Python Course Cheat Sheet 2

Tuple

A tuple is a collection of objects which is ordered and immutable. Tuples are similar to lists, the main difference is the immutability. In Python tuples are written with round brackets and comma separated values.

```
my_tuple = ("Max", 28, "New York")
```

Create a tuple

Tuples are created with round brackets and comma separated values. Or use the built-in tuple function.

```
tuple_1 = ("Max", 28, "New York")
```

Access elements

You access the tuple items by referring to the index number. Note that the indices start at 0.

```
item = tuple_1[0]
print(item)
```

Note: You can’t add or change items to tuple.

Iterating

```
# Iterating over a tuple by using a for in loop
for i in tuple_1:
    print(i)
```

Useful methods

```
my_tuple = ('a','p','p','l','e',)

# len() : get the number of elements in a tuple
print(len(my_tuple))

# count(x) : Return the number of items that is equal to x
print(my_tuple.count('p'))

# index(x) : Return index of first item that is equal to x
print(my_tuple.index('l'))

# repetition
my_tuple = ('a', 'b') * 5
print(my_tuple)

# concatenation
my_tuple = (1,2,3) + (4,5,6)
print(my_tuple)

# convert list to a tuple and vice versa
my_list = ['a', 'b', 'c', 'd']
list_to_tuple = tuple(my_list)
print(list_to_tuple)

tuple_to_list = list(list_to_tuple)
print(tuple_to_list)

# convert string to tuple
string_to_tuple = tuple('Hello')
print(string_to_tuple)
```

Dictionaries

A dictionary is a collection which is unordered, changeable and indexed. A dictionary consists of a collection of key-value pairs. Each key-value pair maps the key to its

associated value. A dictionary is written in braces. Each key is separated from its value by a colon (:), and the items are separated by commas.

```
my_dict = {"name":"Max", "age":28, "city":"New York"}
```

Create dictionary

```
my_dict = {"name":"Max", "age":28, "city":"New York"}
print(my_dict)
```

Access items

```
# call item with key name
name_in_dict = my_dict["name"]
print(name_in_dict)
```

Add and change items

Simply add or access a key and assign the value.

```
# add a new key
my_dict["email"] = "max@xyz.com"
print(my_dict)
```

Sets

A Set is an unordered collection data type that is unindexed, mutable, and has no duplicate elements. Sets are created with braces.

```
my_set = {"apple", "banana", "cherry"}
```

Create a set

Use curly braces or the built-in set function.

```
my_set = {"apple", "banana", "cherry"}
print(my_set)

# or use the set function and create from an iterable, e.g. list, tuple, string
my_set_2 = set(["one", "two", "three"])
my_set_2 = set(("one", "two", "three"))
print(my_set_2)
```

Add elements

```
my_set = set()

# use the add() method to add elements
my_set.add(42)
my_set.add(True)
my_set.add("Hello")

# note: the order does not matter, and might differ when printed
print(my_set)

# nothing happens when the element is already present:
my_set.add(42)
print(my_set)
```

Union and Intersection

```
odds = {1, 3, 5, 7, 9}
evens = {0, 2, 4, 6, 8}
primes = {2, 3, 5, 7}

# union() : combine elements from both sets, no duplication
# note that this does not change the two sets
```

```
u = odds.union(evens)
print(u)

# intersection(): take elements that are in both sets
i = odds.intersection(evens)
print(i)

i = odds.intersection(primes)
print(i)

i = evens.intersection(primes)
print(i)
```