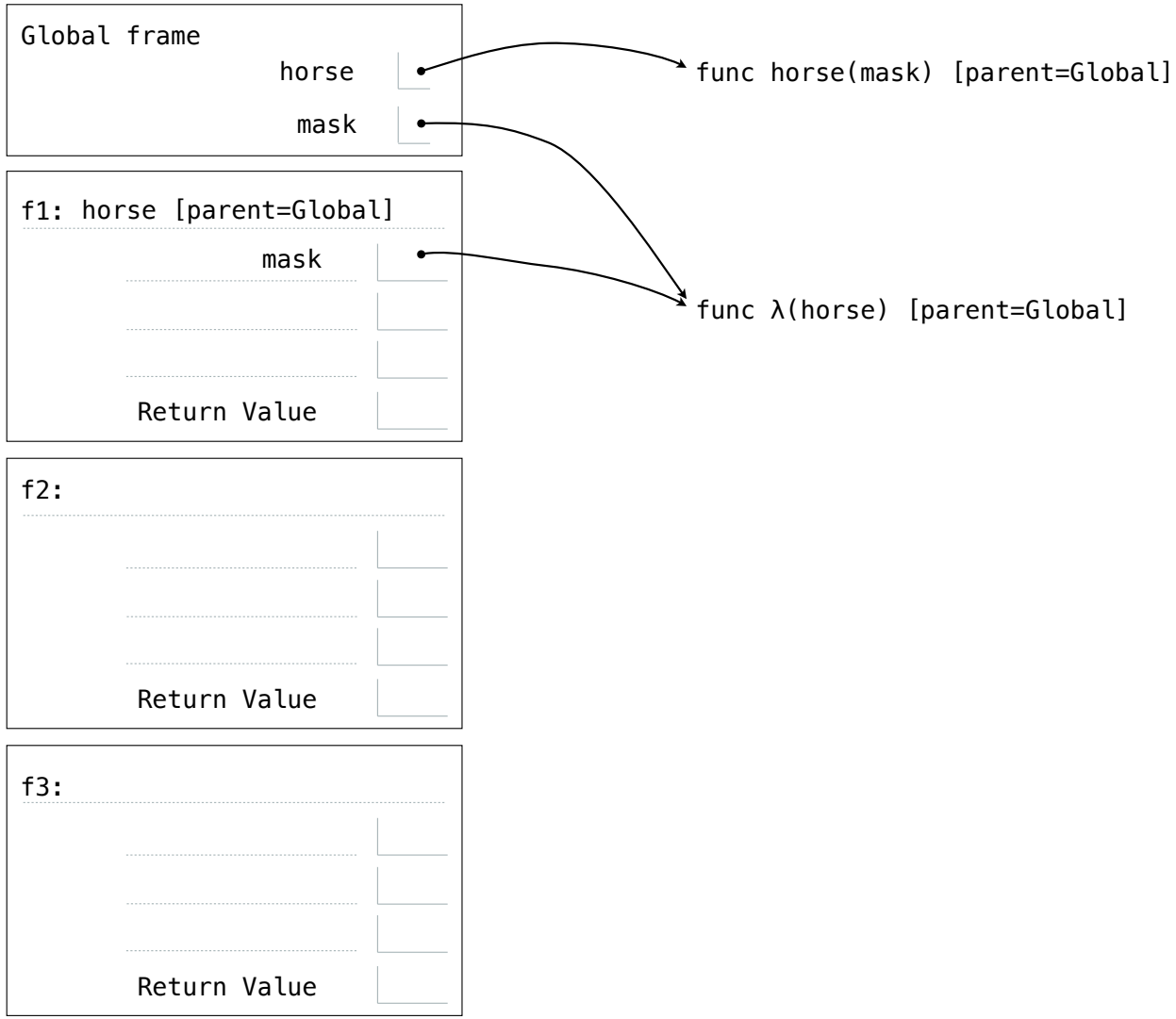
```
def horse(mask):
    horse = mask
    def mask(horse):
        return horse
    return horse(mask)

mask = lambda horse: horse(2)

horse(mask)
```



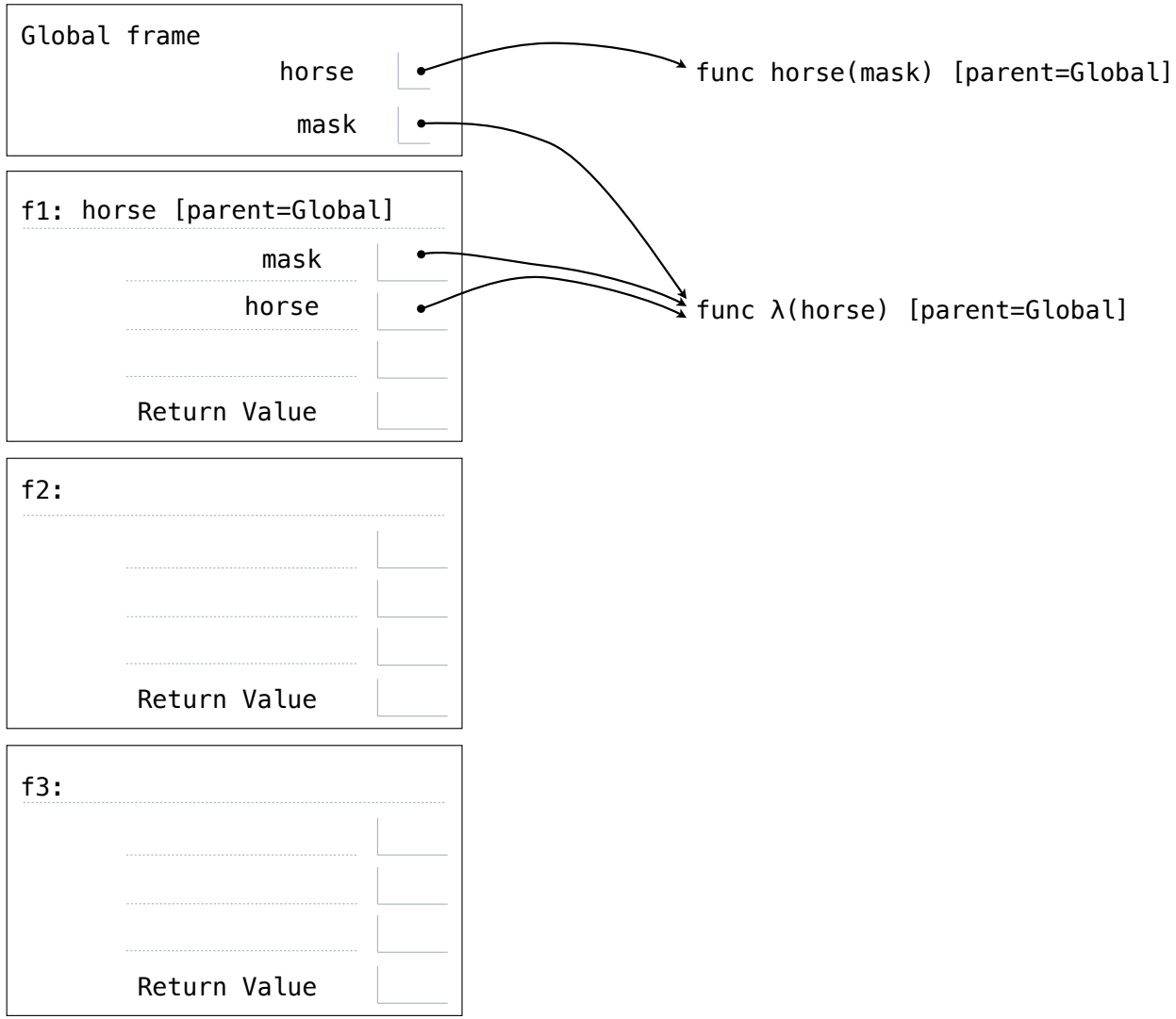
```
def horse(mask):
    horse = mask
    def mask(horse):
        return horse
    return horse(mask)

mask = lambda horse: horse(2)

horse(mask)
```



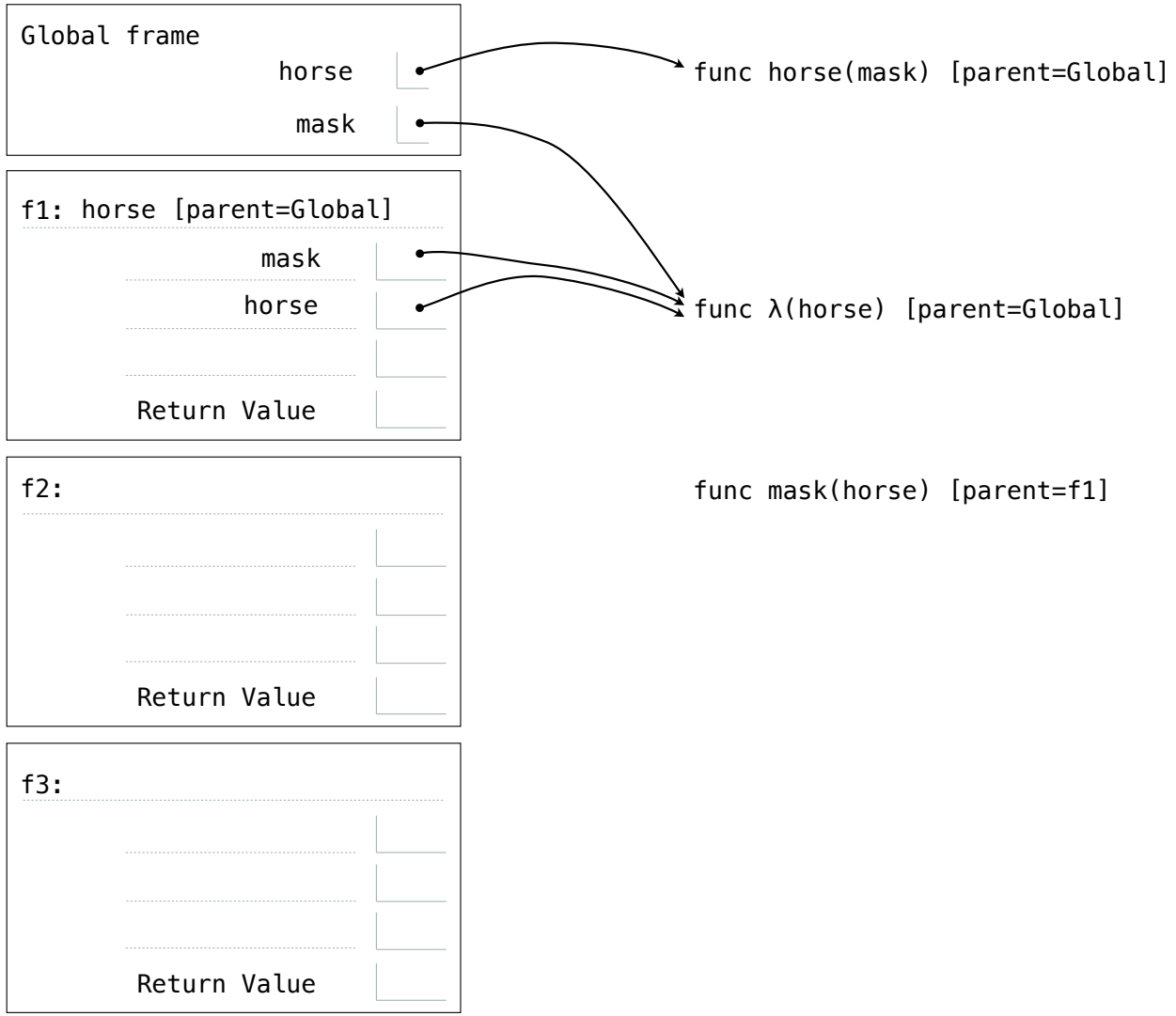
```
def horse(mask):
    horse = mask
    def mask(horse):
        return horse
    return horse(mask)

mask = lambda horse: horse(2)

horse(mask)
```



