

# Strings



# Strings



# Basic Data Types

Strings

Integers

Booleans

FLOATS



# Strings

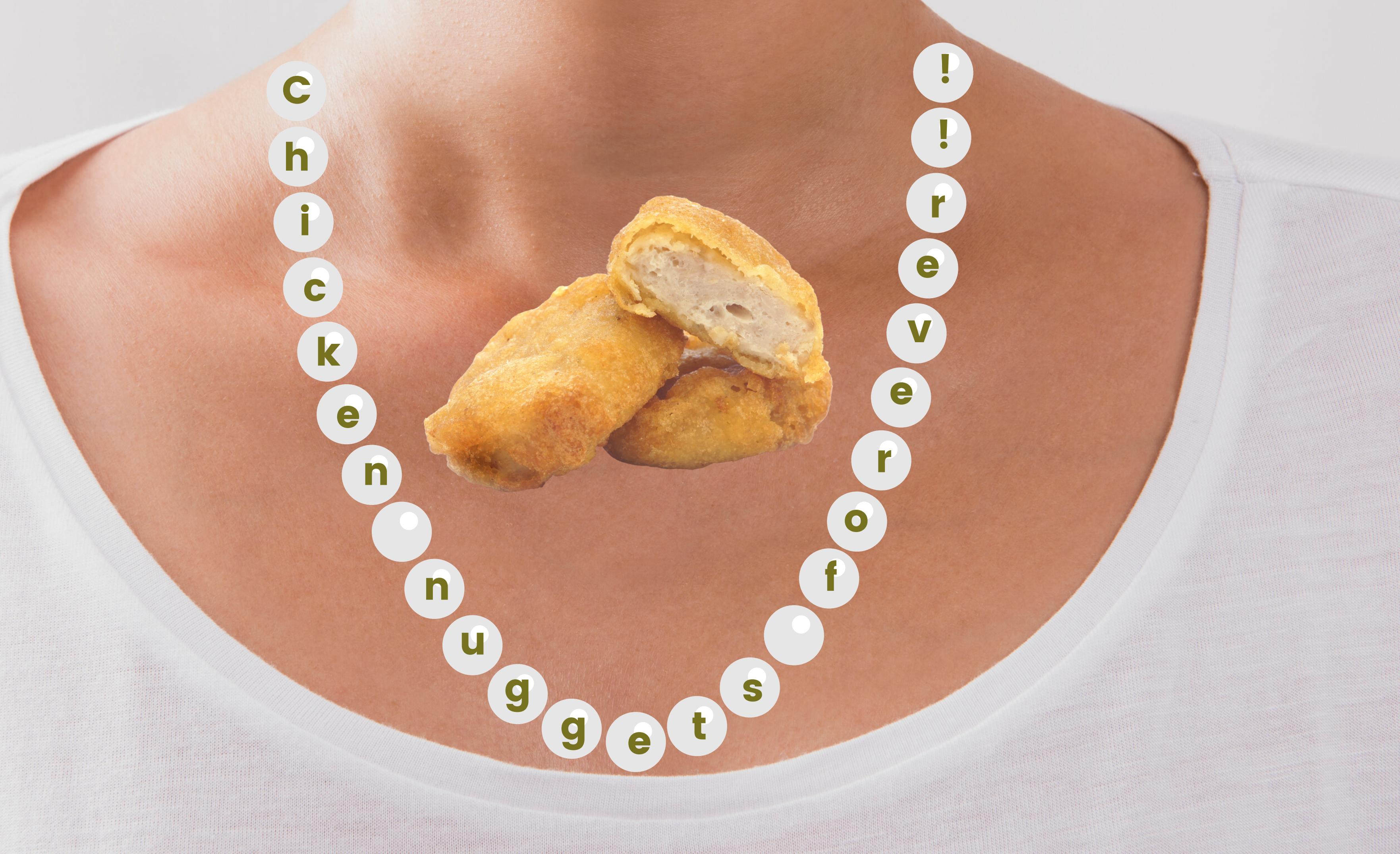
**"STRINGS OF CHARACTERS"**

Strings are a textual datatype and must be wrapped in quotes

c  
h  
i  
c  
k  
e  
n

n  
u  
g  
g  
e  
t  
s

!  
!  
r  
e  
v  
e  
r  
o  
f



chicken nuggets!



```
color = "Magenta"
```



```
twitter_handle = '@POTUS'
```



```
url = "www.reddit.com/r/formula1/"
```

# Quotes

A screenshot of a Python terminal window. The command prompt shows three colored dots (red, yellow, green) followed by '>>>'. The string 'Hello World!' is entered in red. A green checkmark is positioned in the top right corner of the terminal window.

