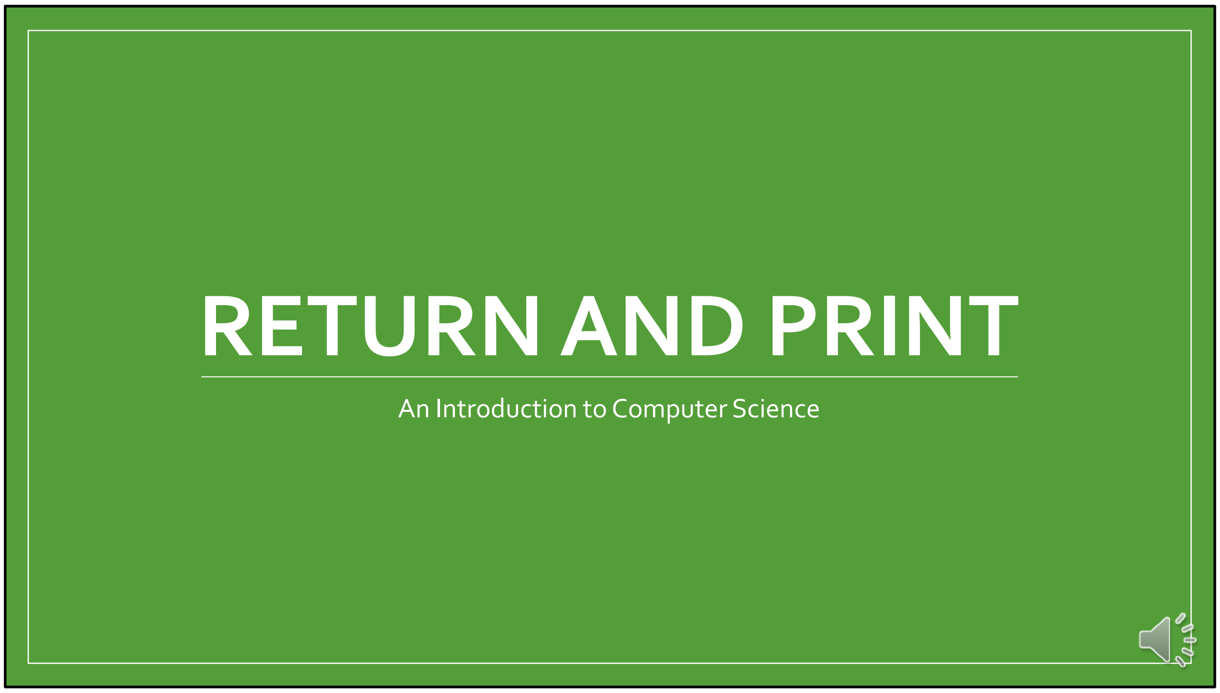
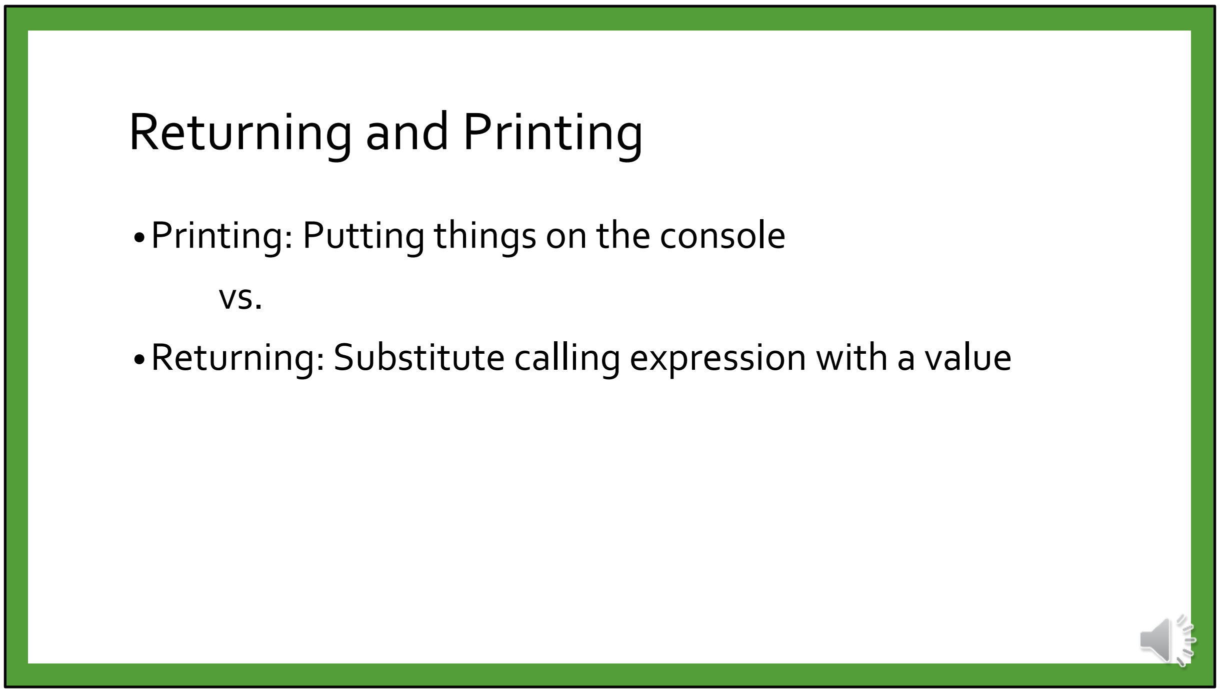