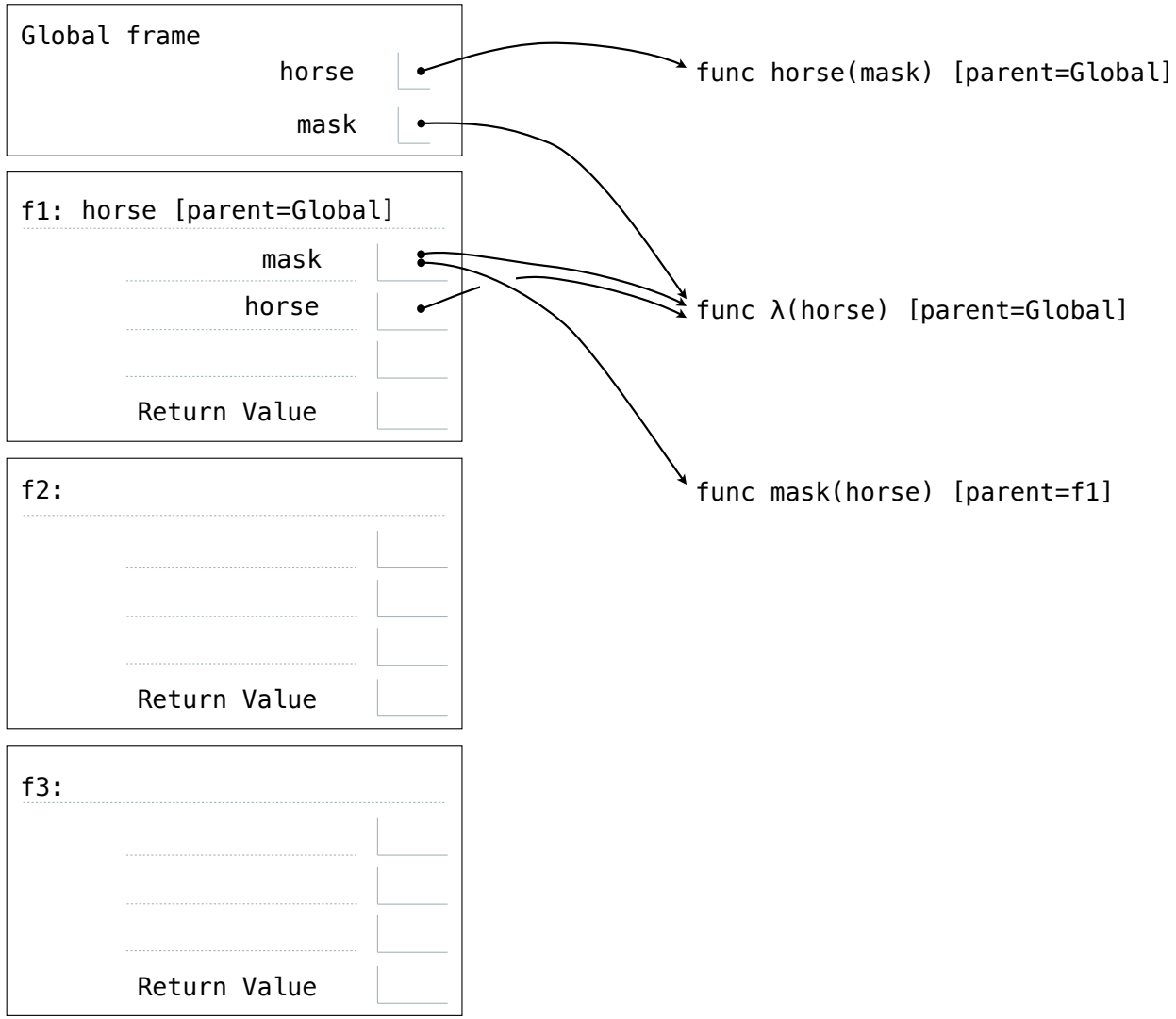
```
def horse(mask):
    horse = mask
    def mask(horse):
        return horse
    return horse(mask)

mask = lambda horse: horse(2)

horse(mask)
```



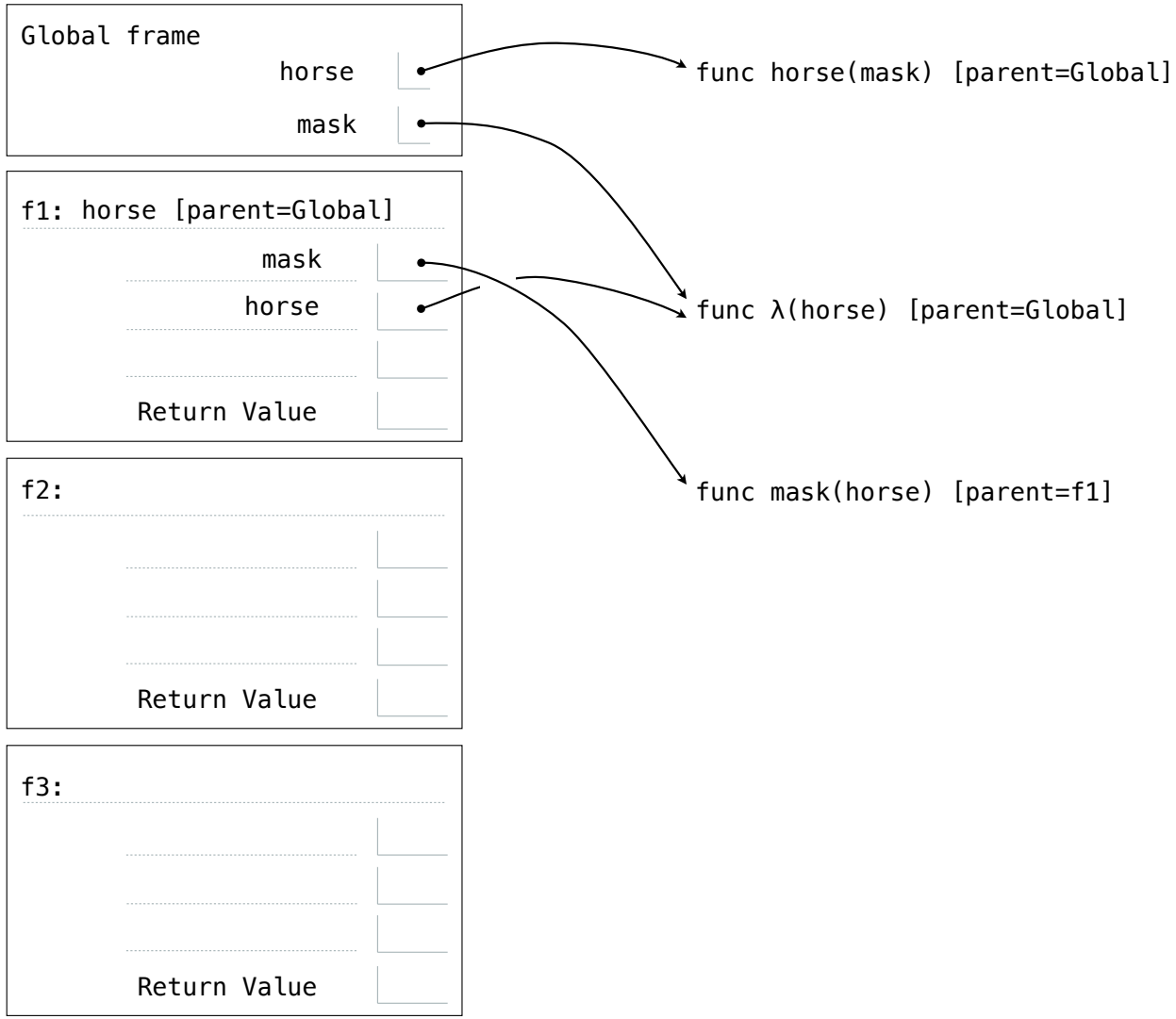
```
def horse(mask):
    horse = mask
    def mask(horse):
        return horse
    return horse(mask)

mask = lambda horse: horse(2)

horse(mask)
```



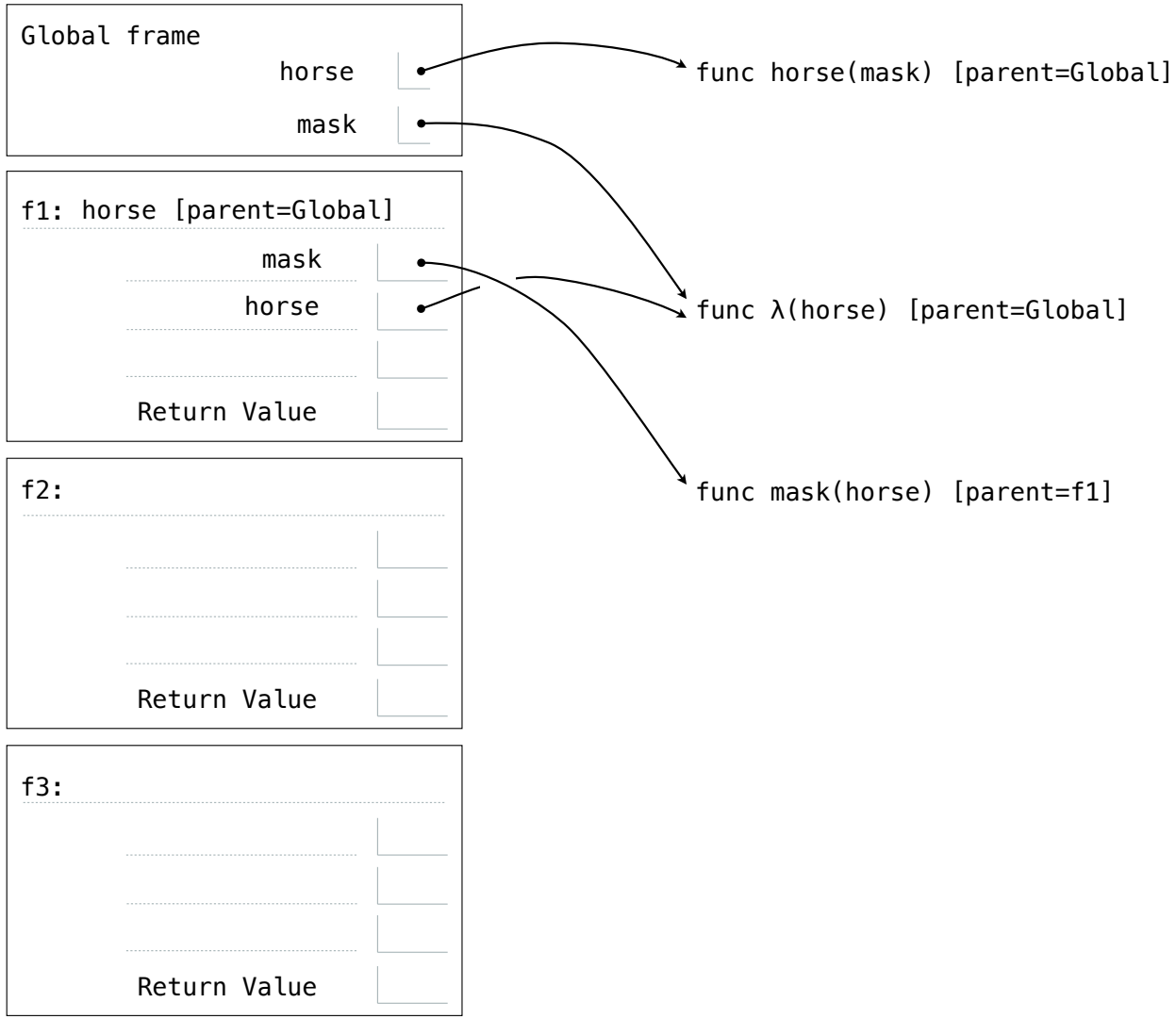
```
def horse(mask):
    horse = mask
    def mask(horse):
        return horse
    return horse(mask)

mask = lambda horse: horse(2)

horse(mask)
```