>>> 'Hello World!'

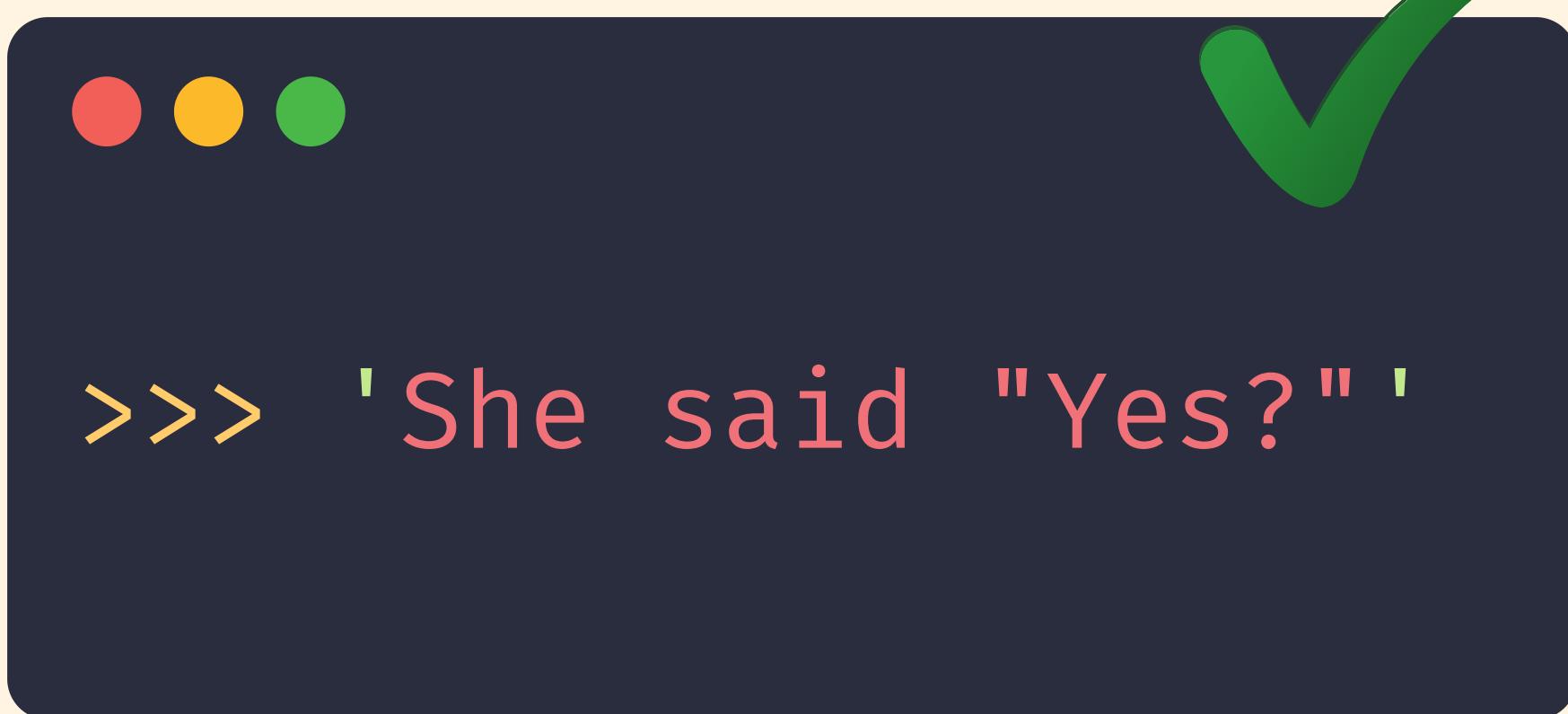
Start      String      End

A screenshot of a Python terminal window. The command prompt shows three colored dots (red, yellow, green) followed by '>>>'. The string 'The cat's toy' is entered in red. A large red 'X' is positioned in the top right corner of the terminal window.

