Operators

Assignment Operator

Python uses `=` for assignment.

```
In [ ]: my_var = 1
```

Math Operators

Python uses mathematical operators `+`, `-`, `*`, `/` for 'sum', 'substract', 'multiply', and 'divide'. These operators return numbers.

In [ ]:
```
print(2 + 3)
```

In [ ]:
```
div_result = 4 / 2
print(div_result)
```

More Math

Python also has `**` for exponentiation and `%` for remainder (called modulus). These also return numbers.

```
In [ ]: 2 ** 3
```

```
In [ ]: 17 % 7
```

Boolean Logic

Python has `and`, `or` and `not` for boolean logic. These operators return booleans.

In [ ]:
```
True and True
```

In [ ]:
```
True and not False
```

Clicker #1

What will the following boolean expression evaluate as:

True and not True or False

- a) True
- b) False
- c) None
- d) This code will fail

Clicker Question Answer

```
In [ ]: True and not True or False
```

Comparison Operators

Python has comparison operators `==`, `!=`, `<`, `>`, `<=`, and `>=` for value comparisons. These operators return booleans.

```
In [ ]: True == True
```

```
In [ ]: 12 >= 13
```

Comparing Identity Operators

Python uses `is` and `is not` to compare identity. These operators return booleans.

In [ ]:
```
a = 927
b = a
c = 927
```

In [ ]:
```
print(a is b)
print(c is a)
```

In [ ]:
```
a == b == c
```

String Concatenation

Operators sometimes do different things on different data. For example, `+` on strings does concatenation.

```
In [ ]: 'a' + 'b' + 'c'
```

Chaining Operators

Operators and variables can also be chained together into arbitrarily complex expressions.

In [ ]:
```
# Note that you can use parentheses to chunk sections
(13 % 7 >= 7) and ('COGS' + '18' == 'COGS18')
```

Clicker #2

What will the following expression evaluate as:

```
2**2 >= 4 and 13%3 > 1
```

- a) True
- b) False
- c) None
- d) This code will fail

Clicker Question Answer

```
In [ ]: 2**2 >= 4 and 13%3 > 1
```