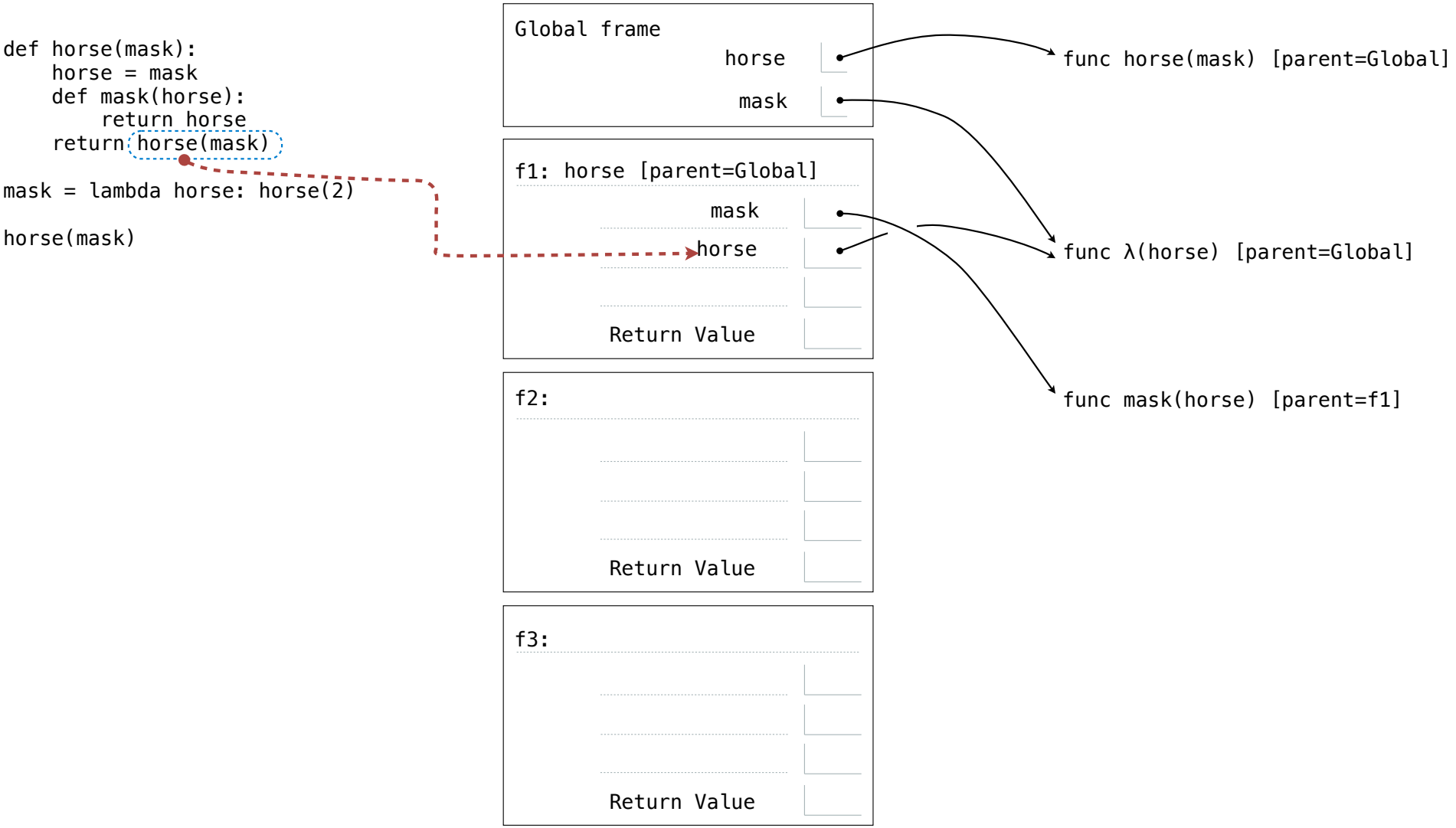


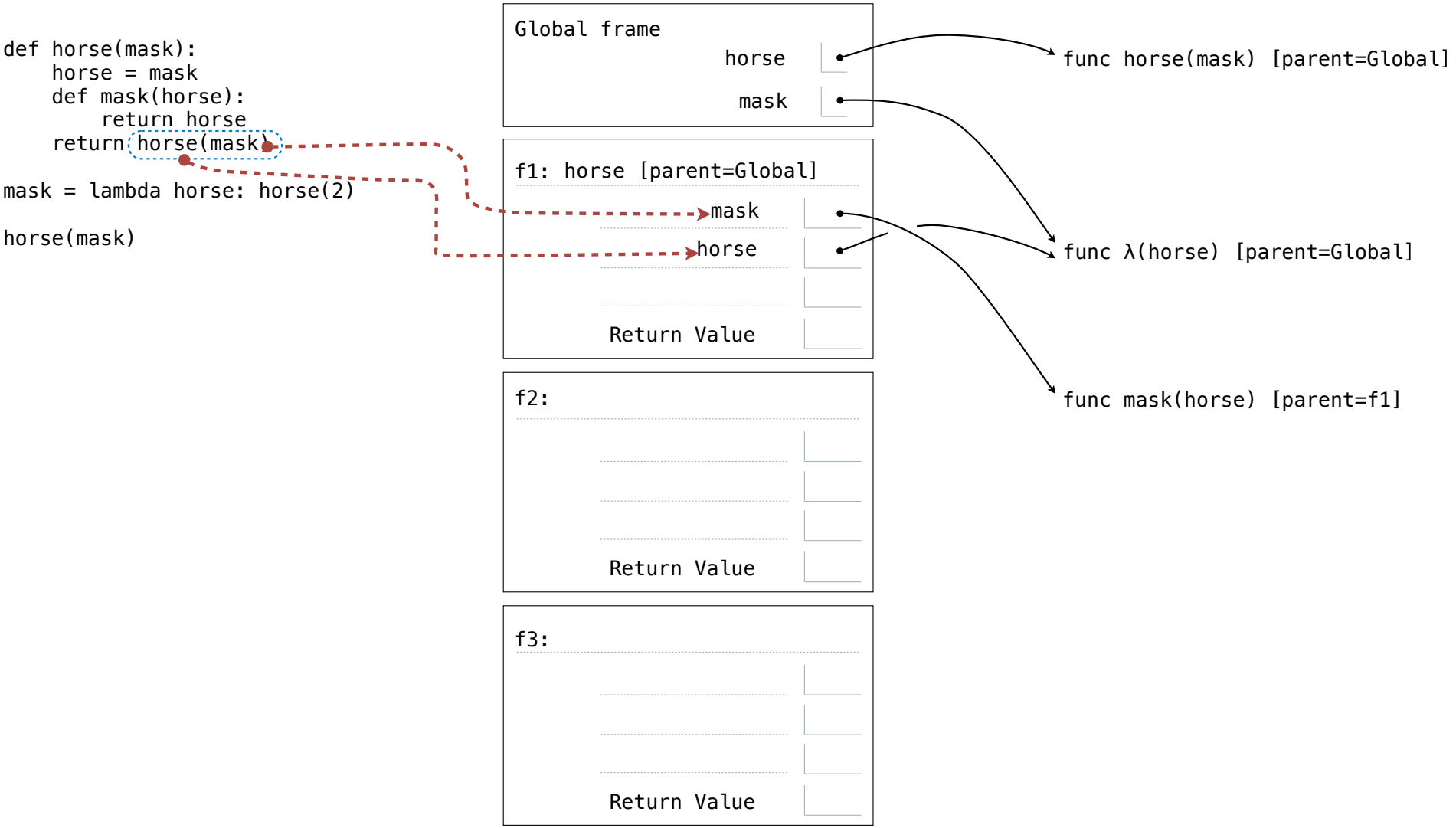
```
def horse(mask):
    horse = mask
    def mask(horse):
        return horse
    return horse(mask)

mask = lambda horse: horse(2)

horse(mask)
```



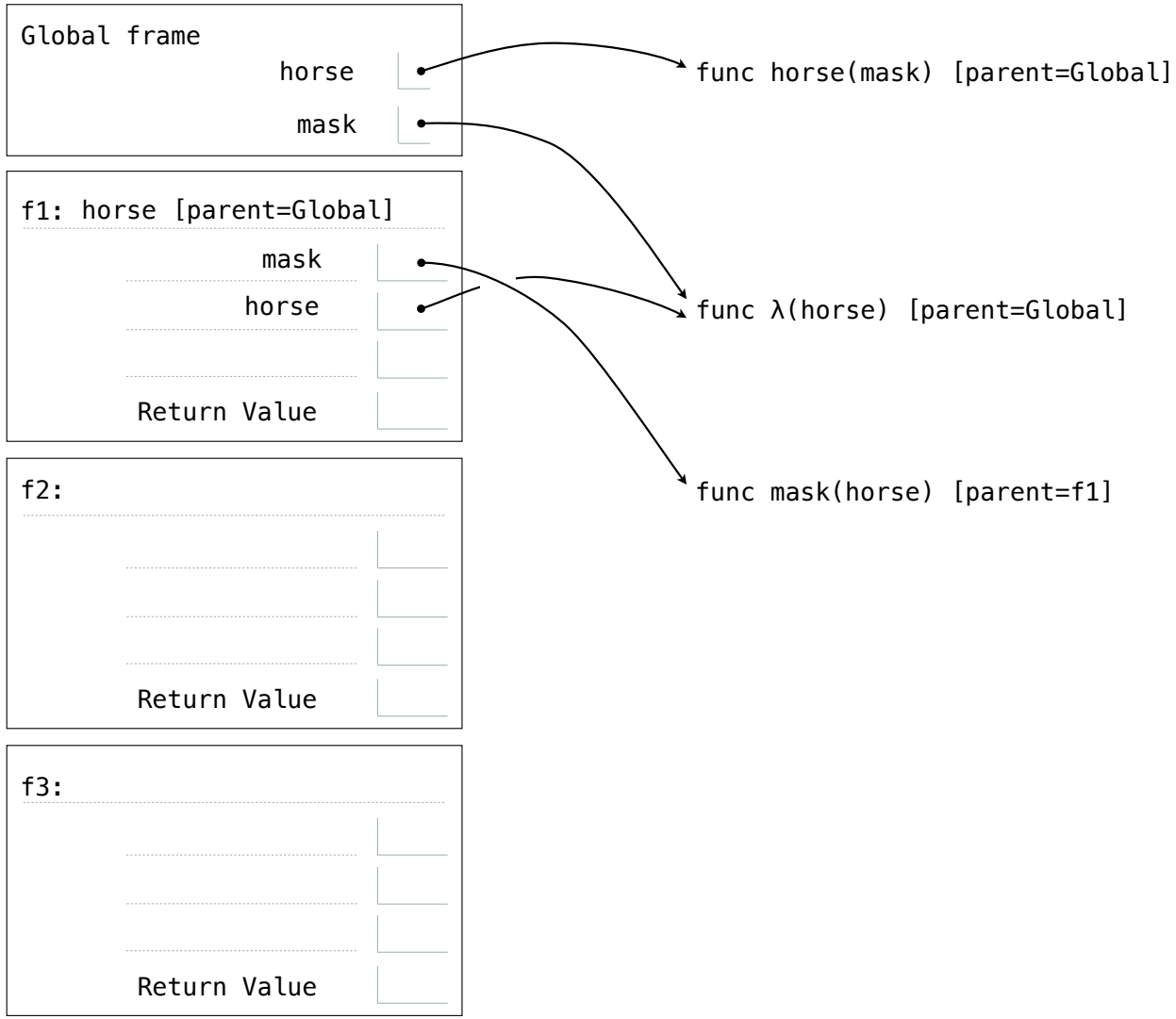
```
def horse(mask):
    horse = mask
    def mask(horse):
        return horse
    return horse(mask)

mask = lambda horse: horse(2)

horse(mask)
```