>>> 'The cat's toy'

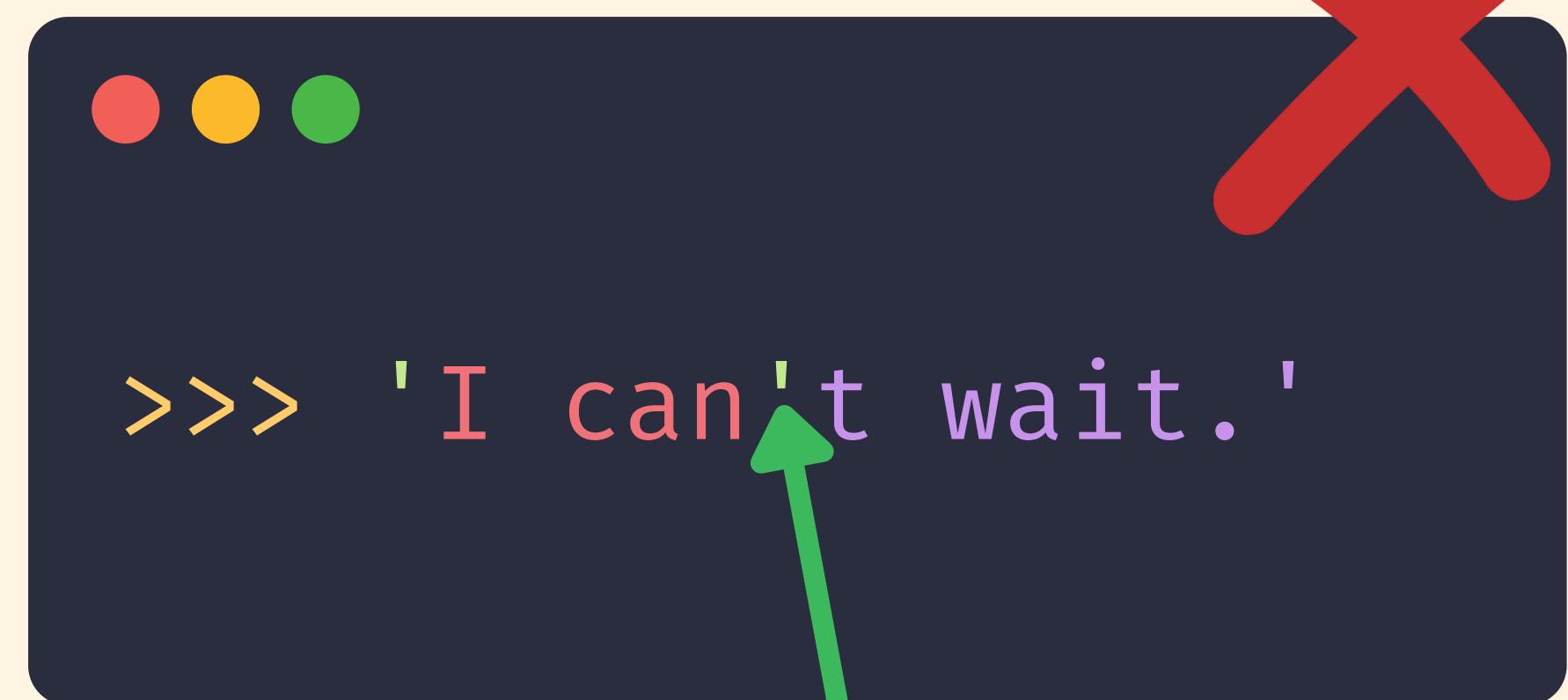
Start      String      End      Error

# Single Quotes



A dark blue terminal window icon with three colored dots (red, yellow, green) in the top-left corner. A large green checkmark is positioned in the top-right corner of the window area.

```
>>> 'She said "Yes?" '
```