# RETURN AND PRINT

An Introduction to Computer Science

Let's learn about Returning and Printing.



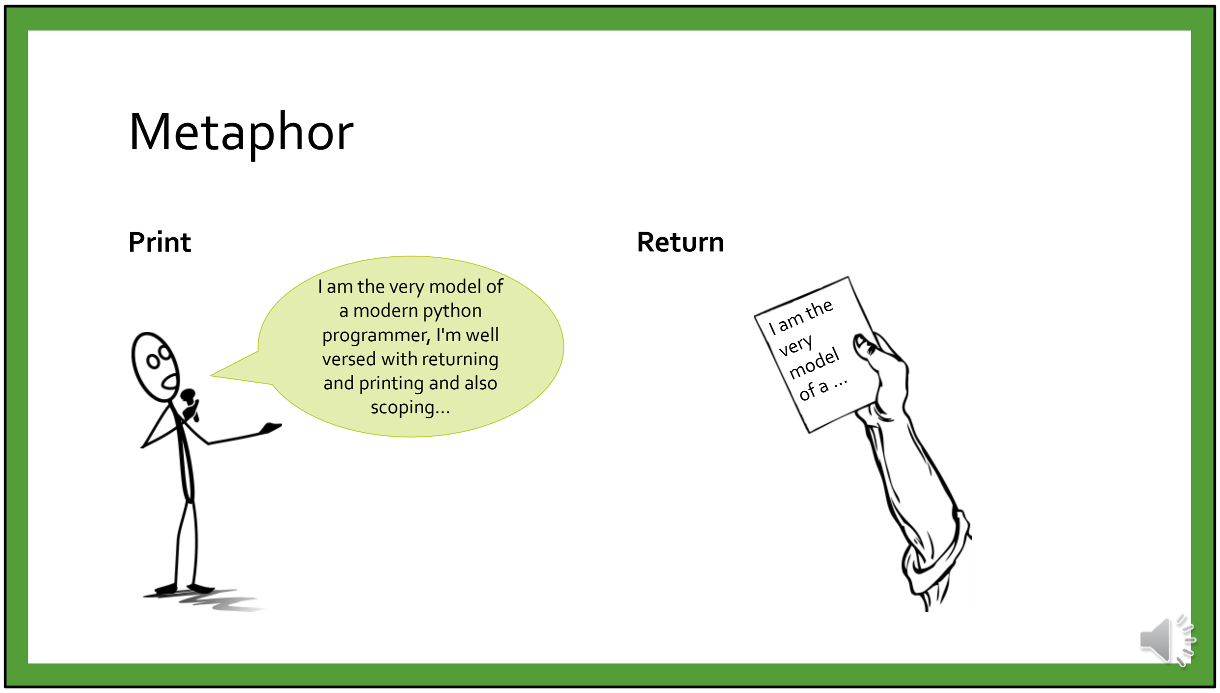
## Returning and Printing

- Printing: Putting things on the console

  vs.

