DATA CULTURE SOCIETY
CDCS.ED.AC.UK

SCOTTISH GRADUATE SCHOOL FOR ARTS & HUMANITIES
Sgoil Ceumnachaidh na h-Alba airson Ealain agus Daonnachdan
2018 - 2028

DAY 3

SEMINAR 2

Hannah Claus

Research Assistant
The Ada Lovelace Institute

THE UNIVERSITY of EDINBURGH
Centre for Data, Culture & Society

DATA CULTURE SOCIETY

COFFEE BREAK

WE ARE GOING TO RESTART AT 11:00

DATA CULTURE SOCIETY
CDCS.ED.AC.UK

SCOTTISH GRADUATE SCHOOL FOR ARTS & HUMANITIES
Sgoil Ceumnachaidh na h-Alba airson Ealain agus Daonnachdan
2014 - 2024

COLLECTIONS

LISTS

```
planet0 = "Mercury"
planet1 = "Venus"
planet2 = "Earth"
planet3 = "Mars"
planet4 = "Jupyter"
planet5 = "Saturn"
planet6 = "Uranus”
```

LISTS

```
planet0 = "Mercury"
planet1 = "Venus"
planet2 = "Earth"
planet3 = "Mars"
planet4 = "Jupyter"
planet5 = "Saturn"
planet6 = "Uranus”
```

```
planets = ["Mercury", "Venus", "Earth",
"Mars", "Jupyter", "Saturn", "Uranus"]
```

WHY USE LISTS?

```
planets = ["Mercury", "Venus", "Earth", "Mars",
"Jupyter", "Saturn", "Uranus"]
```

- ✅ Count how many items are in the list
- ✅ Check if a specific item is in the list
- ✅ Find the location of a specific item
- ✅ Add and remove items
- ✅ Sort (alphabetically or otherwise)

Pluto (not a planet)

ALTERNATIVES TO LISTS

Tuple

planets = ("Mercury", "Venus", "Earth", "Mars", "Jupyter", "Saturn", "Uranus”)

- Uses () instead of []
- Same as a list, except it cannot be changed after creating it

ALTERNATIVES TO LISTS

Set

planets = {"Mercury", "Venus", "Earth", "Mars", "Jupyter", "Saturn", "Uranus”}

- Uses {} instead of [] (list) or () (tuple)
- Same as a list, except:
  - Every item is unique (items cannot be listed twice)
  - Order does not matter and will change (no indexing)

ALTERNATIVES TO LISTS

Dictionaries

- Used to store multiple pieces of information about one thing
- Uses key-value pairs: each piece of data (value) has a label (key)

mercury = {“name”: “Mercury”, “day_length”: 59, “hottest_temp”: 430}

ALTERNATIVES TO LISTS

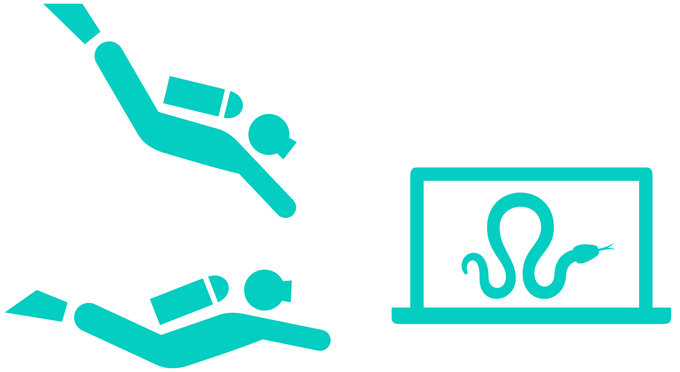
Dictionaries and Lists Together

- Combining lists and dictionaries is useful for real-world data: We often have multiple pieces of information about lots of different things and want to work with all of it at the same time!

```
planets = [
    {“name”: “Mercury”, “day_length”: 59, “hottest_temp”: 430},
    {“name”: “Venus”, “day_length”: 243.025, “hottest_temp”: 462},
    {“name”: “Earth”, “day_length”: 1, “hottest_temp”: 56.7},
    …
    ]
```

