

**Starter:** There are three mistakes in this program. What are the mistakes and how would you fix them?

```
In [ ]: carrots = input('How many carrots do you have? ')
rabbits = 6

if rabbits < carrots:
    print('There are not enough carrots')
elif rabbits > carrots:
    print('There are too many carrots')
else:
    print('You have the right number of carrots')
```



Python Session 4

This session:

1. Lists
2. Dictionaries

**Lists**

**List:** an ordered collection of values

List are written inside square brackets and separated by commas

A list of integers

```
In [3]: lottery_numbers = [4, 8, 15, 16, 23, 42]
```

A list of strings

```
In [ ]: student_names = ['Diedre', 'Hank', 'Helena', 'Salome']
```

Lists can be made up of values of one or more data types

```
In [ ]: person = ['Jess', 32]
```

List values can be accessed using their **index** in square brackets

```
In [7]: student_names = ['Diedre', 'Hank', 'Helena', 'Salome']  
print(student_names[2])
```

Helena



List indexes start counting from 0

```
In [8]: student_names = [  
        'Diedre',    # index 0  
        'Hank',      # index 1  
        'Helena',    # index 2  
        'Salome'     # index 3  
    ]  
  
    print(student_names[0])
```

Diedre

You can also set the values in lists using their indexes, similar to how you would set a variable

```
In [10]: student_names = [  
    'Diedre',    # index 0  
    'Hank',      # index 1  
    'Helena',    # index 2  
    'Salome'     # index 3  
]
```

```
student_names[1] = 'Joshua'  
print(student_names)
```

```
['Diedre', 'Joshua', 'Helena', 'Salome']
```

**Exercise 4.1:** When I'm travelling in the winter I often forget to pack warm clothes. Let's write a program to help me to remember the right clothes.

The program should check if the first item in the `clothes` list is `"shorts"`. If it is it should change the value to `"warm coat"`.

**Extension:** Change the other items in the list to clothing more appropriate to winter if the first item is `shorts`.

```
In [12]: clothes = [  
    "shorts",  
    "shoes",  
    "t-shirt",  
    ]
```

```
['warm coat', 'shoes', 'jumper']
```

## More list indexing

How do we get the last item from the list?

```
In [29]: clothes = ["shorts", "shoes", "t-shirt", "dress", "hat"]  
         last_item = clothes[-1]  
         print(last_item)
```

hat

## List Slicing

List slicing enables us to get subsets of elements from lists, without using loops.

For example, if we want to get the first two items from our list:

```
In [30]: clothes = ["shorts", "shoes", "t-shirt", "dress", "hat"]  
         first_two_items = clothes[: 2]  
         print(first_two_items)  
  
['shorts', 'shoes']
```

Get middle two items

```
In [31]: clothes = ["shorts", "shoes", "t-shirt", "dress", "hat"]  
middle_items = clothes[2 : 4]  
print(middle_items)  
['t-shirt', 'dress']
```

Get last three items

```
In [32]: clothes = ["shorts", "shoes", "t-shirt", "dress", "hat"]  
last_items = clothes[2 :]  
print(last_items)  
['t-shirt', 'dress', 'hat']
```

**Exercise:** Create a list with six items. Save as variables and print:

1) the second item

2) the last item

3) the first two items

4) the second, third and fourth items

5) the last two items



## List Functions

There are functions designed for lists

- `len()`: the number of items in a list
- `max()`: The biggest value in a list
- `min()`: The smallest value in a list

```
In [3]: costs = [1.2, 4.3, 2.0, 0.5]
```

```
print(len(costs))  
print(max(costs))  
print(min(costs))
```

```
4  
4.3  
0.5
```

## Functions for changing the order of a list

- `sorted()`: Sorts the list
- `reversed()`: Reverses the order of a list

```
In [4]: costs = [1.2, 4.3, 2.0, 0.5]

print(sorted(costs))
print(list(reversed(costs))) # Need to convert back to a list with list()
```

```
[0.5, 1.2, 2.0, 4.3]
```

```
[0.5, 2.0, 4.3, 1.2]
```

