

# Python Basics: Takeaways

by Dataquest Labs, Inc. - All rights reserved © 2018

## Syntax

---

### COMMON ARITHMETIC OPERATORS

- Parentheses `()` : `(5 / 5) + 5`
- Exponent `**` : `65 ** 5`
- Multiplication `*` : `5 * 5`
- Division `/` : `5 / 5`
- Addition `+` : `5 + 5`
- Subtraction `-` : `5 - 5`

### COMMON VARIABLE OPERATIONS

- Assigning a value directly to a variable:

```
integer_val = 5  
float_val = 5.0  
string_val = "5"
```

- Assigning the result of a calculation to a variable:

```
total = 5 + 5  
average = (5 + 5 + 5) / 3
```

### DISPLAYING VALUES

- Displaying a value:

```
integer_val = 5  
print(5)  
print(integer_val)  
print(5 + 5 + 5)
```

- Displaying a value's data type:

```
integer_val = 5  
print(type(integer_val))
```

---

## LIST OPERATIONS

- Creating an empty list:

```
crime_rates = []
```

- Creating a list with values:

```
crime_rates = [749, 371, 828, 503, 1379]
```

- Appending a value to a list:

```
crime_rates = []  
crime_rates.append(749)  
crime_rates.append(371)
```

- Accessing individual elements in a list:

```
crime_rates = [749, 371, 828, 503, 1379]  
cr_first = crime_rates[0]  
cr_third = crime_rates[2]
```

- Working with the length of a list:

```
crime_rates = [749, 371, 828, 503, 1379]  
length = len(crime_rates)  
last_element = crime_rates[length - 1]
```

- Accessing slices of values in a list:

```
crime_rates = [749, 371, 828, 503, 1379]  
cr_slice = crime_rates[0:3] # Values at 0, 1, 2
```

## Concepts

- When evaluating expressions, Python uses the [order of operations](#) rules from mathematics.
- Every value in Python has a data type associated with it. The common data types are:
  - Strings: `"6"`
  - Integers: `6`
  - Floats: `6.0`

## Resources

- [Documentation on all arithmetic operators](#)
- [List of reserved words in Python](#)
- [Documentation on lists](#)



Takeaways by Dataquest Labs, Inc. - All rights reserved © 2018