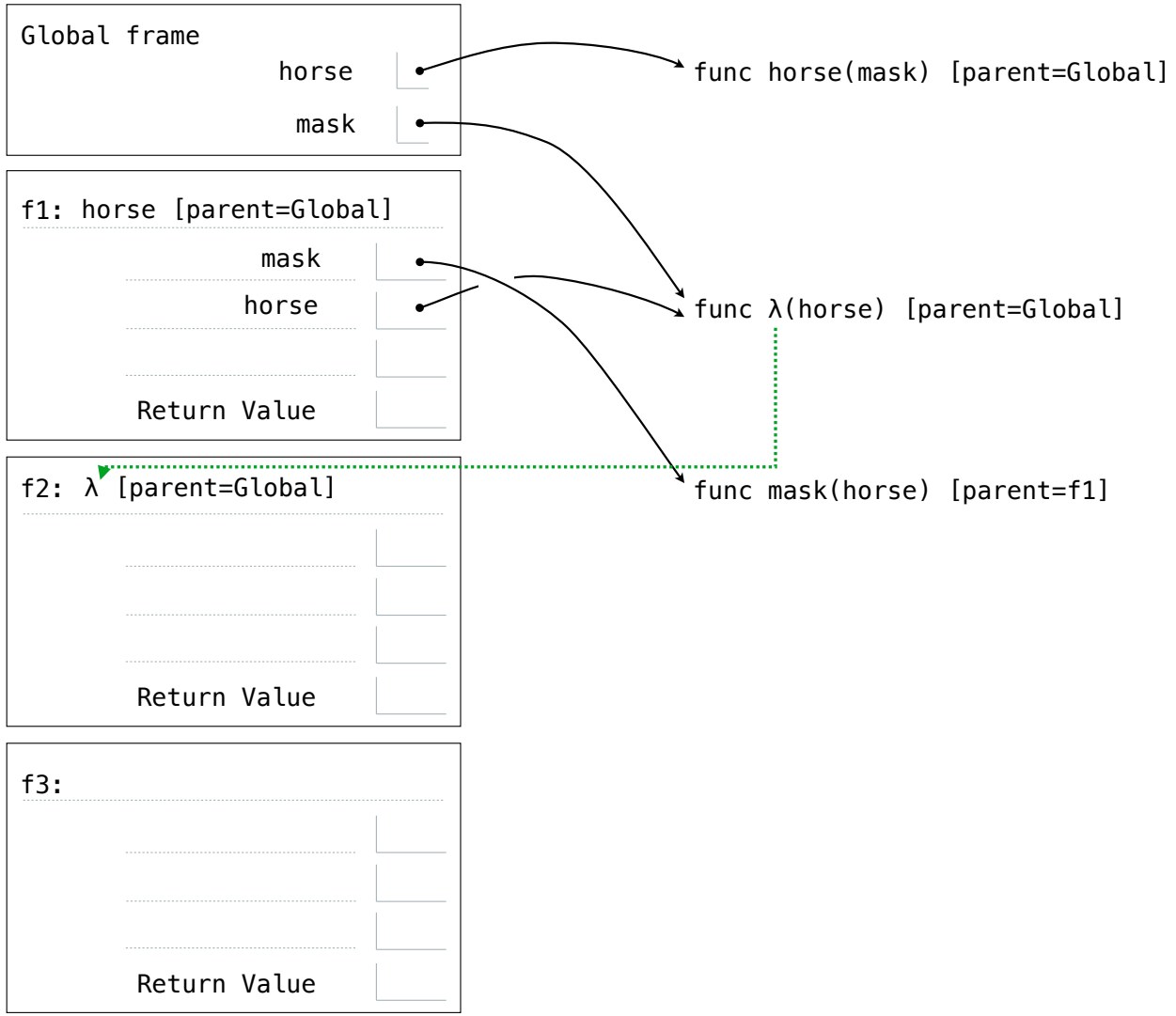

def horse(mask):
horse = mask
def mask(horse):
return horse
return horse(mask)
mask = lambda horse: horse(2)
horse(mask)
Global frame
horse
mask
func horse(mask) [parent=Global]
f1: horse [parent=Global]
mask
horse
Return Value
func λ(horse) [parent=Global]
func mask(horse) [parent=f1]
f2:
Return Value
f3:
Return Value


def horse(mask):
horse = mask
def mask(horse):
return horse
return horse(mask)
mask = lambda horse: horse(2)
horse(mask)
Global frame
horse
mask
func horse(mask) [parent=Global]
f1: horse [parent=Global]
mask
horse
Return Value
func λ(horse) [parent=Global]
func mask(horse) [parent=f1]
f2:
Return Value
f3:
Return Value

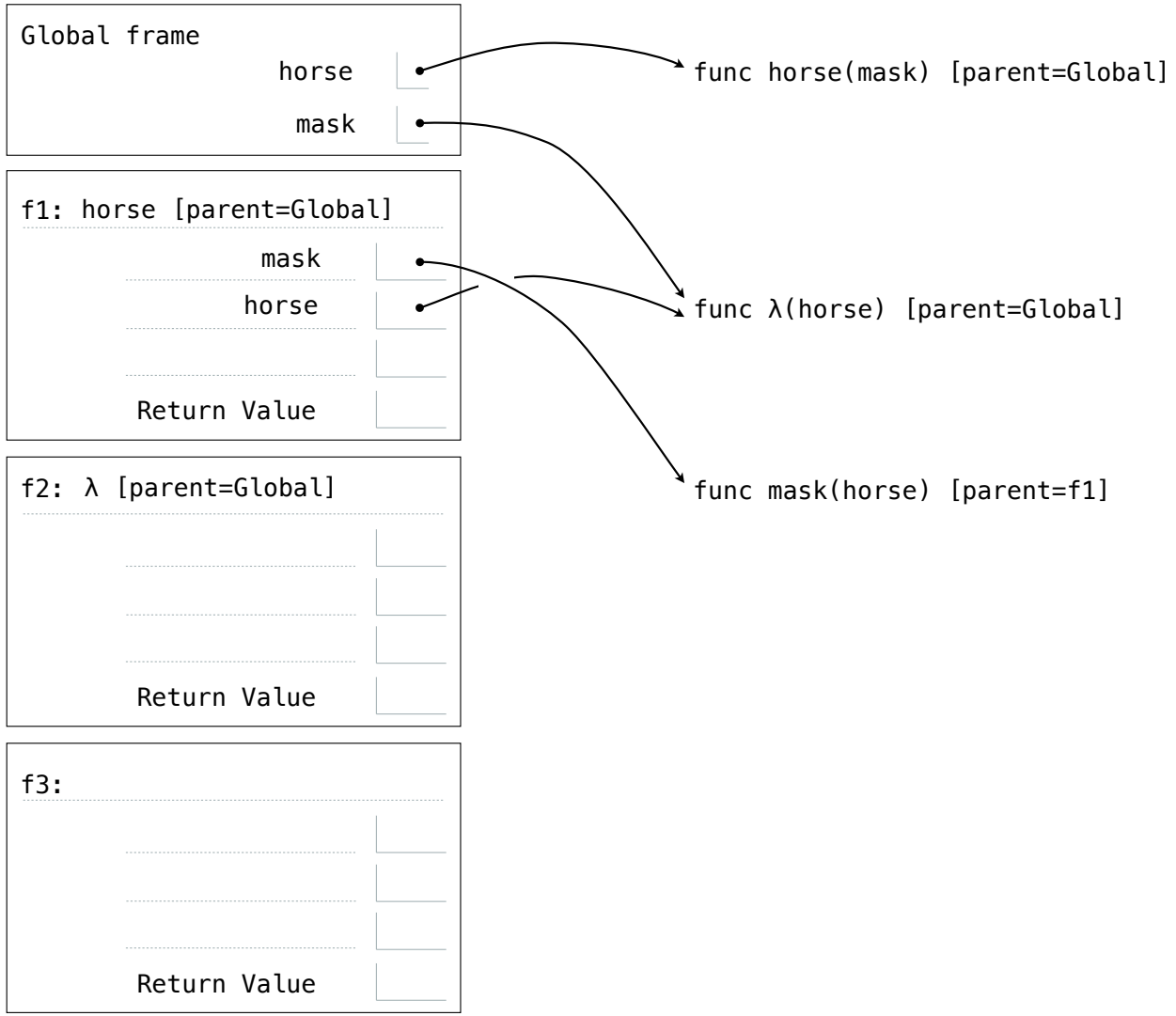
```
def horse(mask):
    horse = mask
    def mask(horse):
        return horse
    return horse(mask)

mask = lambda horse: horse(2)

horse(mask)
```



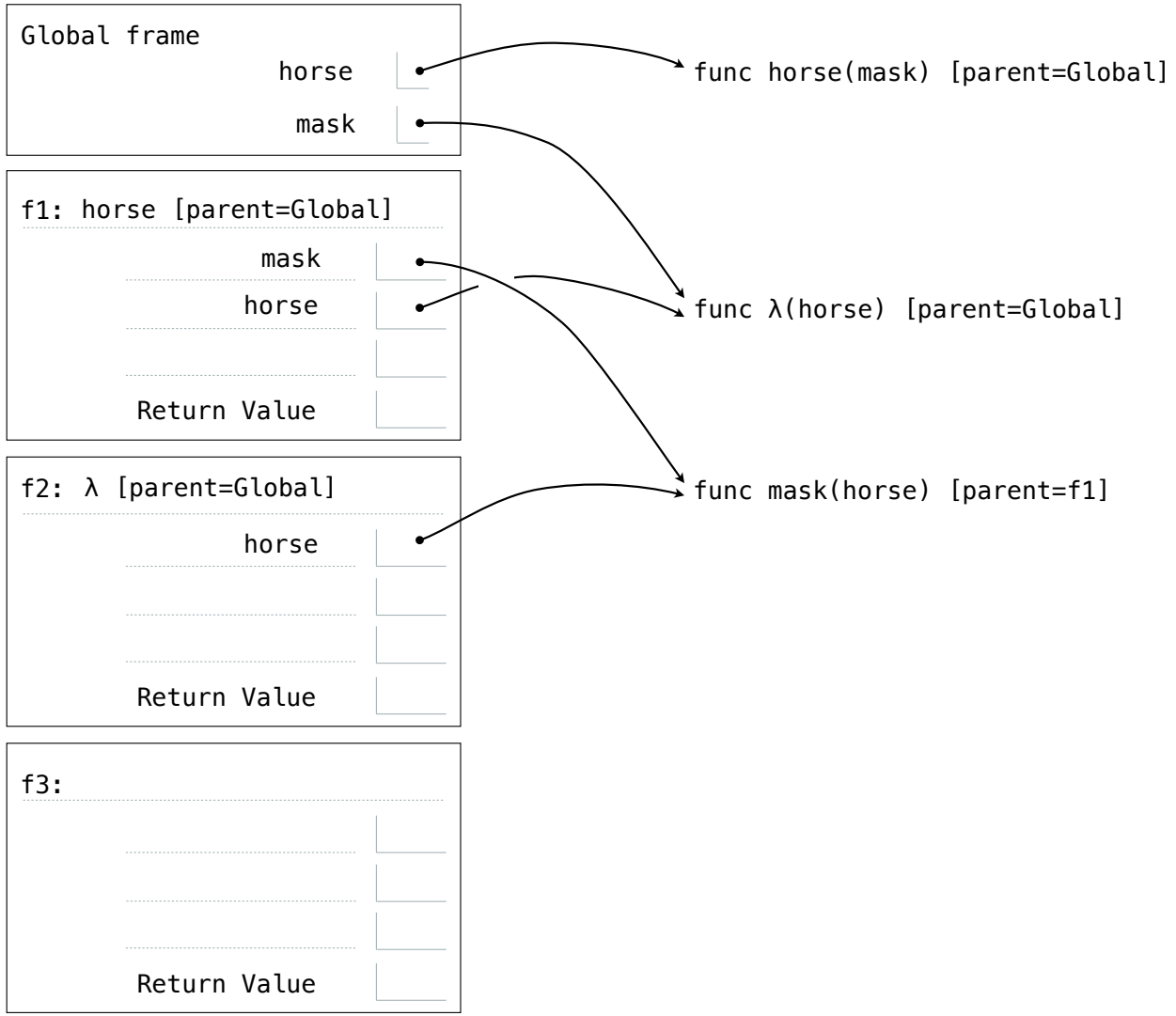
```
def horse(mask):
    horse = mask
    def mask(horse):
        return horse
    return horse(mask)

mask = lambda horse: horse(2)

horse(mask)
```