Sorted() can also be used on lists of strings to sort items alphabetically

```
In [5]: animals = ['dog', 'cat', 'rabbit', 'hamster', 'mouse', "bird"]  
  
sort_animals = sorted(animals)  
print(sort_animals)
```

```
['bird', 'cat', 'dog', 'hamster', 'mouse', 'rabbit']
```

**Exercise 4.2:** Make a list of game scores. Using list functions write code to output information of the scores in the following format:

```
Number of scores: 10  
Highest score: 200  
Lowest score: 3
```

**Extension:** Output all of the scores in descending order

**append() and in**

You can check if an value is in a list using the `in` operator. If the value is in the list it will result in `True` and `False` if it is not.

```
In [20]: student_name = input('Which student are you looking for? ')

students = [
    'Diedre', 'Hank', 'Helena', 'Salome',
]

if student_name in students:
    print('{} is in the class'.format(student_name))
else:
    print('{} is not in the class'.format(student_name))
```

```
Which student are you looking for? Bob
Bob is not in the class
```

To check if an item is not in a list, you can use the `not in` operator. If the value is not in the list it will result in `True` and `False` if it is.

```
In [15]: fridge = [  
        'cheese',  
        'pizza',  
        'coke',  
    ]  
  
    if 'milk' not in fridge:  
        print('You have no milk in the fridge')
```

You have no milk in the fridge



The `.append()` method is used to add items to a list

```
In [9]: students = [  
        'Diedre', 'Hank', 'Helena', 'Salome',  
        ]  
        student_name = input('What is the name of the new student? ')  
  
        students.append(student_name)  
  
        print(students)
```

```
What is the name of the new student? bob  
['Diedre', 'Hank', 'Helena', 'Salome', 'bob']
```

**Exercise 4.3:** Whenever I'm shopping and I buy some bread I always forget to buy butter. Create a list and if 'bread' is in the list, add 'butter' to the shopping list.

Try running the program with and without bread in the list to check that your program works.

Remember the `in` operator checks if an item is in a list and the `.append()` method adds an item to a list.

**Extension:** Only add butter to the list if it is not already in the list, using the operator `not in`.

For Loops ♥ Lists

## Using lists and for loops together

```
In [12]: student_names = ['Diedre', 'Hank', 'Helena', 'Salome']  
  
for student_name in student_names:  
    print(student_name)
```

```
Diedre  
Hank  
Helena  
Salome
```

## Counting the total number of items in a list using a for loop

```
In [16]: student_names = ['Diedre', 'Hank', 'Helena', 'Salome']  
count = 0  
  
for student_name in student_names:  
    count = count + 1  
  
print(count)
```

Normally we would do this without a for-loop using `len()` , which returns the length of a list.

```
In [15]: student_names = ['Diedre', 'Hank', 'Helena', 'Salome']  
print(len(student_names))
```

4

**Exercise 4.4:** I want to work out how much money I've spent on lunch this week. I've created a list of what I spent each day.

Write a program that uses a `for` loop to calculate the total cost

```
In [14]: costs = [8.30, 7.12, 5.01, 1.00, 0.99, 5.92, 3.50]  
total_cost = 0
```

**Extension:** Work out the average that I spend on lunch for the week

There is an easier way to do the last program without a for loop. The `sum()` function can be used to add up all of the values in a list:

```
In [21]: costs = [8.30, 7.12, 5.01, 1.00, 0.99, 5.92, 3.50]
total = sum(costs)

print(total)
```