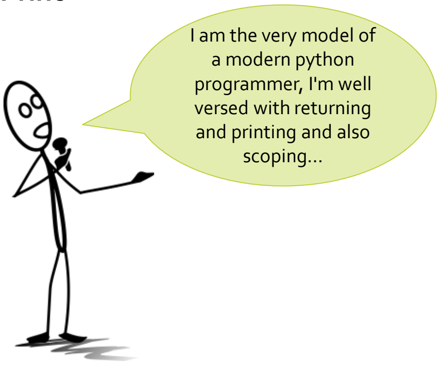
- Returning: Substitute calling expression with a value

Printing and returning are two related but different ideas.
However, many beginners confuse them.

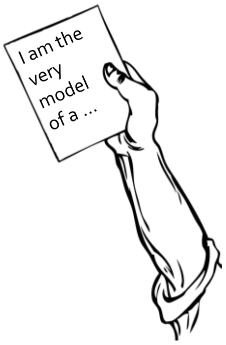


## Metaphor

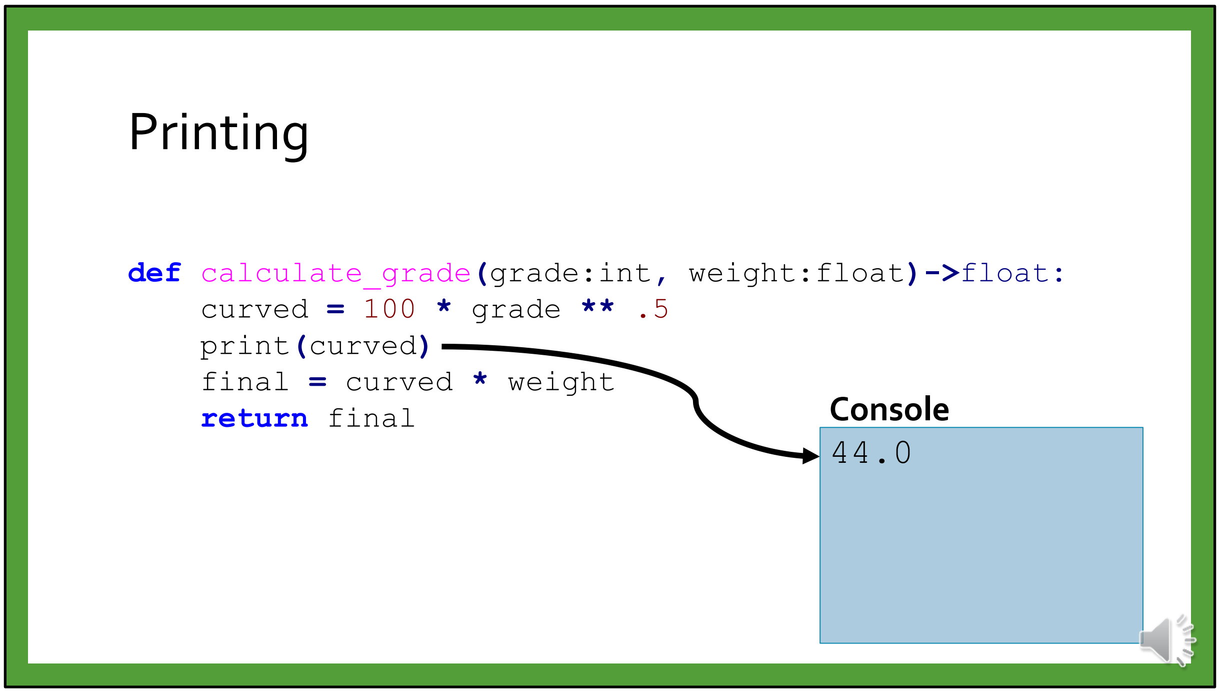
**Print**


I am the very model of a modern python programmer, I'm well versed with returning and printing and also scoping…

**Return**



Consider these two different scenarios.
Say I ask you to recite a poem out loud.
One you are done, the poetry is no longer available to me.
I heard it, and then it was gone.
This is like printing.
Now, say I asked you to write that poem down and hand it to me.
The poem on paper is a concrete thing that I could hand off to someone else, make edits to, or even destroy.
This is like returning a value.