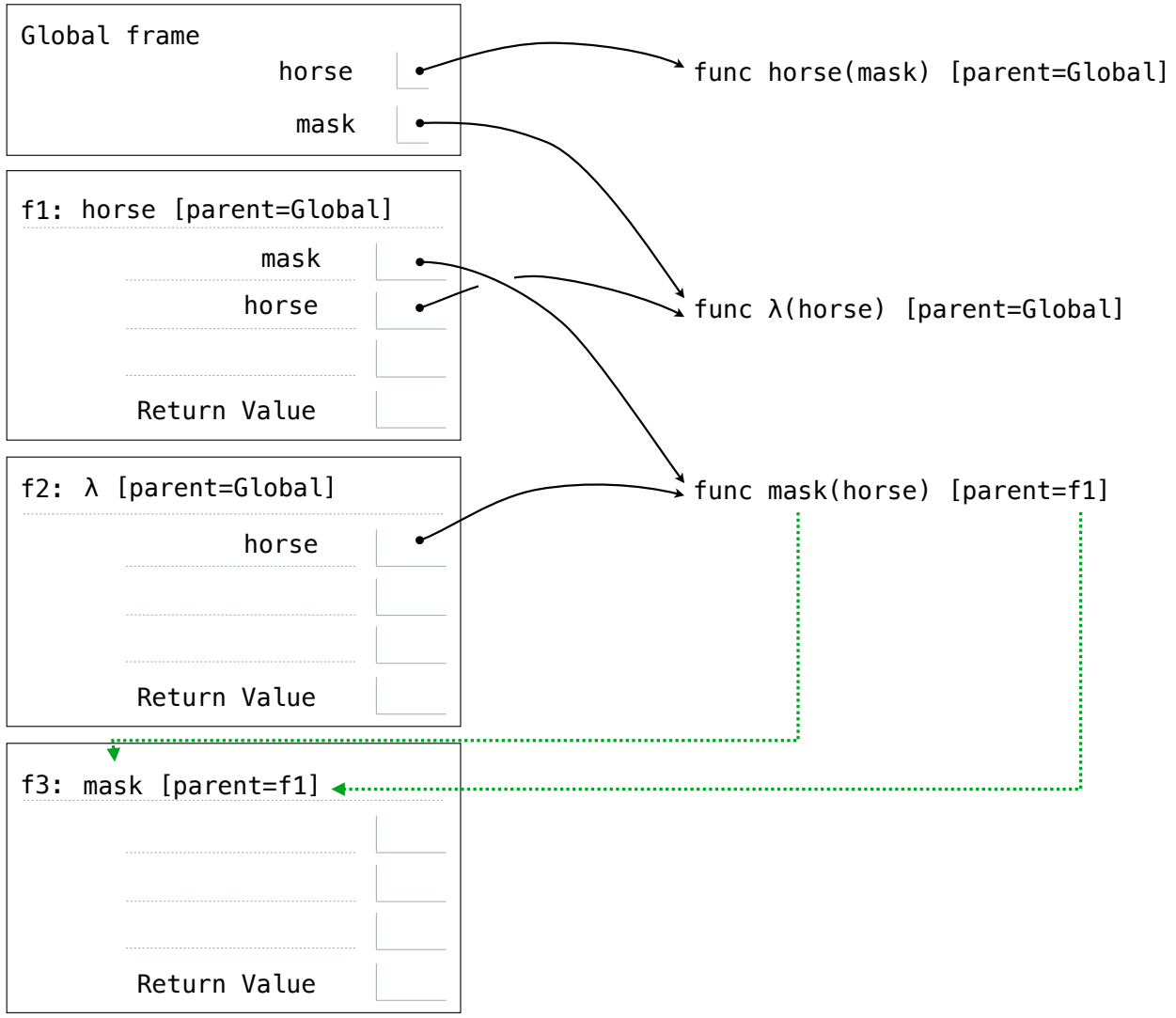
```
def horse(mask):
    horse = mask
    def mask(horse):
        return horse
    return horse(mask)

mask = lambda horse: horse(2)

horse(mask)
```



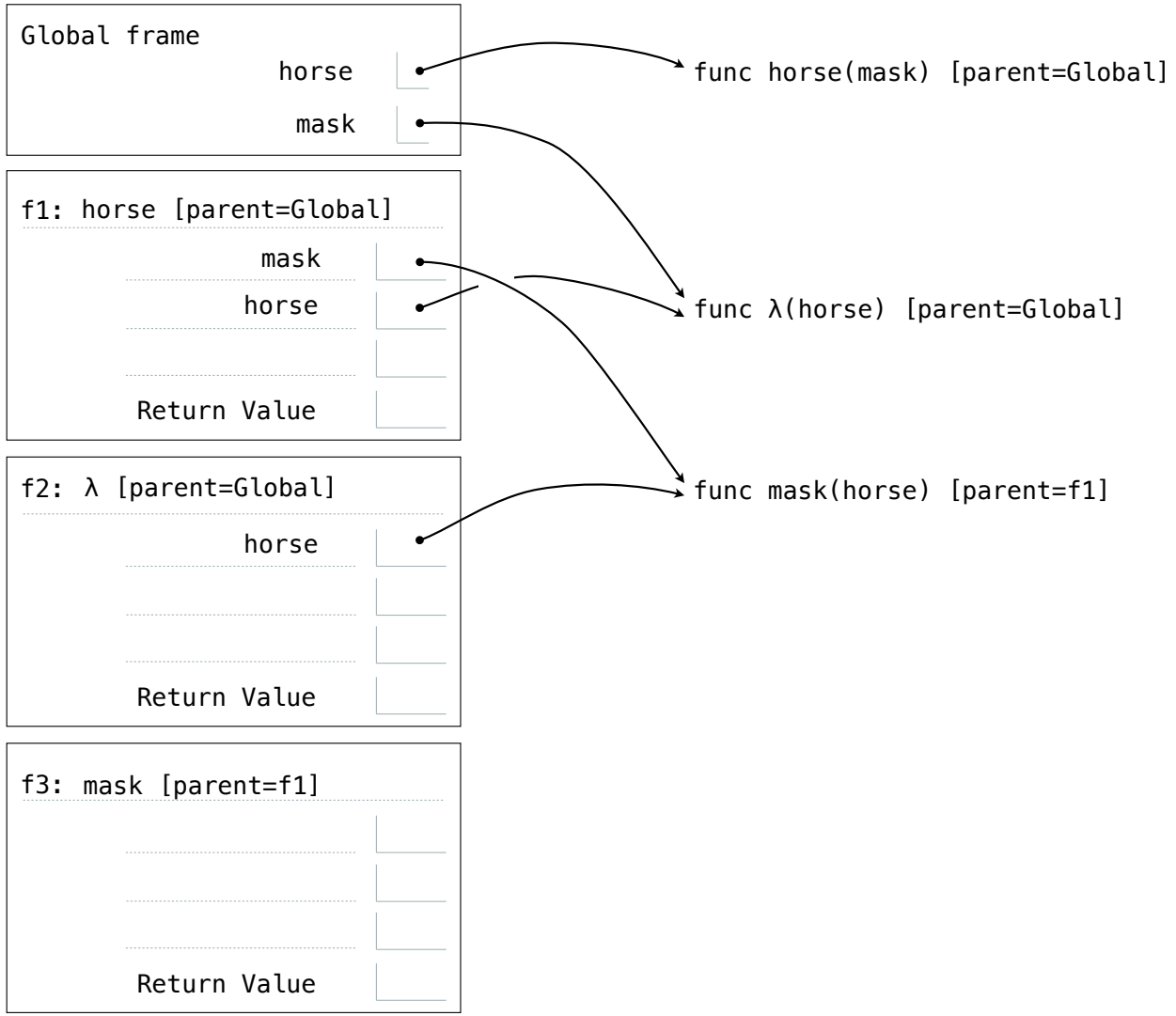
```
def horse(mask):
    horse = mask
    def mask(horse):
        return horse
    return horse(mask)

mask = lambda horse: horse(2)

horse(mask)
```



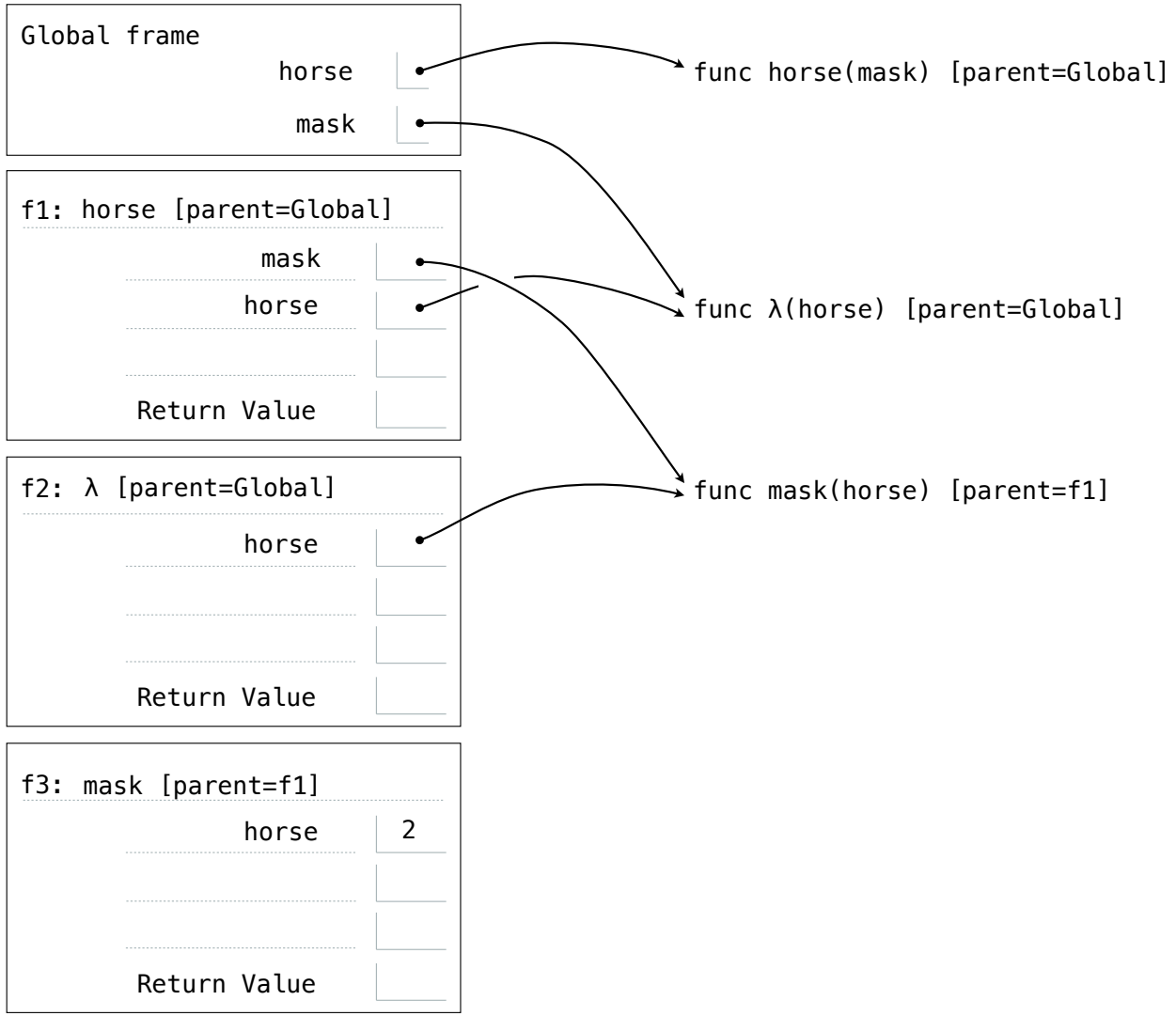
```
def horse(mask):
    horse = mask
    def mask(horse):
        return horse
    return horse(mask)

mask = lambda horse: horse(2)

horse(mask)
```