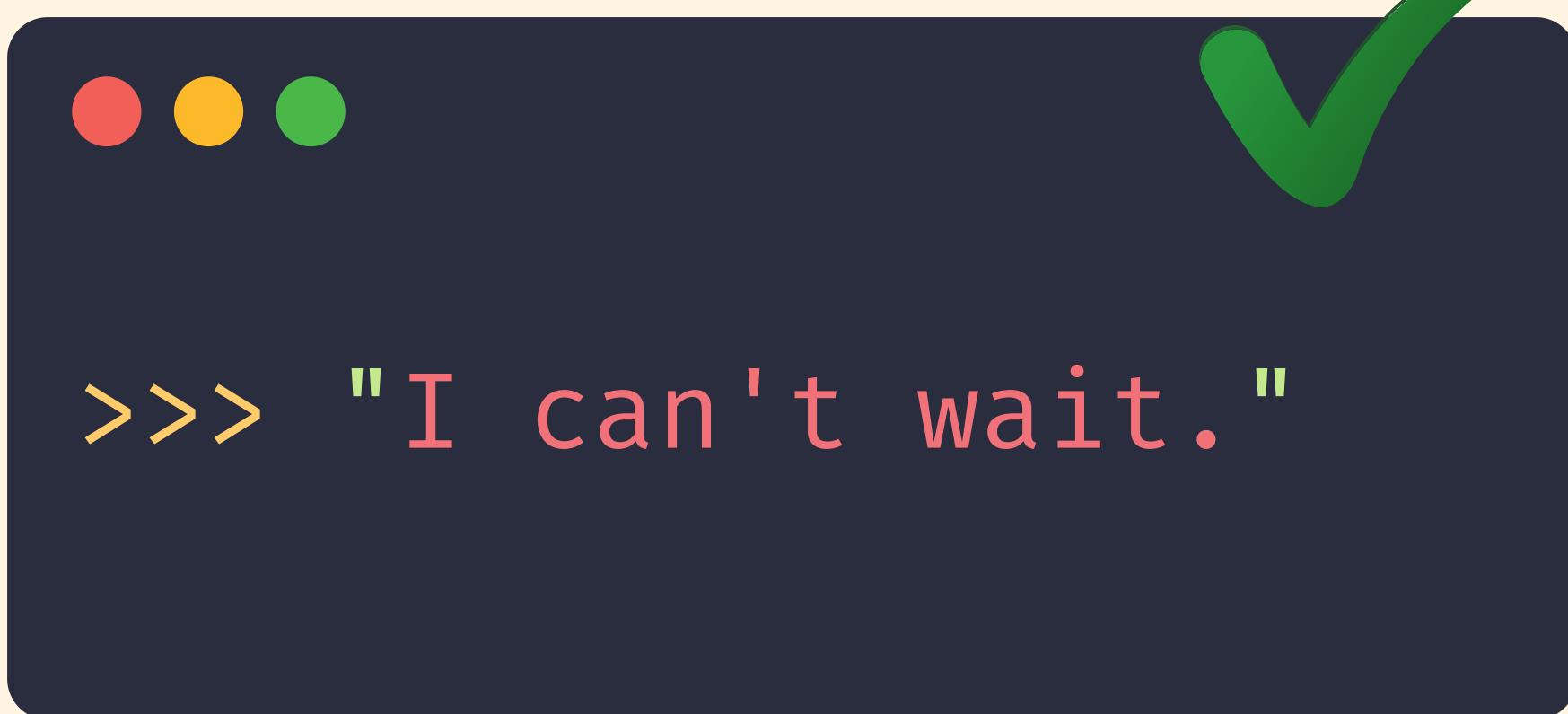


A dark blue terminal window icon with three colored dots (red, yellow, green) in the top-left corner. A large red X is positioned in the top-right corner of the window area.

```
>>> 'I can't wait.'
```

