Tuples, Sets, and Dictionaries

Tuples

Tuples are written with roundy brackets and with commas between each item[1]:

```
t = (1, 3, 5, 'hello')
```

The empty tuple is written `()`, and a tuple with one item must have a comma, e.g. `(0,)`.

The main difference between tuples and lists is that you can't change values that are within tuples, e.g. `t[3] = 4` – this code will give an error.

Sets

Sets are written with curly brackets.

They can't contain duplicate elements (duplicates entered will automatically be detected, and won't be entered).

Checking if something is in a set using `in` is fast.

Sets also support mathematical set operations like union and set difference.

- The empty set is created using `set()` because `{}` is the empty dictionary.
- You can use set comprehensions to create sets, like with list comprehensions.

Dictionaries

Dictionaries are unordered sets of key:value pairs. The keys must be unique within a dictionary. They're written with curly brackets:

```
d = {'France':'Paris', 'Ireland':'Dublin', 'England':'London'}
```

- `in` will check if a specific key is in a dictionary:

  `'France' in d` will evaluate to `True` for the dictionary above.
- For loops over a dictionary will loop through the keys in arbitrary order:

```
for key in d:
    print(key)
```

- Values in a dictionary can be accessed using square bracket notation, similar to list indices:

  `d['France']` will return 'Paris' from the dictionary above.

  You can also change values in the dictionary or add new pairs using this notation:

  `d['Spain'] = 'Madrid'`

  `d['Ireland'] = 'Cork'`

Handouts & Assignments

- Handout 17 - Tuples
- Handout 18 - Sets
- Handout 19 - Dictionaries
- Assignemnt 14 - Dictionaries

---

1. Technically you can write them without the brackets, but you need the brackets if you want to nest them.↩