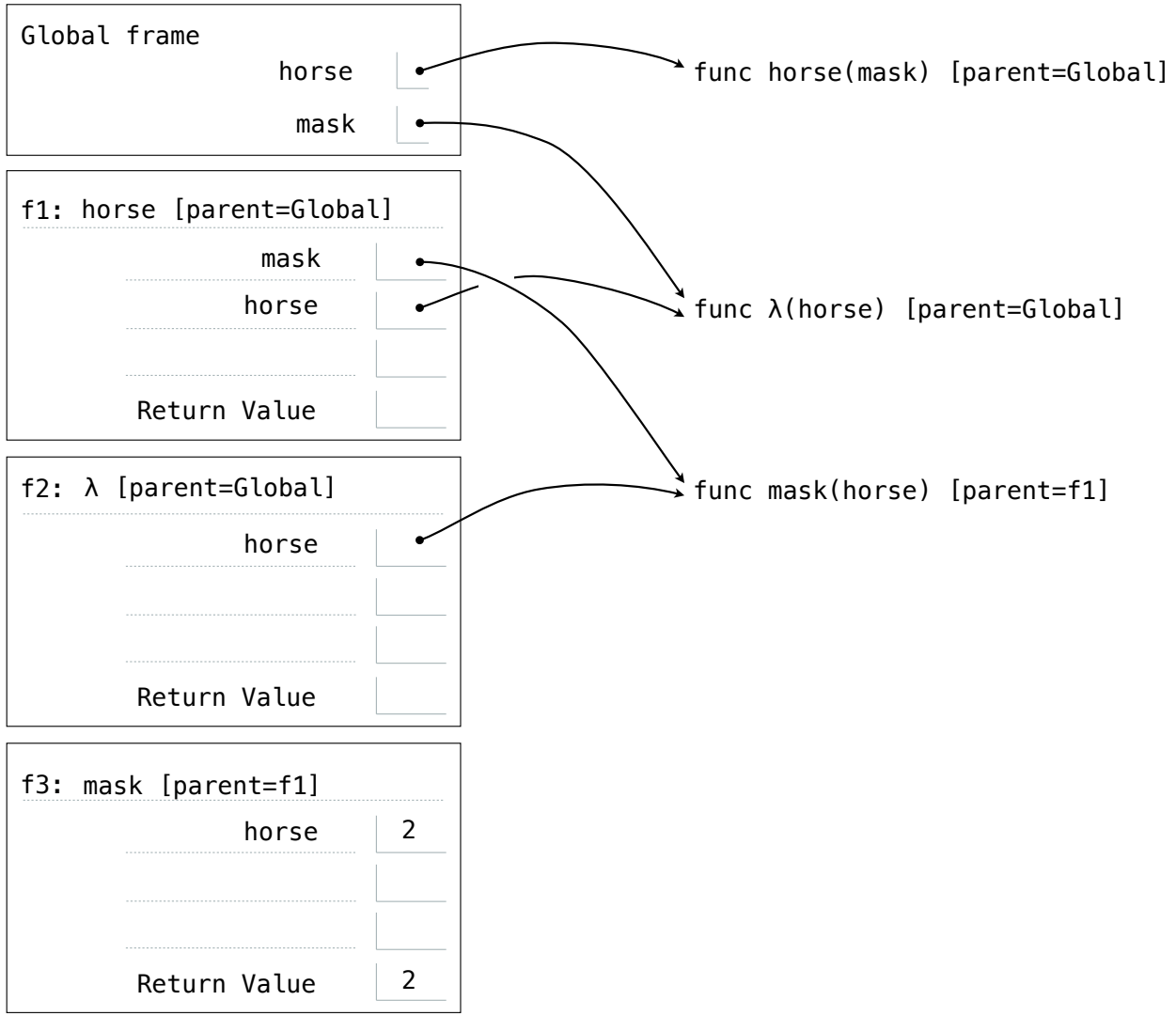
```
def horse(mask):
    horse = mask
    def mask(horse):
        return horse
    return horse(mask)

mask = lambda horse: horse(2)

horse(mask)
```



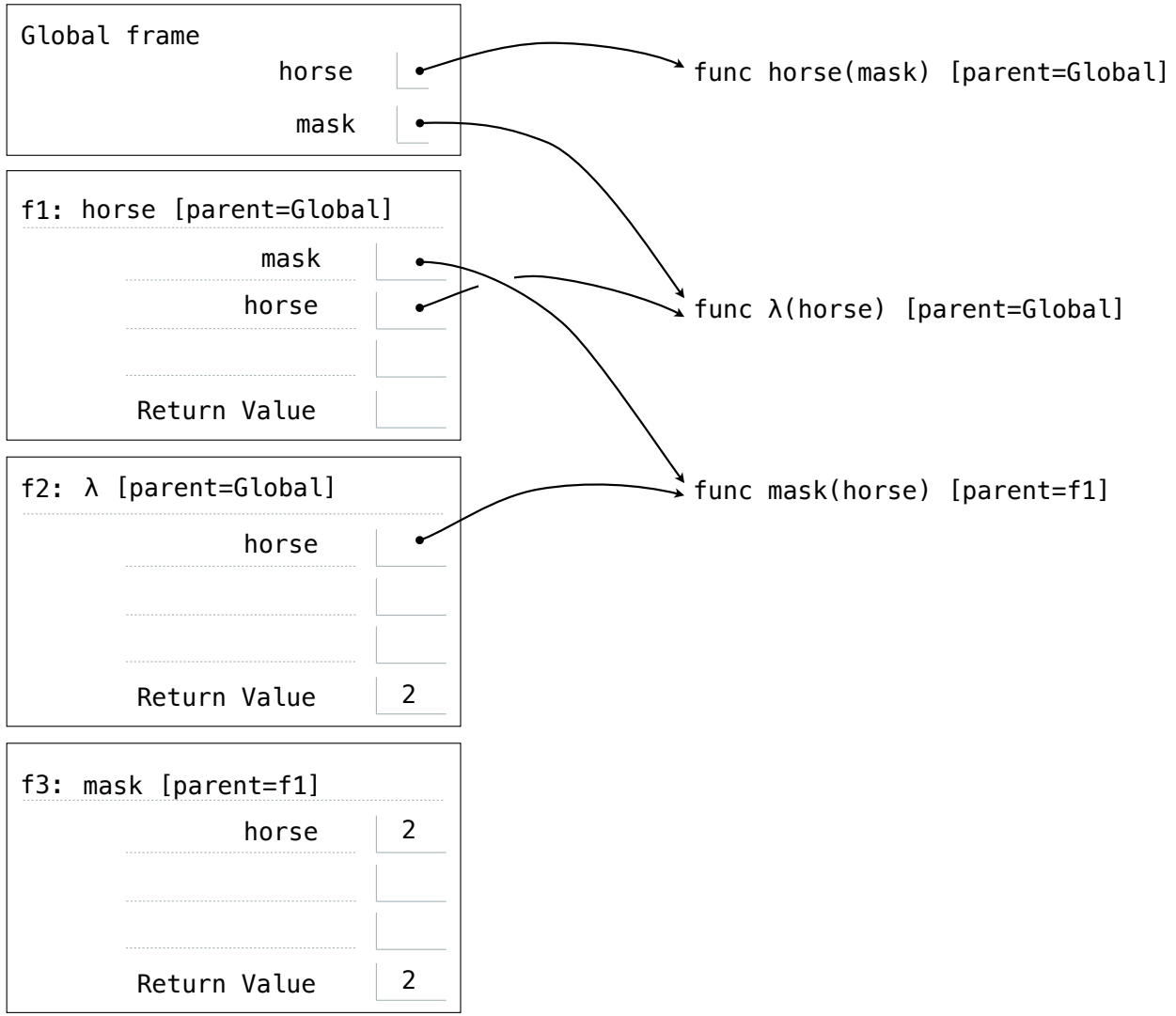
```
def horse(mask):
    horse = mask
    def mask(horse):
        return horse
    return horse(mask)

mask = lambda horse: horse(2)

horse(mask)
```



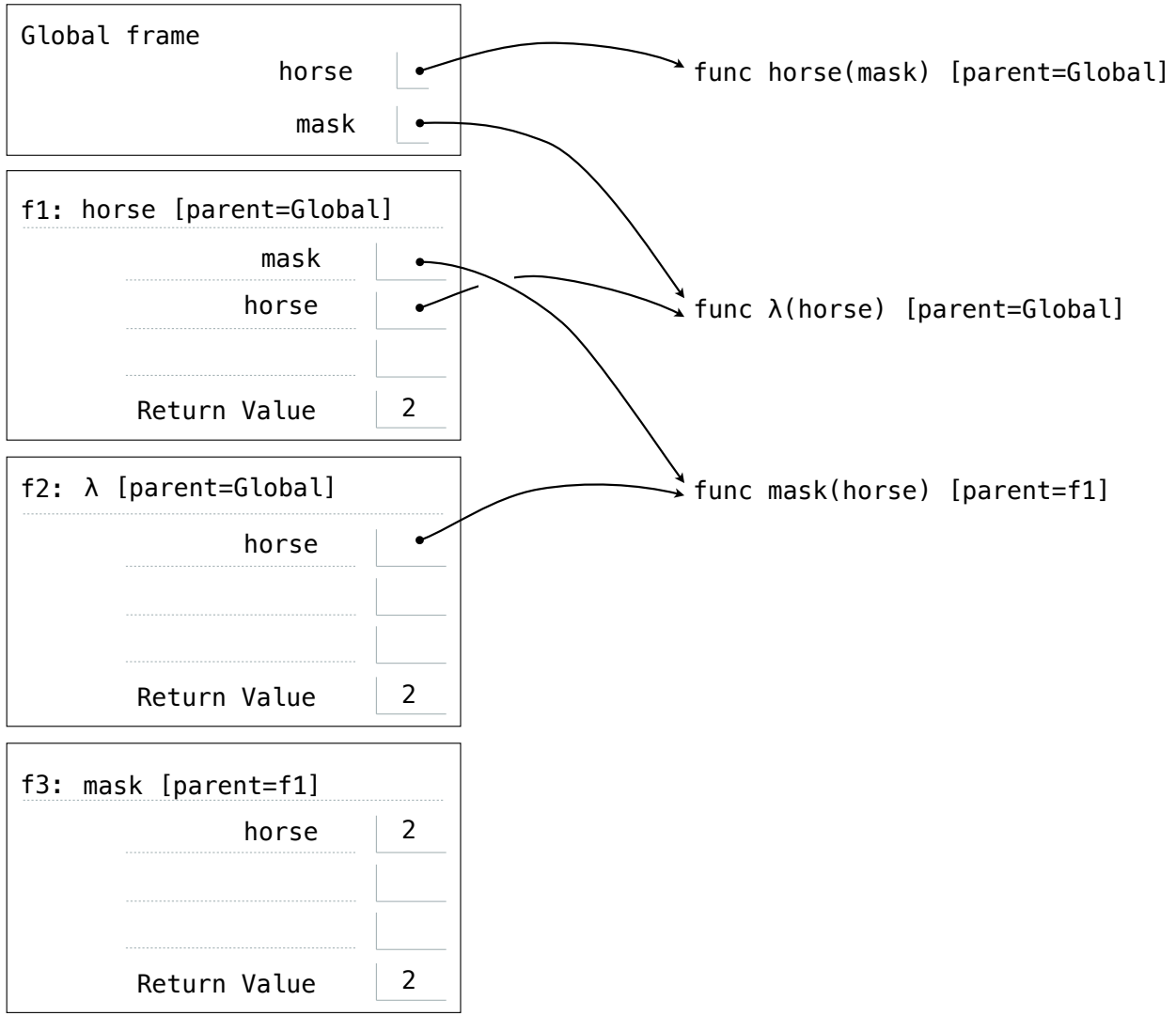
```
def horse(mask):
    horse = mask
    def mask(horse):
        return horse
    return horse(mask)

mask = lambda horse: horse(2)

horse(mask)
```



```
def horse(mask):
    horse = mask
    def mask(horse):
        return horse
    return horse(mask)

mask = lambda horse: horse(2)

horse(mask)
```



```
def horse(mask):
    horse = mask
    def mask(horse):
        return horse
    return horse(mask)

mask = lambda horse: horse(2)

horse(mask)
```



Implementing Functions

Implementing a Function

```
def remove(n, digit):
    """Return all digits of non-negative N
       that are not DIGIT, for some
       non-negative DIGIT less than 10.

    >>> remove(231, 3)
    21
    >>> remove(243132, 2)
    4313
    """
    kept, digits = 0, 0

    while ____________________________________:

        n, last = n // 10, n % 10

        if ____________________________________:

            kept = ____________________________

            digits = ___________________________

    return ____________________________________
```