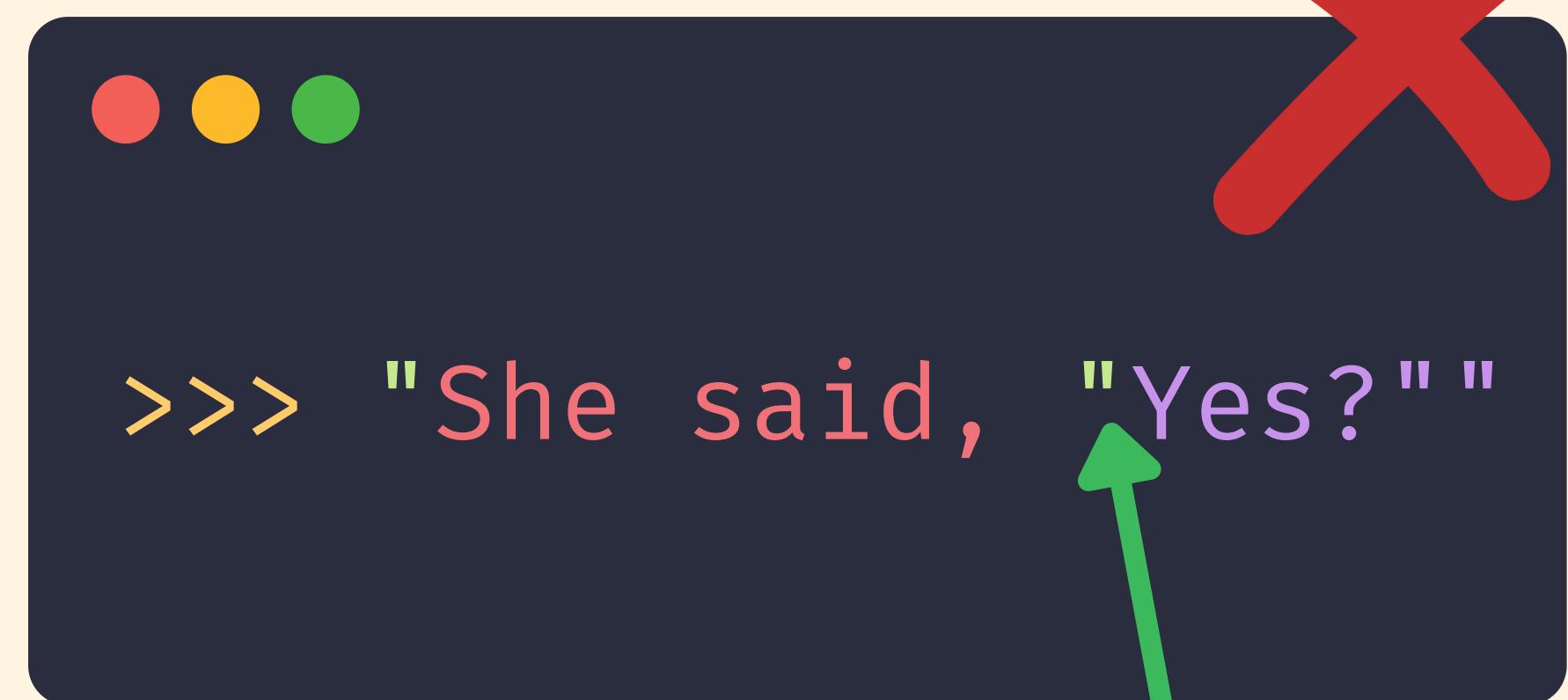
End

# Double Quotes



A dark blue terminal window icon featuring three colored dots (red, yellow, green) in the top-left corner. A large green checkmark is positioned in the top-right corner of the window area.

```
>>> "I can't wait."
```



A dark blue terminal window icon featuring three colored dots (red, yellow, green) in the top-left corner. A large red X is positioned in the top-right corner of the window area.

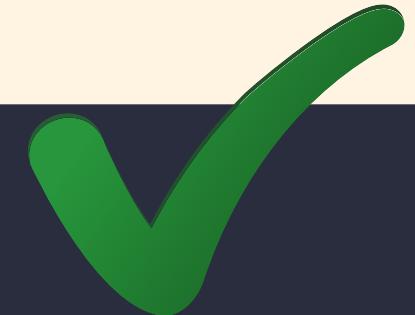
```
>>> "She said, "Yes?""
```

End



# Triple Quotes

## Single Or Double



```
>>> """Colt's sourdough bread is  
the best," Paul Hollywood stated."""
```

# Print

The `print()` function prints out any values we pass to it to "standard output". It does not return anything.



```
>>> print("hello")
```

# Escape Characters



Newline - \n

Double Quote - \"

Tab - \t

Single Quote - \'

Backslash - \\

# Concatenation

We can concatenate strings together by using the plus sign. No space will be added between them.

```
>>> 'pan' + 'cake'  
'pancake'
```

# Multiplication

We can also multiply a string by a number,  
which will repeat that string.

```
>>> 'ha'*4  
'hahahaha'
```

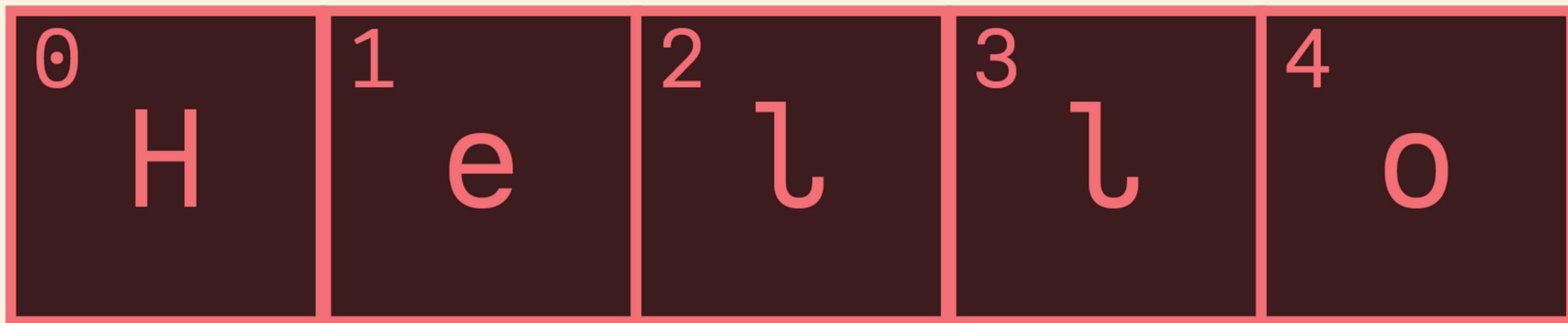
# None

None is a special value in Python that denotes the **lack of value**. It is not the same as zero or an empty string (those are still values).

```
>>> user = None
```

