Lists

SHARP SIGHT

What you'll learn

- What is a list

- Retrieving data from a list
  - Single items
  - “slices”

- Creating lists

- Changing lists
  - adding, removing items
  - list methods

List basics

What are lists?

- Lists are sequences of values

- Values in the list are called elements (AKA, items)

- Lists can be changed
  - Lists are “mutable”
  - contrast this with tuples, which are immutable

- Lists can contain different data types within a single list
  - e.g., you can have an integer and a string in the same list

How to create a list

- Create lists with a sequence of items, enclosed with brackets

```
list_variable = [ list-value1, list-value2, … ]
```

Inside of the brackets, you have a sequence of items or values that you want the list to contain

Example: creating a list

- Here, we're creating a list of string objects
  - the strings are car names

```
car_list = ['ferrari','porsche','bugatti']
type(car_list)
list
```

List items have indexes

- Each element in the list has an index
  - Indexes start at 0, just like in strings and other sequences

```
car_list = ['ferrari','porsche','bugatti']
```

There are *other* ways to create lists

- You can create lists using the `list()` function
  - `list()` transforms different structures into lists

```
japanese_car_set = {'honda', 'toyota'}


japanese_car_list = list(japanese_car_set)
type(japanese_car_list)
list
```

Why do we use lists?

- Lists *can* change
  - This is in contrast to tuples

- Use lists when you have duplicate items
  - e.g., list of names, where there are two people have the same name

- Use when the order matters
  - Lists are ordered

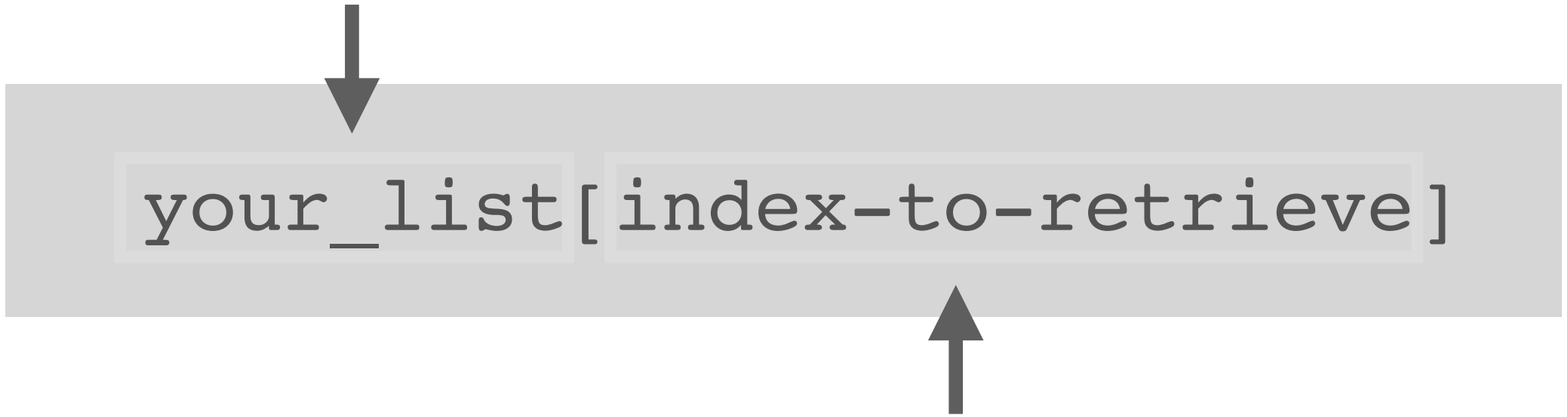
Accessing list items

You can access list items using "bracket" notation

- Access individual items using bracket notation (e.g. `car_list[]`)
  - use the appropriate index

Syntax: how to retrieve items from a list

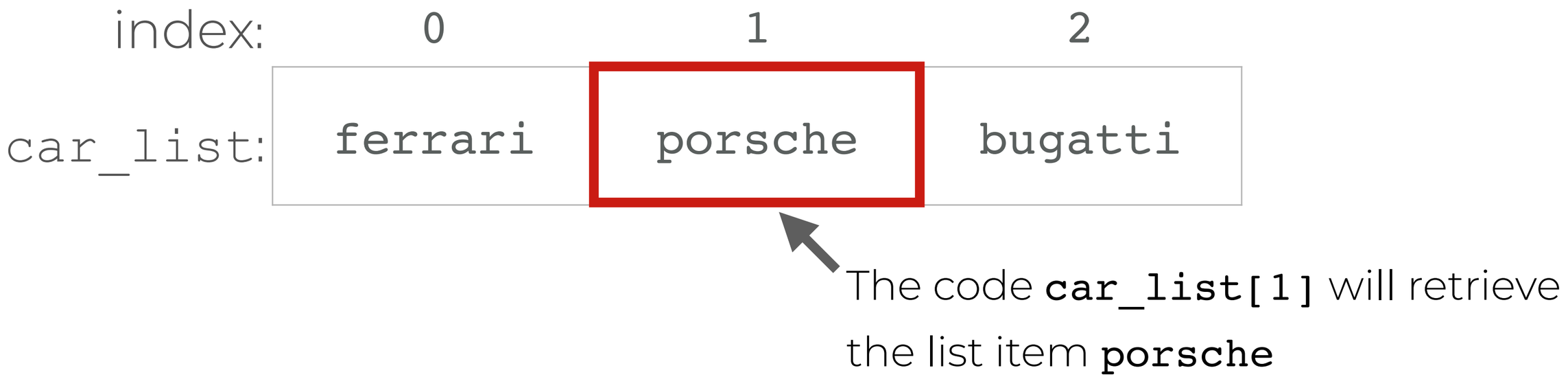
To retrieve an item from a list, type the name of the list, followed by brackets