Implementing a Function

```
def remove(n, digit):
    """Return all digits of non-negative N
       that are not DIGIT, for some
       non-negative DIGIT less than 10.

    >>> remove(231, 3)
    21
    >>> remove(243132, 2)
    4313
    """
    kept, digits = 0, 0

    while ____________________________________:

        n, last = n // 10, n % 10

        if ____________________________________:

            kept = ____________________________

            digits = ___________________________

    return ____________________________________
```

Read the description

Implementing a Function

```
def remove(n, digit):
    """Return all digits of non-negative N
       that are not DIGIT, for some
       non-negative DIGIT less than 10.

    >>> remove(231, 3)
    21
    >>> remove(243132, 2)
    4313
    """
    kept, digits = 0, 0

    while ____________________________________:

        n, last = n // 10, n % 10

        if ____________________________________:

            kept = ____________________________

            digits = ___________________________

    return ____________________________________
```

Read the description

Verify the examples & pick a simple one

Implementing a Function

```
def remove(n, digit):
    """Return all digits of non-negative N
       that are not DIGIT, for some
       non-negative DIGIT less than 10.

    >>> remove(231, 3)
    21
    >>> remove(243132, 2)
    4313
    """
    kept, digits = 0, 0

    while ____________________________________:

        n, last = n // 10, n % 10

        if ____________________________________:

            kept = ____________________________

            digits = ___________________________

    return ____________________________________
```

Read the description

Verify the examples & pick a simple one

Read the template

Implementing a Function

```
def remove(n, digit):
    """Return all digits of non-negative N
       that are not DIGIT, for some
       non-negative DIGIT less than 10.

    >>> remove(231, 3)
    21
    >>> remove(243132, 2)
    4313
    """
    kept, digits = 0, 0

    while ____________________________________:

        n, last = n // 10, n % 10

        if ____________________________________:

            kept = ________________________

            digits = ________________________

    return ____________________________________
```

Read the description

Verify the examples & pick a simple one

Read the template

Implement without the template, then change your implementation to match the template.
OR
If the template is helpful, use it.

Implementing a Function

```
def remove(n, digit):
    """Return all digits of non-negative N
       that are not DIGIT, for some
       non-negative DIGIT less than 10.

    >>> remove(231, 3)
    21
    >>> remove(243132, 2)
    4313
    """
    kept, digits = 0, 0

    while ____________________________________:

        n, last = n // 10, n % 10

        if ____________________________________:

            kept = ____________________________

            digits = ___________________________

    return ____________________________________
```

Read the description

Verify the examples & pick a simple one

Read the template

Implement without the template, then change your implementation to match the template.
OR
If the template is helpful, use it.

Annotate names with values from your chosen example

Implementing a Function

```
def remove(n, digit):
    """Return all digits of non-negative N
       that are not DIGIT, for some
       non-negative DIGIT less than 10.

    >>> remove(231, 3)
    21
    >>> remove(243132, 2)
    4313
    """
    kept, digits = 0, 0

    while ____________________________________:

        n, last = n // 10, n % 10

        if ____________________________________:

            kept = ____________________________

            digits = ___________________________

    return ____________________________________
```

Read the description

Verify the examples & pick a simple one

Read the template

Implement without the template, then change your implementation to match the template.
OR
If the template is helpful, use it.

Annotate names with values from your chosen example

Write code to compute the result

Implementing a Function

```
def remove(n, digit):
    """Return all digits of non-negative N
       that are not DIGIT, for some
       non-negative DIGIT less than 10.

    >>> remove(231, 3)
    21
    >>> remove(243132, 2)
    4313
    """
    kept, digits = 0, 0

    while ____________________________________:

        n, last = n // 10, n % 10

        if ____________________________________:

            kept = ____________________________

            digits = ____________________________

    return ____________________________________
```

Read the description

Verify the examples & pick a simple one

Read the template

Implement without the template, then change your implementation to match the template.
OR
If the template is helpful, use it.

Annotate names with values from your chosen example

Write code to compute the result

Did you really return the right thing?

Implementing a Function

```
def remove(n, digit):
    """Return all digits of non-negative N
       that are not DIGIT, for some
       non-negative DIGIT less than 10.

    >>> remove(231, 3)
    21
    >>> remove(243132, 2)
    4313
    """
    kept, digits = 0, 0

    while ____________________________________:

        n, last = n // 10, n % 10

        if ____________________________________:

            kept = ____________________________

            digits = ___________________________

    return ____________________________________
```

Read the description

Verify the examples & pick a simple one

Read the template

Implement without the template, then change your implementation to match the template.
OR
If the template is helpful, use it.

Annotate names with values from your chosen example

Write code to compute the result

Did you really return the right thing?

Check your solution with the other examples

Implementing a Function

```
def remove(n, digit):
    """Return all digits of non-negative N
    that are not DIGIT, for some
    non-negative DIGIT less than 10.

    >>> remove(231, 3)
    21
    >>> remove(243132, 2)
    4313
    """
    kept, digits = 0, 0

    while ____________________________________:

        n, last = n // 10, n % 10

        if ____________________________________:

            kept = ____________________________

            digits = ___________________________

    return ____________________________________
```

231

3

Read the description

Verify the examples & pick a simple one

Read the template

Implement without the template, then change your implementation to match the template.
OR
If the template is helpful, use it.

Annotate names with values from your chosen example

Write code to compute the result

Did you really return the right thing?

Check your solution with the other examples

Implementing a Function

```
def remove(n, digit):
    """Return all digits of non-negative N
    that are not DIGIT, for some
    non-negative DIGIT less than 10.

    >>> remove(231, 3)
    21
    >>> remove(243132, 2)
    4313
    """
    kept, digits = 0, 0

    while ____________________________________:

        n, last = n // 10, n % 10

        if ____________________________________:

            kept = ____________________________

            digits = ___________________________

    return ____________________________________
```

231

3

21

Read the description

Verify the examples & pick a simple one

Read the template

Implement without the template, then change your implementation to match the template.
OR
If the template is helpful, use it.

Annotate names with values from your chosen example

Write code to compute the result

Did you really return the right thing?

Check your solution with the other examples

Implementing a Function

```
def remove(n, digit):
    """Return all digits of non-negative N
    that are not DIGIT, for some
    non-negative DIGIT less than 10.

    >>> remove(231, 3)
    21
    >>> remove(243132, 2)
    4313
    """
    kept, digits = 0, 0

    while ____________n > 0_____________:

        n, last = n // 10, n % 10

        if ____________________________:

            kept = ____________________

            digits = ___________________

    return ____________________________
```

231

3

21

Read the description

Verify the examples & pick a simple one

Read the template

Implement without the template, then change your implementation to match the template.
OR
If the template is helpful, use it.

Annotate names with values from your chosen example

Write code to compute the result

Did you really return the right thing?

Check your solution with the other examples

Implementing a Function

```
def remove(n, digit):
    """Return all digits of non-negative N
    that are not DIGIT, for some
    non-negative DIGIT less than 10.

    >>> remove(231, 3)
    21
    >>> remove(243132, 2)
    4313
    """
    kept, digits = 0, 0
    while ____________n > 0____________:
        n, last = n // 10, n % 10
        if ____last != digit____:
            kept = ____________________
            digits = ____________________
    return ____________________________
```

231

3

21

Read the description

Verify the examples & pick a simple one

Read the template

Implement without the template, then change your implementation to match the template.
OR
If the template is helpful, use it.

Annotate names with values from your chosen example

Write code to compute the result

Did you really return the right thing?

Check your solution with the other examples

Implementing a Function

```
def remove(n, digit):
    """Return all digits of non-negative N
    that are not DIGIT, for some
    non-negative DIGIT less than 10.

    >>> remove(231, 3)
    21
    >>> remove(243132, 2)
    4313
    """
    kept, digits = 0, 0
    while ____________n > 0____________:
        n, last = n // 10, n % 10
        if ____last != digit____:
            kept = _____________________
            digits = ___________________
    return ____kept____
```

231

3

21

Read the description

Verify the examples & pick a simple one

Read the template

Implement without the template, then change your implementation to match the template.
OR
If the template is helpful, use it.

Annotate names with values from your chosen example

Write code to compute the result

Did you really return the right thing?

Check your solution with the other examples

Implementing a Function

```
def remove(n, digit):
    """Return all digits of non-negative N
    that are not DIGIT, for some
    non-negative DIGIT less than 10.

    >>> remove(231, 3)
    21
    >>> remove(243132, 2)
    4313
    """
    kept, digits = 0, 0
    while ____________n > 0____________:
        n, last = n // 10, n % 10
        if ____________last != digit____________:
            kept = ____________kept + last____________
        digits = ____________________
    return ____________kept____________
```

231

3

21

Read the description

Verify the examples & pick a simple one

Read the template

Implement without the template, then change your implementation to match the template.
OR
If the template is helpful, use it.

Annotate names with values from your chosen example

Write code to compute the result

Did you really return the right thing?

Check your solution with the other examples

Implementing a Function

```
def remove(n, digit):
    """Return all digits of non-negative N
    that are not DIGIT, for some
    non-negative DIGIT less than 10.

    >>> remove(231, 3)
    21
    >>> remove(243132, 2)
    4313
    """
    kept, digits = 0, 0

    while ____________n > 0____________:

        n, last = n // 10, n % 10

        if ____________last != digit____________:

            kept = ____________10*kept + last____________

            digits = ____________________

    return ____________kept____________
```

231

3

21

Read the description

Verify the examples & pick a simple one

Read the template

Implement without the template, then change your implementation to match the template.
OR
If the template is helpful, use it.

Annotate names with values from your chosen example

Write code to compute the result

Did you really return the right thing?

Check your solution with the other examples

Implementing a Function

```
def remove(n, digit):
    """Return all digits of non-negative N
    that are not DIGIT, for some
    non-negative DIGIT less than 10.

    >>> remove(231, 3)
    21
    >>> remove(243132, 2)
    4313
    """
    kept, digits = 0, 0

    while ____________n > 0____________:

        n, last = n // 10, n % 10

        if ____________last != digit____________:

            kept = ~~10 *~~ kept + last ____________

        digits = ____________

    return ____________kept____________
```

231

3

21

Read the description

Verify the examples & pick a simple one

Read the template

Implement without the template, then change your implementation to match the template.
OR
If the template is helpful, use it.

Annotate names with values from your chosen example

Write code to compute the result

Did you really return the right thing?

Check your solution with the other examples

Implementing a Function

```
def remove(n, digit):
    """Return all digits of non-negative N
    that are not DIGIT, for some
    non-negative DIGIT less than 10.

    >>> remove(231, 3)
    21
    >>> remove(243132, 2)
    4313
    """
    kept, digits = 0, 0

    while ____________n > 0____________:

        n, last = n // 10, n % 10

        if ____________last != digit____________:

            kept = ~~10*~~kept + last*10

        digits = ____________________

    return ____________kept____________
```

231

3

21

Read the description

Verify the examples & pick a simple one

Read the template

Implement without the template, then change your implementation to match the template.
OR
If the template is helpful, use it.

Annotate names with values from your chosen example

Write code to compute the result

Did you really return the right thing?

Check your solution with the other examples

Implementing a Function

```
def remove(n, digit):
    """Return all digits of non-negative N
    that are not DIGIT, for some
    non-negative DIGIT less than 10.

    >>> remove(231, 3)
    21
    >>> remove(243132, 2)
    4313
    """
    kept, digits = 0, 0
    while ____________________________:
        n, last = n // 10, n % 10
        if ____________________________:
            kept = ____________________
        digits = __________________
    return ____________________________
```

231

3

1

+ 20

21

n > 0

last != digit

~~10*~~kept + last*10

21

kept

Read the description

Verify the examples & pick a simple one

Read the template

Implement without the template, then change your implementation to match the template.
OR
If the template is helpful, use it.

Annotate names with values from your chosen example

Write code to compute the result

Did you really return the right thing?