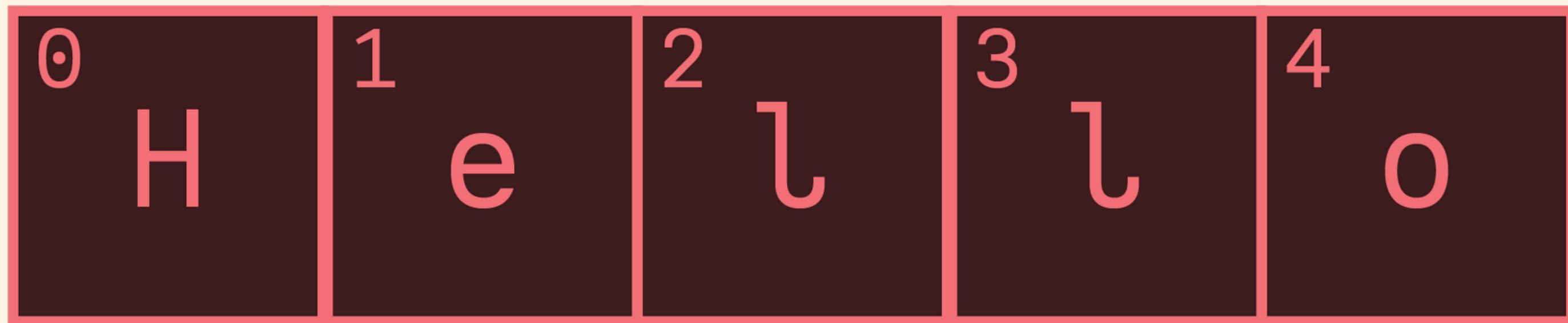
==

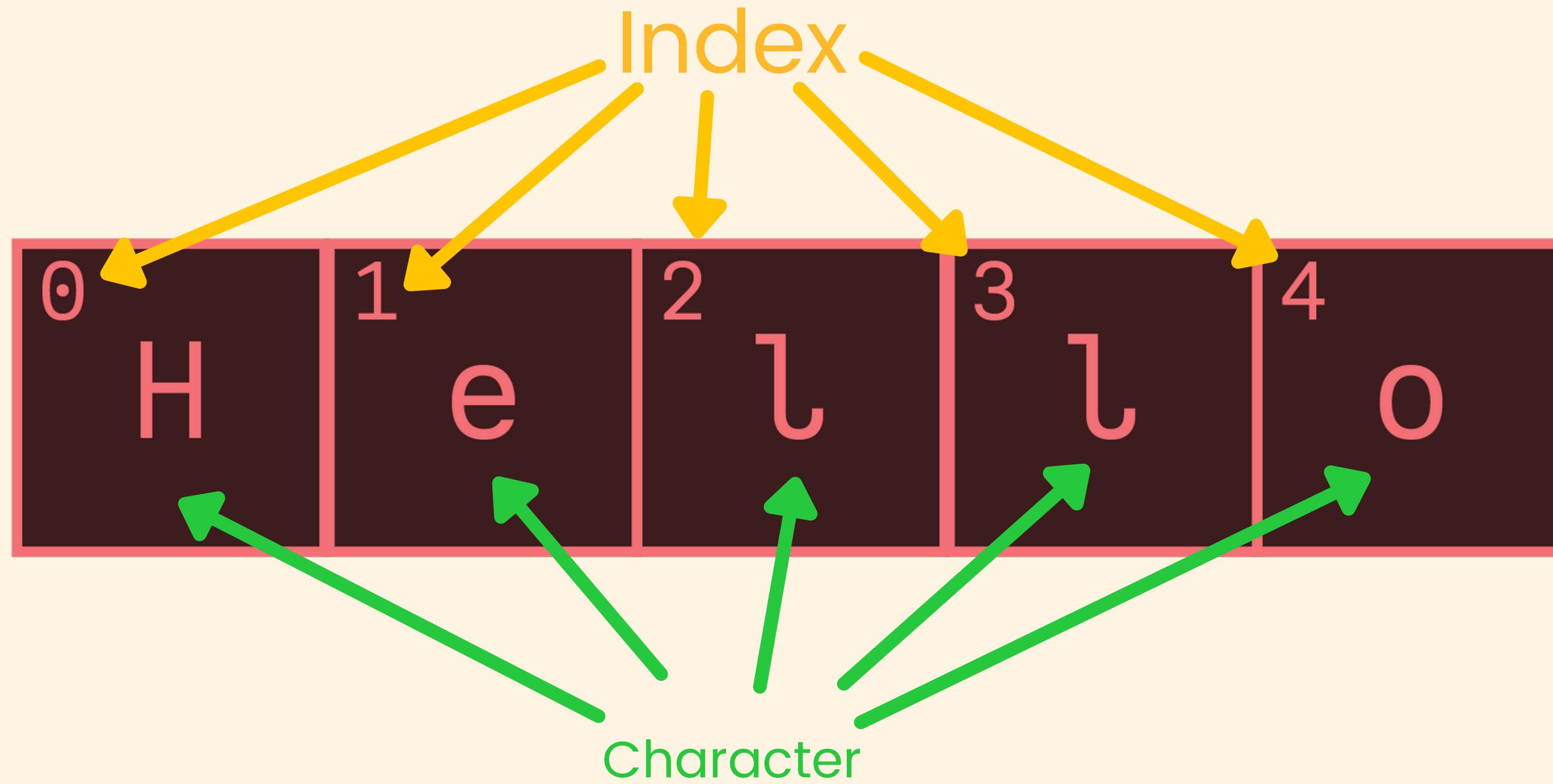
# Strings Are Ordered



==

# Strings Are Indexed

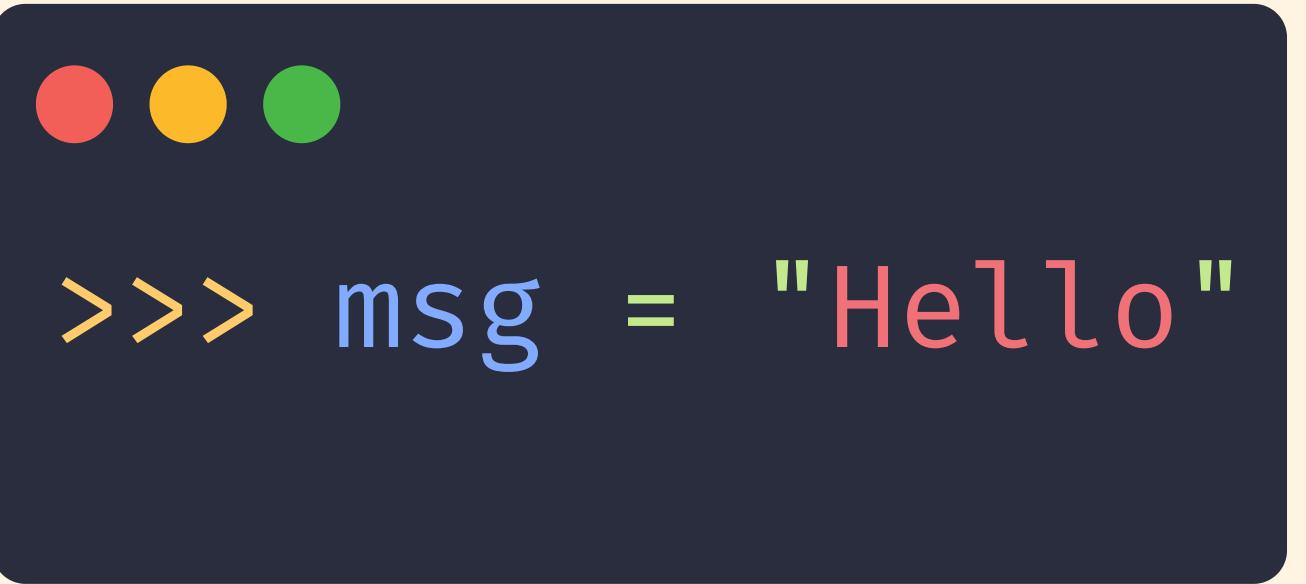




≡

# What is a name?

What is a name?  
That which we  
call a string  
by any other name  
would still say the same thing.



```
>>> msg = "Hello"
```