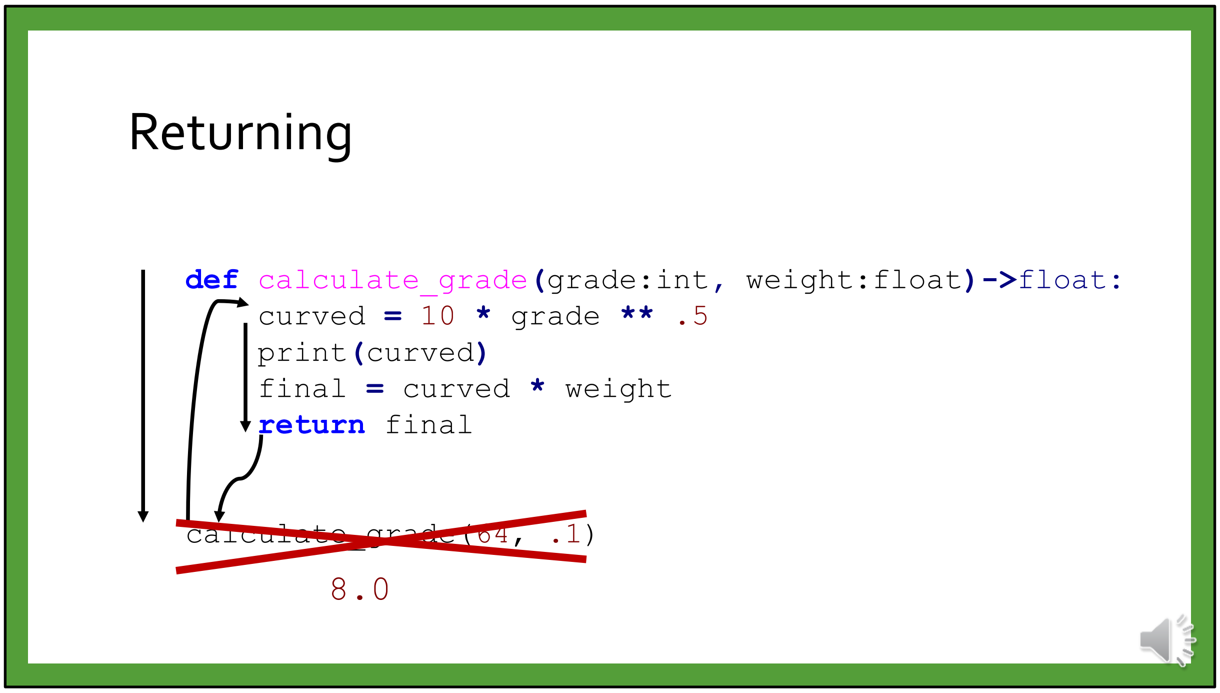
## Printing

```
def calculate_grade(grade:int, weight:float)->float:
    curved = 100 * grade ** .5
    print(curved)
    final = curved * weight
    return final
```

Console

```
44.0
```

Variables and data stored inside of a computer are invisible to us.
In order to see the results of computation, we often print variables and values.
Calling the built-in print function will cause text to appear on the console.
Printing data to the console is a one-way trip, and we cannot get data written there back.