Inside of the brackets, you have the index associated with the value you want to get

Example: how to retrieve items from a list

```
car_list = ['ferrari','porsche','bugatti']
car_list[1]
```



List "slicing"

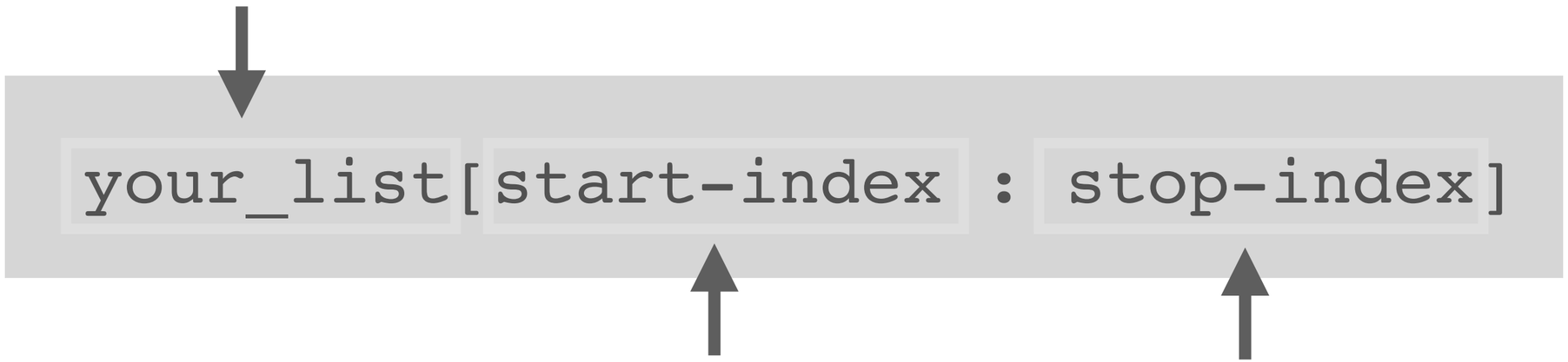
You can also retrieve "slices" of lists with bracket notation

- You can "slice" lists just like you slice other sequences (e.g. `car_list[]`)

- This is very similar to slicing strings, etc

Syntax: how to access a "slice" of a list

The name of
the list

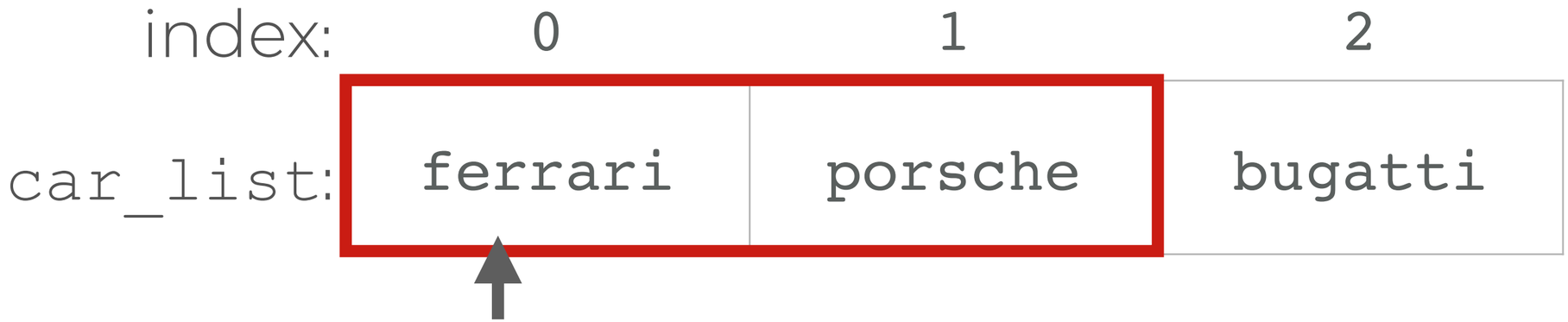


The index associated with
the first list item you want
to retrieve

The index of the "stoping
point" of your slice
... this will not be included!