# How Variables Work

Your Code

```
>>> msg = "Hello"
```

Names

msg  Hello

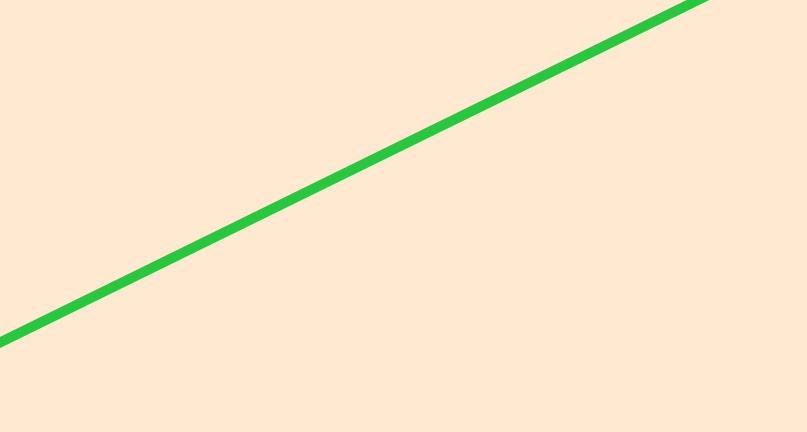
Objects

# How Variables Work

Your Code

```
>>> msg = "Hello"  
>>> greet = "Hello"
```

Names

msg       Hello  
greet     Hello

Objects

# How Variables Work

## Your Code

```
>>> msg = "Hello"  
>>> greet = "Hello"  
>>> msg = "Hello!!!"
```

## Names

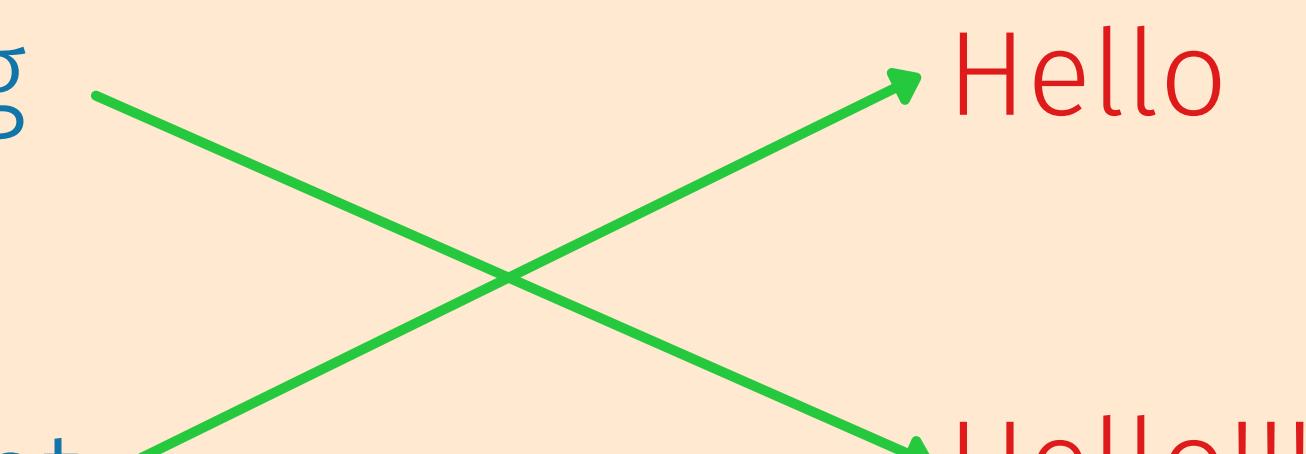
msg

greet

## Objects

Hello

Hello!!!



# Indexes



```
>>> msg = "I <3 Cats"
>>> msg[0]
'I'
>>> msg[5]
'C'
```

A screenshot of a terminal window showing Python code. The code defines a string `msg` with the value "I <3 Cats". It then prints the first character at index 0 ('I') and the character at index 5 ('C'). A brown arrow points from the bottom right towards the index 5 in the last line of code.

0 1 2 3 4 5 6 7 8

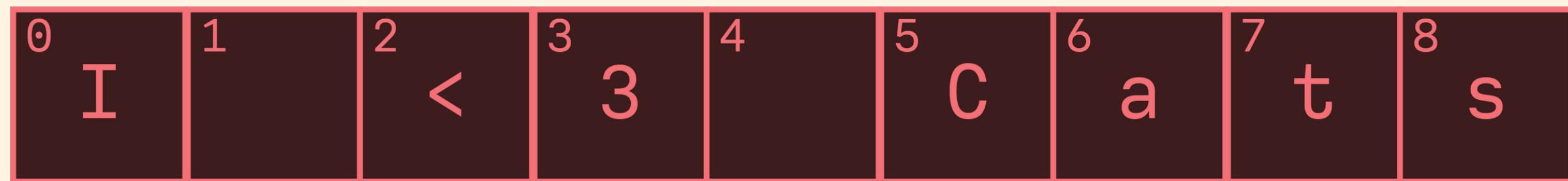
0	I	1	2	<	3	3	4	5	C	6	a	7	t	8	s
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

0 1 2 3 4 5 6 7 8

0	I	1		2	<	3	3	4		5	C	6	a	7	t	8	s
---	---	---	--	---	---	---	---	---	--	---	---	---	---	---	---	---	---

-9 -8 -7 -6 -5 -4 -3 -2 -1

# Slices



```
>>> msg[2:6]
'<3 C'
```



# Slices with a Step

