## 0. Commenting & Printing

```
# This is a comment
print()
```

| Operating System | Shortcut |
|---|---|
| Mac OS X | CMD + / |
| Windows | CTRL + / |

## 1. Data Types

| Data Type | Description |
|---|---|
| `int` | Integer numbers |
| `float` | Decimal numbers |
| `str` | Sequence of characters bounded by single or double quotes |
| `bool` | `True` or `False` |
| `None` | Nothing – represents absence of a value |

`type()` checks a value's data type

| Function | Description |
|---|---|
| `int()` | Casts value as an int |
| `float()` | Casts value as a float |
| `str()` | Casts value as a str |

## 2. Variables

Declare a variable with `=`

**Variable Naming:**

1. Can't start with a number or special character
2. No spaces allowed
3. Use only letters, numbers, and underscores _
4. Be descriptive with names

**Assembling `str` w/variable using f-strings:**

```
print(f"This is an f-string w/a {variable}.")
```

## 3. Operators

**Arithmetic Operators**

| Operator | Description |
|---|---|
| `+` | Addition |
| `-` | Subtraction |
| `*` | multiplication |
| `/` | division |
| `**` | exponent |

**Comparison Operators**

| Operator | Description |
|---|---|
| `==` | Equal in value |
| `!=` | Not equal in value |
| `>` | Greater than |
| `<` | Less than |
| `>=` | Greater than or equal to |
| `<=` | Less than or equal to |

## 4. Data Structures

| Property | `list` | `tuple` | `dict` | `set` |
|---|---|---|---|---|
| **Ordered?** | Yes | Yes | Yes | No |
| **Change items?** | Yes | No | Yes | Yes |
| **Duplicate items?** | Yes | Yes | No duplicate keys | No |

## 5. Lists

```
mylist = [1, 2, 3]
```

**Working with lists**

| Task | Syntax |
|---|---|
| Find list length | `len(mylist)` |
| Access list items with indices | `mylist[0]` |
| Add item to end of list | `mylist.append(item)` |
| Add item to specific index | `mylist.insert(index, item)` |
| Remove item from list by value | `mylist.remove(value)` |
| Remove item from list by index | `mylist.pop(index)` |
| Empty a list | `mylist.clear()` |

## 6. Tuples

```
mytuple = (1, 2, 3)
```

**Working with tuples**

| Task | Syntax |
|---|---|
| Find tuple length | `len(mytuple)` |
| Access tuple items with indices | `mytuple[0]` |
| Remove/add/change item from tuple | `ERROR: Can't change tuple` |

## 7. Dictionaries

```
mydict = {"Start": 1,
          "Middle": 2,
          "End": 3}
```

**Working with dictionaries**

| Task | Syntax |
|---|---|
| Find number of mappings | `len(mydict)` |
| Use keys to access dict values | `mydict["Start"]` |

## 8. Sets

```
myset = {1, 2, 3}
```

**Working with sets**

| Task | Syntax |
|---|---|
| Find set length | `len(myset)` |
| Access items with indices | `ERROR: Sets are unordered so cannot use indices` |
| Add duplicate items | `No error – sets will just ignore duplicates` |
| Empty a list | `mylist.clear()` |

## 9. For Loops

```
for item in data_structure:
    # do something to each item
```

## 10. If Statements

```
if condition_1:
    # do this if condition_1 True
elif condition_2:
    # do this if condition_1
    False & condition_2 True
else:
    # do this if none of above
    True
```

| Operator | Description |
|---|---|
| `and` | Returns `True` if both statements true |
| `or` | Returns `True` if at least one of the statements is true |

**Loop Interruptions**

| Syntax | Description |
|---|---|
| `break` | Exists a loop |
| `continue` | Skips to the next iteration of a loop |

## 11. Functions

```
def function_name(parameters):
    """ Description here
    Parameters
    ----------
    parameters: description of
    parameters

    Return
    ----------
    what is returned

    """
    # do task here
    return value
```

**Accessing a Function's Docstring**

| Syntax | Description |
|---|---|
| `function_name?` | Returns function's docstring |