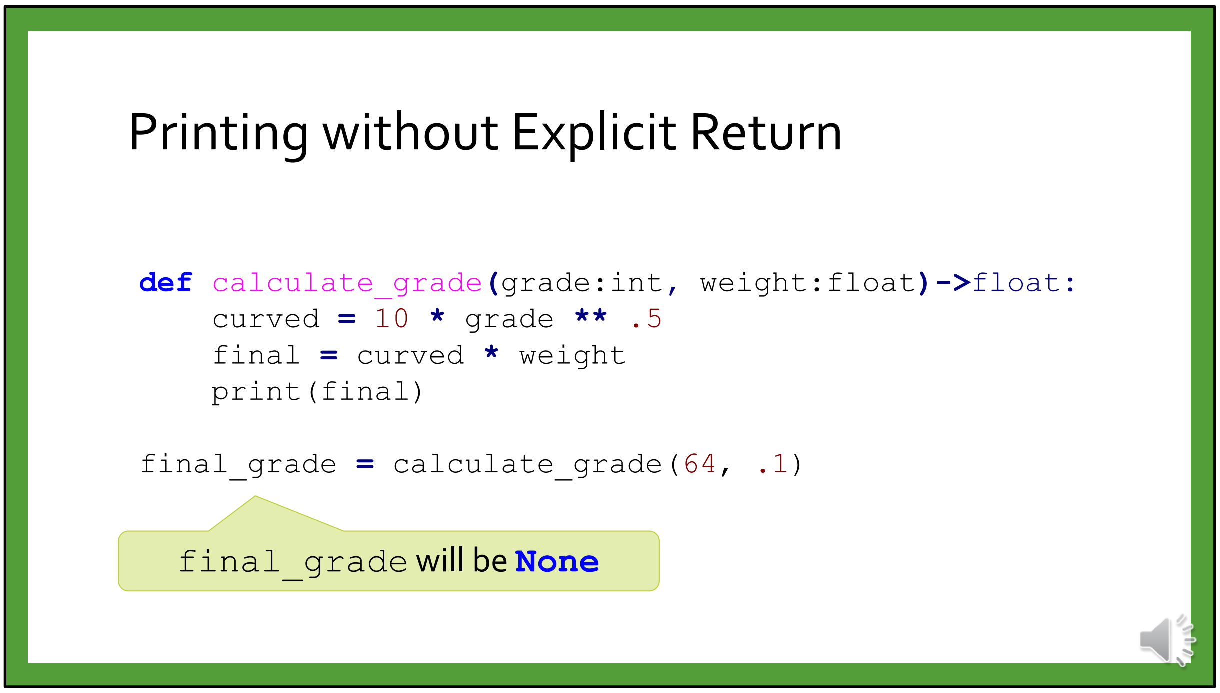
## Returning

```
def calculate_grade(grade:int, weight:float)->float:
    curved = 10 * grade ** .5
    print(curved)
    final = curved * weight
    return final
```

~~calculate_grade(64, .1)~~

```
          8.0
```

When a function is called, execution jumps to somewhere else in the program.
Some computations occur, and sometimes a result emerges.
When we want to use this result in the calling location, we need to return it.
When we use the return statement, execution jumps back to where the function was called.
The value returned is substituted in place of the function call.



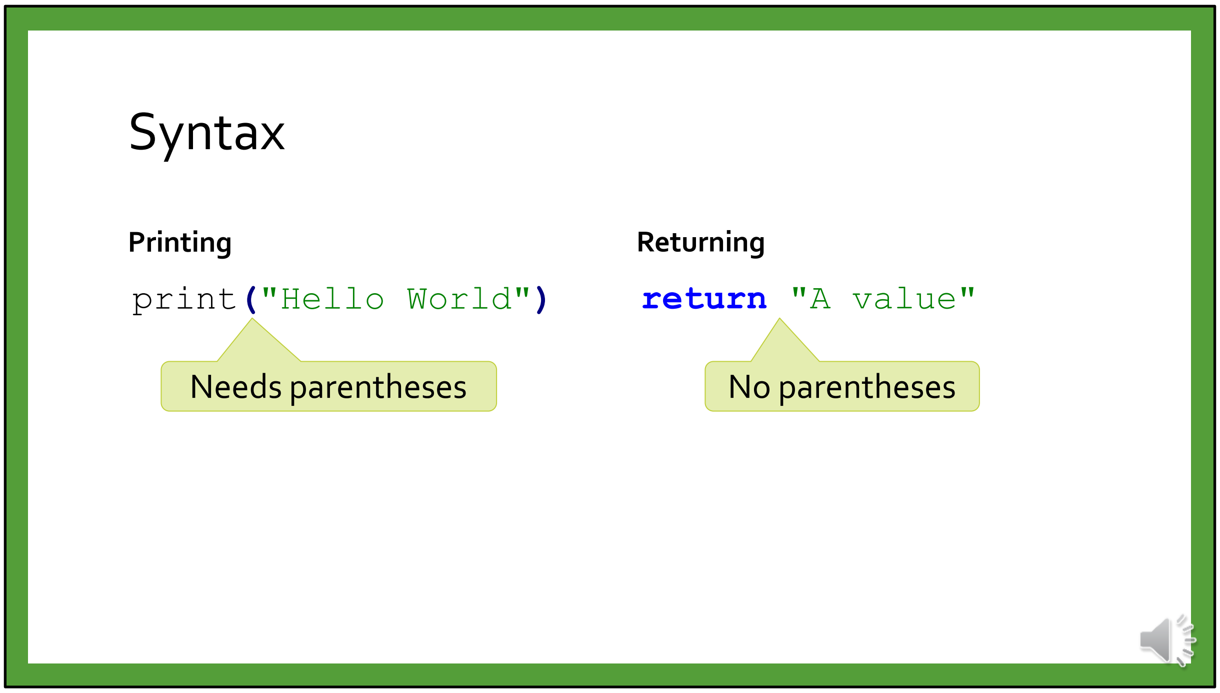
## Printing without Explicit Return

```
def calculate_grade(grade:int, weight:float)->float:
    curved = 10 * grade ** .5
    final = curved * weight
    print(final)

final_grade = calculate_grade(64, .1)
```

