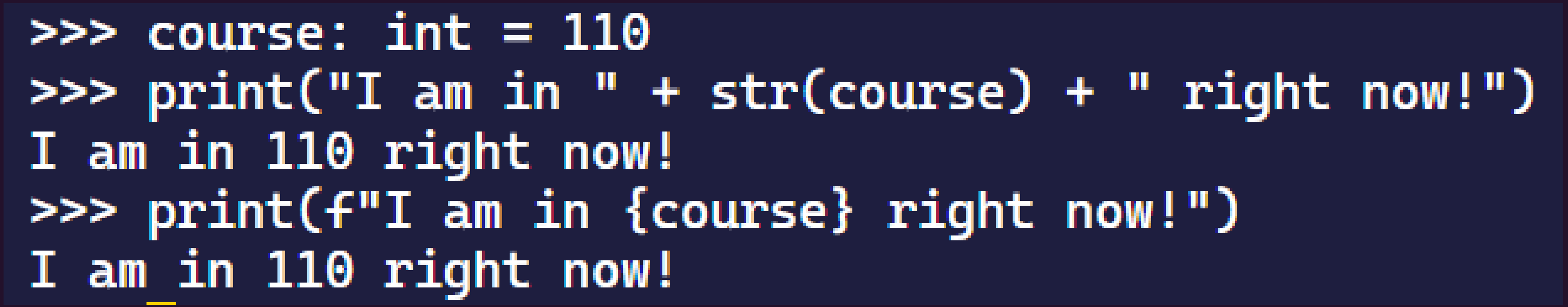
String Interpolation with **f-Strings**

Format Stings in Python

- Building up str values via concatenation can involve a lot of syntax (and work)!
  - Especially when you're concatenating non-str types and must construct str values

- Python has a special kind of string literal called a Format String or f-string
  - It doesn't give you any new capabilities over concatenation, it's just "syntactic sugar"

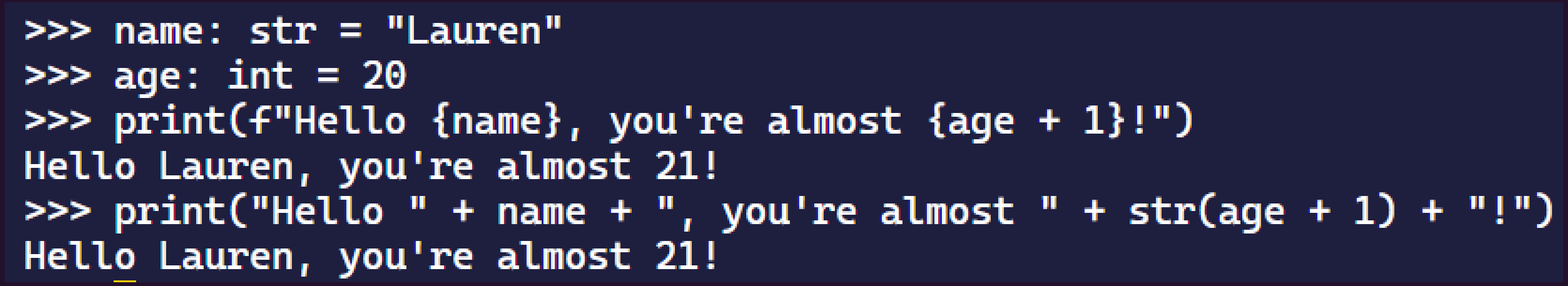
- Consider the following examples:



```
>>> course: int = 110
>>> print("I am in " + str(course) + " right now!")
I am in 110 right now!
>>> print(f"I am in {course} right now!")
I am in 110 right now!
```

How to write an f-String Literal

1. Place an `f` before the opening quotation of a `str` literal
2. Surround any *expression* in curly braces
   - Many curly-brace surrounded expressions can be in same f-String



```
>>> name: str = "Lauren"
>>> age: int = 20
>>> print(f"Hello {name}, you're almost {age + 1}!")
Hello Lauren, you're almost 21!
>>> print("Hello " + name + ", you're almost " + str(age + 1) + "!")
Hello Lauren, you're almost 21!
```

When an f-String literal is evaluated, each curly brace expression is:

1. evaluated as if it were a normal expression
2. converted to a str
3. concatenated into the string literal

String Interpolation

- This concept is commonly called String Interpolation
- Most modern programming languages have some syntax for doing this!
- More powerful things you can do in f-Strings are beyond our scope
  - For more, refer to this guide: https://realpython.com/python-f-strings/