Example: how to access a slice of a list

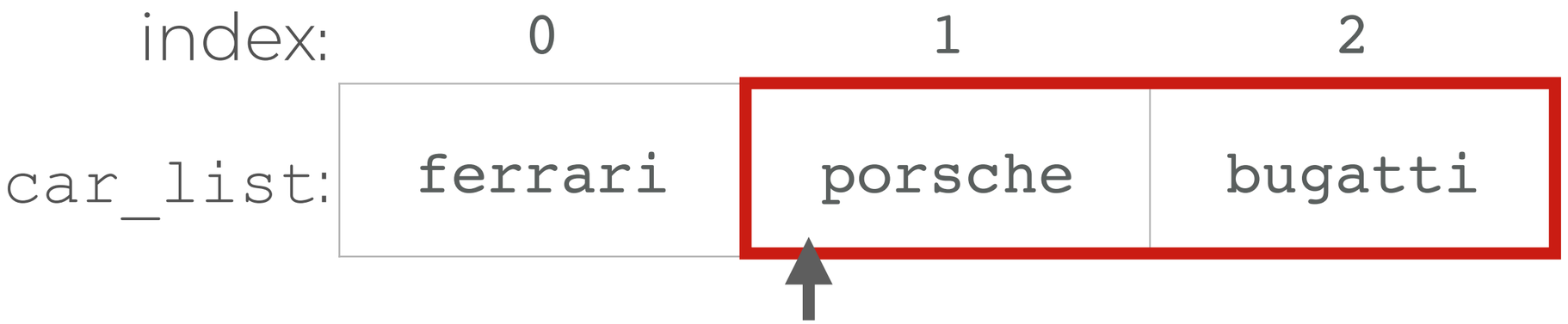
```
car_list = ['ferrari','porsche','bugatti']
car_list[0:2]
```



The code **`car_list[0:2]`** will retrieve the list items **`ferrari`** and **`porsche`**

You can retrieve items from start position to the end of the list

```
car_list = ['ferrari','porsche','bugatti']
car_list[1:]
```



The code **`car_list[1:]`** will retrieve the items from index 1 to the end of the list

List slices work the same as slices for other sequences

- Remember, the stop index is excluded!
  - e.g., `car_list[0:2]` will retrieve the items from 0 up to but *excluding* item 2

- You can use `-1` to reference the last element of the list

- Many of the other techniques you learn about creating slices will also work with lists

List methods

List methods

- Lists also have “list methods” that we can use to perform operations on lists
  - call list methods using “dot notation”
  - e.g. **`your_list.append()`**

method	what it does
`append()`	add an item to end of list
`extend()`	extends a list with a new list of items
`remove()`	removes an item from a list
`sort()`	sorts a list (by value, not by index)
`index()`	returns the index of the first occurrence of the given

* note, this is an *abridged* list of list methods

Adding items to a list

Adding items to a list

- There are two primary ways to add items to a list
  - `append()`
  - `extend()`

- You can also concatenate strings together

`append()` ADDS NEW ELEMENTS TO THE END OF A LIST, ONE ITEM AT A TIME

- Call the method using dot notation

- Specify the new value inside of `append()`

```
car_list = ['ferrari','porsche','bugatti']
car_list.append('hennessey')

car_list
['ferrari', 'porsche', 'bugatti', 'hennessey']
```

`extend()` ADDS MULTIPLE ITEMS TO A LIST

- Call the **`extend()`** method using dot notation
- Specify the new items inside of a list

```
car_list = ['ferrari','porsche','bugatti']
extra_cars = ['mclaren','aston martin']

car_list.extend(extra_cars)
car_list
['ferrari', 'porsche', 'bugatti', 'mclaren', 'aston martin']
```

You can combine existing lists together with the + operator

- This is very similar to string concatenation

```
car_list1 = ['ferrari','porsche','bugatti']
car_list2 = ['mclaren','aston martin']

car_list_full = car_list1 + car_list2
car_list_full
['ferrari', 'porsche', 'bugatti', 'mclaren', 'aston martin']
```

Removing items from a list

Removing items from a list

- There are several ways to remove items from a list
  - `del`
  - `remove()`

- When we remove an item from a list, the items shift index position

Delete item from list using `del`

- The **`del`** statement is one way to remove items
  - indicate which item to remove by referencing its index

```
car_list = ['ferrari','porsche','bugatti']
del car_list[0]

car_list
['porsche', 'bugatti']
```

Remove using `.remove()` method

- We can also use the **`.remove()`** method

- With **`.remove()`** we reference the *item* we want to remove
  - useful if you only know the value, not the index

```
car_list = ['ferrari','porsche','bugatti']
car_list.remove('porsche')

car_list
['ferrari', 'bugatti']
```

Removing items shifts index position

- When we remove an item from a list, the items shift index position

```
car_list = ['ferrari','porsche','bugatti']
```

Removing items shifts index position

- When we remove an item from a list, the items shift index position

```
car_list = ['ferrari','porsche','bugatti']
car_list.remove('ferrari')
```

Recap

Recap of what we learned

- Lists are sequences of elements

- How to create lists
  - bracket notation

- How to retrieve data from a list
  - retrieve single items by index
  - retrieve “slices” with slicing syntax

- How to modify lists
  - Delete with `del` and `.remove()`
  - Add items with `.append()` and `.extend()`