LETS GET PROGRAMMING

Session 7: [({collections})]



THE UNIVERSITY of EDINBURGH
Centre for Data, Culture & Society

DATA CULTURE SOCIETY

LUNCH BREAK

WE ARE GOING TO RESTART AT 13:30

DATA CULTURE SOCIETY
CDCS.ED.AC.UK

SCOTTISH GRADUATE SCHOOL FOR ARTS & HUMANITIES
Sgoil Ceumnachaidh na h-Alba airson Ealain agus Daonnachdan
2014 - 2024

LIST COMPREHENSIONS

WHAT ARE LIST COMPREHENSIONS

How do I get someone to pick the shirts from the wardrobe?

1. Say that it is the shirts you want,
2. For each item of clothing, check if it is a shirt,
3. If it’s a shirt, then take it out the wardrobe.

```
shirts = [
        item_of_clothing
        for item in wardrobe:
        if item == shirt
        ]
```

```
initial_list = [thing1, thing2, …]
variable_name = [
        <thing to get new list of>
        for item in initial_list:
        if item ==/>/< <some condition>
        ]
```

Some maths functions that may come in handy…

`max( ) / min()`

Get the largest/smallest element in a group. For letters it will mean 'highest/lowest in the alphabet'.

`len( )`

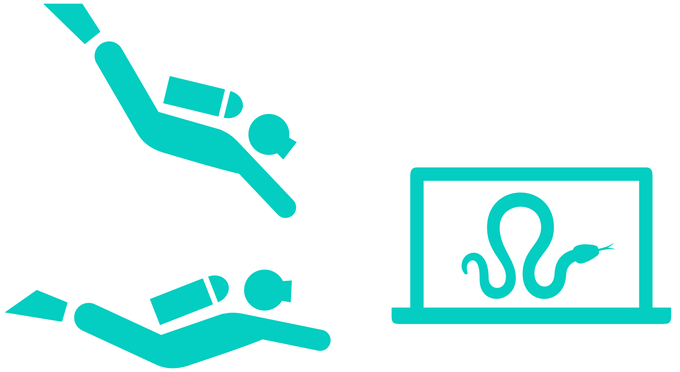
Size of the collection, can be used on lists, dicts, but also on strings.

`sum( )`

Combine all elements. Just used for numbers.

LETS GET PROGRAMMING

Session 8: Lists of Lists



THE UNIVERSITY of EDINBURGH
Centre for Data, Culture & Society

DATA CULTURE SOCIETY

COFFEE BREAK

WE ARE GOING TO RESTART AT 15:30

DATA CULTURE SOCIETY
CDCS.ED.AC.UK

SCOTTISH GRADUATE SCHOOL FOR ARTS & HUMANITIES
Sgoil Ceumnachaidh na h-Alba airson Ealain agus Daonnachdan
2014 - 2024

LOOPS

WHAT IS A LOOP?

Sometimes we are in the case where we want to do something over and over to a collection of items.

To do this we can use a loop to do ‘something’ using each of the items within a collection.

THE FAQ BIT

Why not just use a list comprehension?
Whilst a great syntactic feature of Python, we are slightly limited it what it can achieve.

Why not just use a function?
Typically a function is really good at executing a process on one thing. We may need a loop to get a function to perform on a collection still.

FOR AND WHILE

```
initial_list = [thing1, thing2, …]
```

```
for thing in initial_list:
   <do something>
```

```
counter = 0

while counter < len(initial_list):
   <do something to initial_list[counter]>
   counter += 1
```

BREAKING A LOOP

We may be in the situation whereby we want a loop to stop if we reach a certain condition. For this we can use the in-built command **‘break’**. We can also use **‘continue’** if we do not want a loop to do anything under a certain condition but do not want to the loop to stop.

BREAK THE LOOP

```
for item in collection:
    if some_condition:
        break  # Exit the loop
    if another_condition:
        continue  # Skip to the next iteration
```

LETS GET PROGRAMMING

Session 9: All FOR One, One FOR All