Check your solution with the other examples

Implementing a Function

```
def remove(n, digit):
    """Return all digits of non-negative N
    that are not DIGIT, for some
    non-negative DIGIT less than 10.

    >>> remove(231, 3)
    21
    >>> remove(243132, 2)
    4313
    """
    kept, digits = 0, 0
    while ____________n > 0____________:
        n, last = n // 10, n % 10
        if ____________last != digit____________:
            kept = ____________10*kept + last*10____________
            digits = ____________digits + 1____________
    return ____________kept____________
```

231

3

21

```
  1
+ 20
----
 21
```

Read the description

Verify the examples & pick a simple one

Read the template

Implement without the template, then change your implementation to match the template.
OR
If the template is helpful, use it.

Annotate names with values from your chosen example

Write code to compute the result

Did you really return the right thing?

Check your solution with the other examples

Implementing a Function

```
def remove(n, digit):
    """Return all digits of non-negative N
    that are not DIGIT, for some
    non-negative DIGIT less than 10.

    >>> remove(231, 3)
    21
    >>> remove(243132, 2)
    4313
    """
    kept, digits = 0, 0

    while ____________________________:

        n, last = n // 10, n % 10

        if ____________________________:

            kept = ____________________________

            digits = ____________________________

    return ____________________________
```

231

3

1

+ 20

21

n > 0

last != digit

~~10*~~kept + last*10**digits

digits + 1

21

kept

Read the description

Verify the examples & pick a simple one

Read the template

Implement without the template, then change your implementation to match the template.
OR
If the template is helpful, use it.

Annotate names with values from your chosen example

Write code to compute the result

Did you really return the right thing?

Check your solution with the other examples

Implementing a Function

```
def remove(n, digit):
    """Return all digits of non-negative N
    that are not DIGIT, for some
    non-negative DIGIT less than 10.

    >>> remove(231, 3)
    21
    >>> remove(243132, 2)
    4313
    """
    kept, digits = 0, 0

    while ____________________________:

        n, last = n // 10, n % 10

        if ____________________________:

            kept = ____________________________

            digits = ____________________________

    return ____________________________
```

231

4

21

1

+ 20

21

n > 0

last != digit

~~10*~~kept + last*10**digits

digits + 1

kept

Read the description

Verify the examples & pick a simple one

Read the template

Implement without the template, then change your implementation to match the template.
OR
If the template is helpful, use it.

Annotate names with values from your chosen example

Write code to compute the result

Did you really return the right thing?

Check your solution with the other examples

Implementing a Function

```
def remove(n, digit):
    """Return all digits of non-negative N
    that are not DIGIT, for some
    non-negative DIGIT less than 10.

    >>> remove(231, 3)
    21
    >>> remove(243132, 2)
    4313
    """
    kept, digits = 0, 0

    while _____:
        n, last = n // 10, n % 10

        if _____:
            kept = _____

            digits = _____

    return _____
```

231

4

1

+ 20

21

n > 0

last != digit

~~10*~~kept + last*10**digits

231

digits + 1

kept

Read the description

Verify the examples & pick a simple one

Read the template

Implement without the template, then change your implementation to match the template.
OR
If the template is helpful, use it.

Annotate names with values from your chosen example

Write code to compute the result

Did you really return the right thing?

Check your solution with the other examples

Implementing a Function

```
def remove(n, digit):
    """Return all digits of non-negative N
    that are not DIGIT, for some
    non-negative DIGIT less than 10.

    >>> remove(231, 3)
    21
    >>> remove(243132, 2)
    4313
    """
    kept, digits = 0, 0

    while ____________n > 0____________:

        n, last = n // 10, n % 10

        if ____________last != digit____________:

            kept = ~~10*~~kept + last*10**digits

            digits = ____digits + 1____

    return ____________kept____________
```

231

4

1	1
+ 20	+ 30
	+ 200
21	231

231

Read the description

Verify the examples & pick a simple one

Read the template

Implement without the template, then change your implementation to match the template.
OR
If the template is helpful, use it.

Annotate names with values from your chosen example

Write code to compute the result

Did you really return the right thing?

Check your solution with the other examples

Implementing a Function

```
def remove(n, digit):
    """Return all digits of non-negative N
    that are not DIGIT, for some
    non-negative DIGIT less than 10.

    >>> remove(231, 3)
    21
    >>> remove(243132, 2)
    4313
    """
    kept, digits = 0, 0
    while ____________n > 0____________:
        n, last = n // 10, n % 10
        if ____________last != digit____________:
            kept = ____kept____ + ____last____
        digits = ____________________
    return ____________kept____________
```

231

3

21

Read the description

Verify the examples & pick a simple one

Read the template

Implement without the template, then change your implementation to match the template.
OR
If the template is helpful, use it.

Annotate names with values from your chosen example

Write code to compute the result

Did you really return the right thing?

Check your solution with the other examples

Implementing a Function

```
def remove(n, digit):
    """Return all digits of non-negative N
    that are not DIGIT, for some
    non-negative DIGIT less than 10.

    >>> remove(231, 3)
    21
    >>> remove(243132, 2)
    4313
    """
    kept, digits = 0, 0
    while ____________n > 0____________:
        n, last = n // 10, n % 10
        if ____last != digit____:
            kept = ____kept/10 + last____
            digits = ____________________
    return ____kept____
```

231

3

21

Read the description

Verify the examples & pick a simple one

Read the template

Implement without the template, then change your implementation to match the template.
OR
If the template is helpful, use it.

Annotate names with values from your chosen example

Write code to compute the result

Did you really return the right thing?

Check your solution with the other examples

Implementing a Function

```
def remove(n, digit):
    """Return all digits of non-negative N
    that are not DIGIT, for some
    non-negative DIGIT less than 10.

    >>> remove(231, 3)
    21
    >>> remove(243132, 2)
    4313
    """
    kept, digits = 0, 0
    while ____________n > 0____________:
        n, last = n // 10, n % 10
        if ____last != digit____:
            kept = ____kept/10 + last____
            digits = ____________________
    return ____kept * 10____
```

231

3

21

Read the description

Verify the examples & pick a simple one

Read the template

Implement without the template, then change your implementation to match the template.
OR
If the template is helpful, use it.

Annotate names with values from your chosen example

Write code to compute the result

Did you really return the right thing?

Check your solution with the other examples

Implementing a Function

```
def remove(n, digit):
    """Return all digits of non-negative N
    that are not DIGIT, for some
    non-negative DIGIT less than 10.

    >>> remove(231, 3)
    21
    >>> remove(243132, 2)
    4313
    """
    kept, digits = 0, 0
    while ____________n > 0____________:
        n, last = n // 10, n % 10
        if ____________last != digit____________:
            kept = ____________kept/10 +   last____________
            digits = ____________digits + 1____________
    return ____________kept * 10____________
```

231

3

21

Read the description

Verify the examples & pick a simple one

Read the template

Implement without the template, then change your implementation to match the template.
OR
If the template is helpful, use it.

Annotate names with values from your chosen example

Write code to compute the result

Did you really return the right thing?

Check your solution with the other examples

Implementing a Function

```
def remove(n, digit):
    """Return all digits of non-negative N
    that are not DIGIT, for some
    non-negative DIGIT less than 10.

    >>> remove(231, 3)
    21
    >>> remove(243132, 2)
    4313
    """
    kept, digits = 0, 0
    while ____________n > 0____________:
        n, last = n // 10, n % 10
        if ____last != digit____:
            kept = ____kept/10 +  last____
            digits = ____digits + 1____
    return ____kept * 10 ** (digits-1)____
```

231

3

21

Read the description

Verify the examples & pick a simple one

Read the template

Implement without the template, then change your implementation to match the template.
OR
If the template is helpful, use it.

Annotate names with values from your chosen example

Write code to compute the result

Did you really return the right thing?

Check your solution with the other examples

Implementing a Function

```
def remove(n, digit):
    """Return all digits of non-negative N
    that are not DIGIT, for some
    non-negative DIGIT less than 10.

    >>> remove(231, 3)
    21
    >>> remove(243132, 2)
    4313
    """
    kept, digits = 0, 0
    while ____________n > 0____________:
        n, last = n // 10, n % 10
        if ____________last != digit____________:
            kept = ____kept/10 +   last____
            digits = ____digits + 1____
    return ____round(kept * 10 ** (digits-1))____
```

231

3

21

Read the description

Verify the examples & pick a simple one

Read the template

Implement without the template, then change your implementation to match the template.
OR
If the template is helpful, use it.

Annotate names with values from your chosen example

Write code to compute the result

Did you really return the right thing?

Check your solution with the other examples

Decorators

Function Decorators

(Demo)

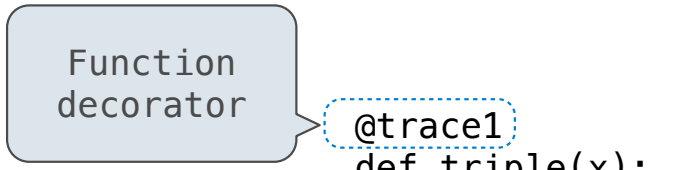
Function Decorators

(Demo)

```
@trace1
def triple(x):
    return 3 * x
```

Function Decorators

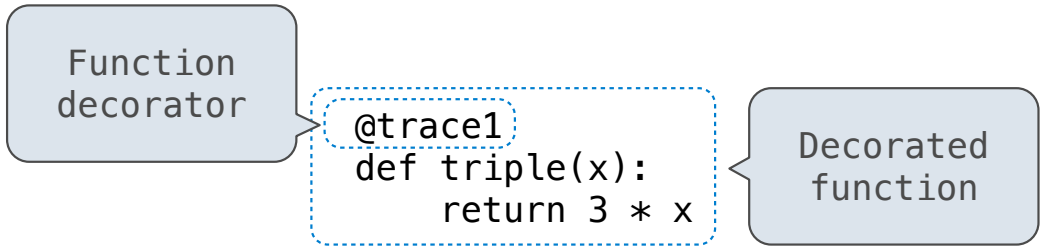
(Demo)




```
@trace1
def triple(x):
    return 3 * x
```

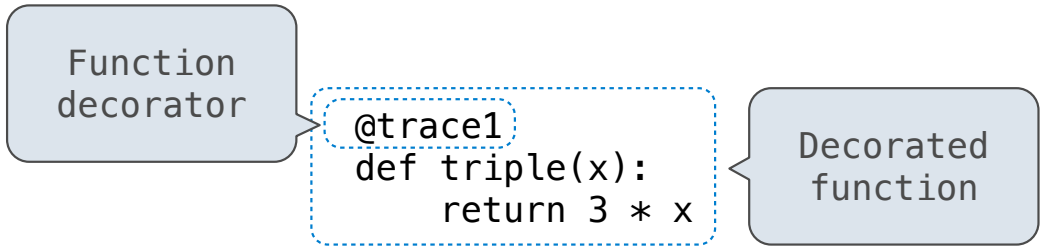
Function Decorators

(Demo)



Function Decorators

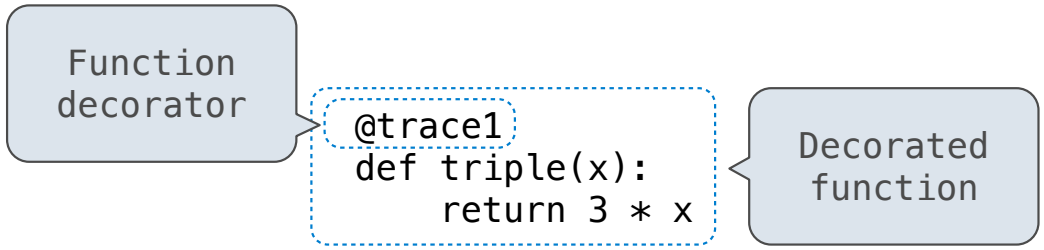
(Demo)



is identical to

Function Decorators

(Demo)



is identical to

```
def triple(x):
    return 3 * x
triple = trace1(triple)
```

Function Decorators

(Demo)

Function decorator

```
@trace1
def triple(x):
    return 3 * x
```

Decorated function

is identical to

Why not just use this?

```
def triple(x):
    return 3 * x
triple = trace1(triple)
```