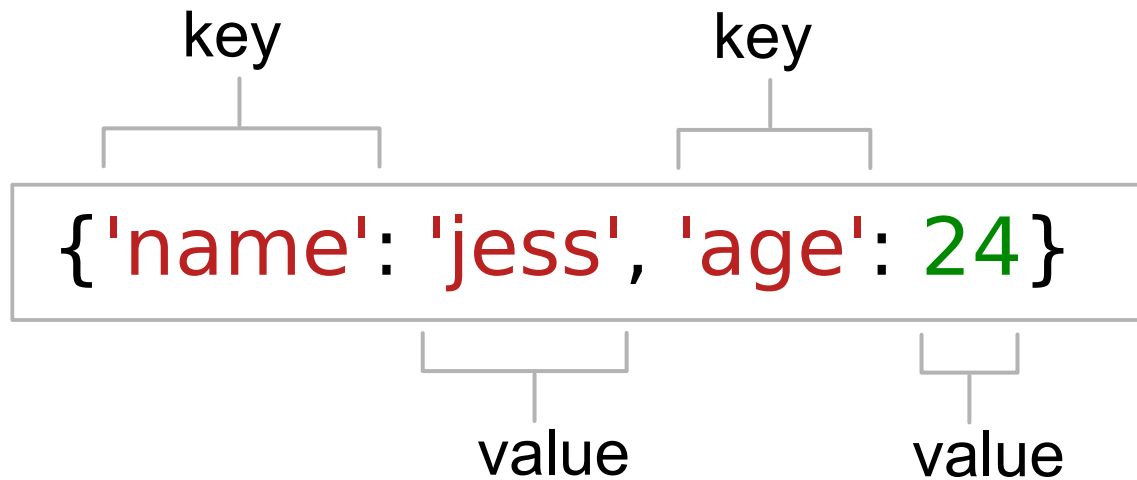
```
31.839999999999996
```



**Dictionaries**

**Dictionary:** Stores a collection of labelled items. Each item has a *key* and a *value*

```
In [ ]: person = {  
        'name': 'Jessica',  
        'age': 23,  
        'height': 172  
    }
```



Values in a dictionary are accessed using their keys

```
In [23]: person = {  
          'name': 'Jessica',  
          'age': 23,  
          'height': 172  
        }  
  
        print(person['name'])  
        print(person['age'])  
        print(person['height'])
```

```
Jessica  
23  
172
```

**Exercise 4.5:** Print the values of `name` , `post_code` and `street_number` from the dictionary

```
In [17]: place = {  
    'name': 'The Anchor',  
    'post_code': 'E14 6HY',  
    'street_number': '54',  
    'location': {  
        'longitude': 127,  
        'latitude': 63,  
    }  
}
```

**Extension:** Print the values of `longitude` and `latitude` from the inner dictionary

## Dictionarys in Lists

Putting dictionaries inside a list is very common

```
In [21]: people = [  
    {'name': 'Jessica', 'age': 23},  
    {'name': 'Trisha', 'age': 24},  
    ]  
  
    for person in people:  
        print(person['name'])  
        print(person['age'])
```

```
Jessica  
23  
Trisha  
24
```

**Exercise 4.6:** Using a for loop, output the values name , colour and price of each dictionary in the list

```
In [ ]: fruits = [  
    {'name': 'apple', 'colour': 'red', 'price': 0.12},  
    {'name': 'banana', 'colour': 'yellow', 'price': 0.2},  
    {'name': 'pear', 'colour': 'green', 'price': 0.19},  
]
```

**Extension:** Add more items to the list.



**Random Choice**

The `choice()` function in the `random` module returns a random item from a list

```
In [25]: import random

colours = ['red', 'green', 'blue']
chosen_colour = random.choice(colours)

print(chosen_colour)
```

blue

**Exercise 4.7:** Write a program to create a random name. You should have a list of random firstnames and a list of lastnames. Choose a random item from each and display the result.

Extension: Create a list of verbs and a list of nouns. Using the four lists create randomised sentences eg. Alice Brown codes Python

**Recap**

This session:

1. Lists
2. Dictionaries

**Question 1:** What shape brackets are used for creating a list and what shape brackets are used for creating a dictionary?

**Question 2:** What is the result of this program?

```
In [ ]: cheeses = [  
        'brie',  
        'cheddar',  
        'wensleydale',  
        'edam',  
    ]  
  
    print(cheeses[4])
```

**Question 3:** This program raises an error when I run it. What do I need to change to get it to run?

```
In [ ]: trees = [  
    {'leaf_colour': 'green', 'height': 2120},  
    {'leaf_colour': 'green', 'height': 2300},  
  
    new_tree = {  
        'leaf_colour': 'green',  
        'height': 1020  
    }  
    trees.append(new_tree)  
    print(trees)
```



**Homework:** Session 4 homework questions on the mini-site