`final_grade` will be **`None`**

So what happens if we try to print instead of returning?
A function in Python always has to return something, so by default it returns None.
You will often find mysterious errors caused by forgetting to return!



## Syntax

**Printing**

```
print("Hello World")
```

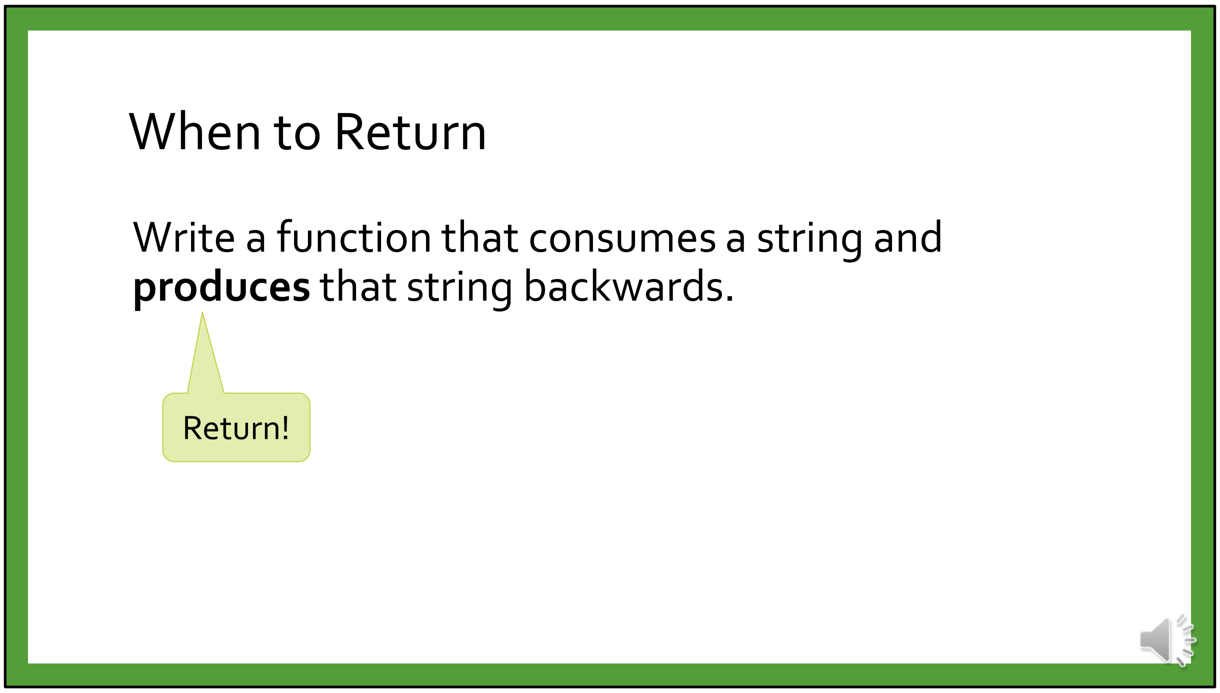
Needs parentheses

**Returning**

```
return "A value"
```

No parentheses

Let's look at the syntactic differences between printing and returning.
Return is a special keyword statement.
Print is a built-in function.
This is why print needs parentheses and return does not.
Code inside of a function can print, return, or both.
But code outside of a function can print, but it cannot return.



## When to Return

Write a function that consumes a string and **produces** that string backwards.

Return!

When someone tells you to write a function to produce a certain value, you will need to return.
The return needs to be the last thing that the function does, because you can only return from a function once.
You only ever return inside functions, to the place where the function was called.
In general, we return data when another part of the program needs that information.