Dictionaries

Simon Robinson
Software Developer

@TechieSimon www.SimonRobinson.com

Types of Collection

Arrays and Lists

Designed for
index-based look-up
and enumerating

Dictionaries

Designed for
finding items

Problem to solve:

Where does the 42 go?

Two Similar Problems

Where does the 42 go?

You can solve both problems with `Array.FindAll()`

This can also be solved with a dictionary!

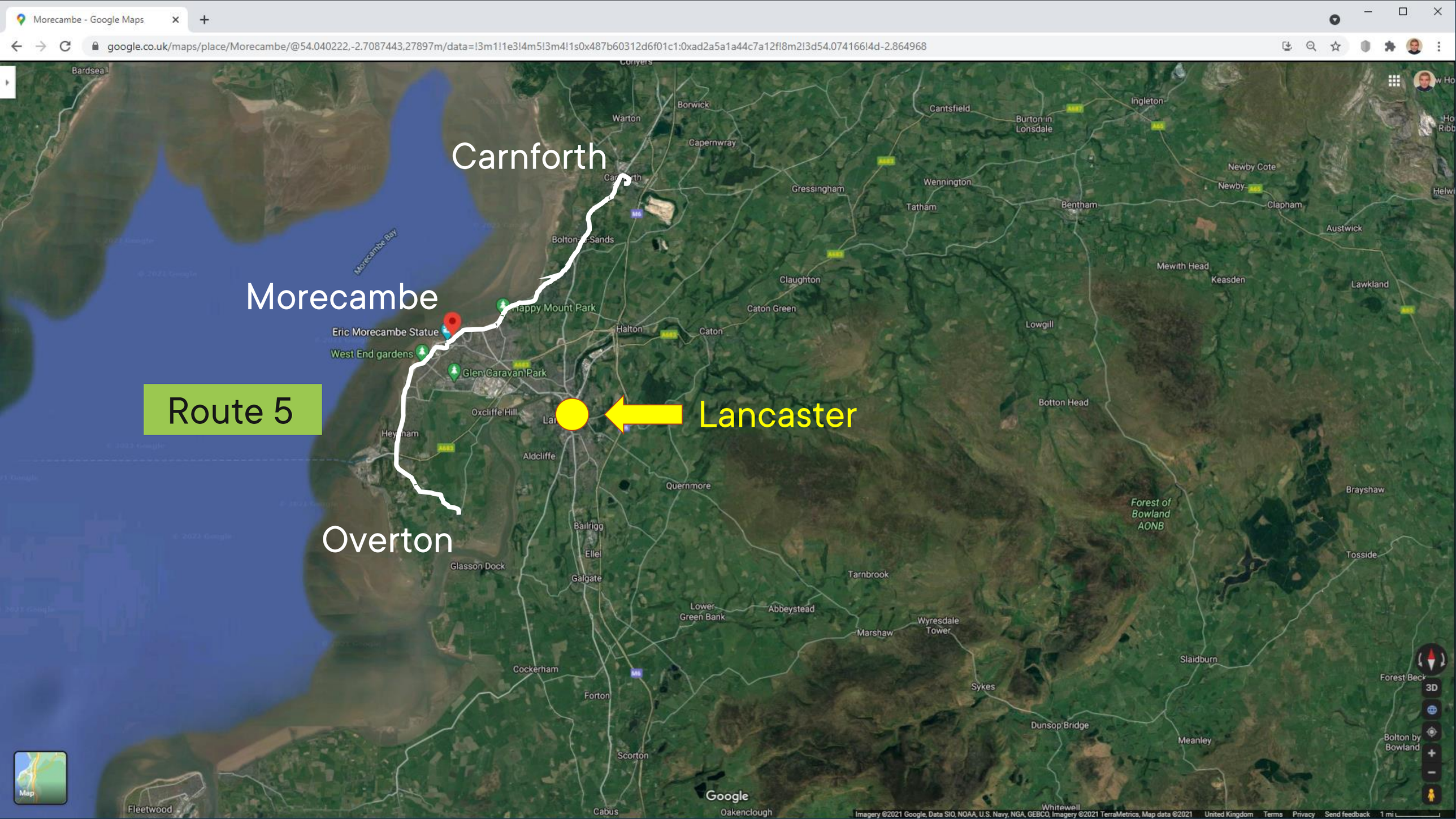
Which bus goes to Blackpool?

Demo

Find route with a given number

- Using an array
- Using a dictionary

Code Demo


Morecambe - Google Maps
google.co.uk/maps/place/Morecambe/@54.040222,-2.7087443,27897m/data=!3m1!1e3!4m5!3m4!1s0x487b60312d6f01c1:0xad2a5a1a44c7a12f!8m2!3d54.074166!4d-2.864968
Carnforth
Morecambe
Eric Morecambe Statue
Happy Mount Park
West End gardens
Glen Caravan Park
Route 5
Lancaster
Overton
Bardsea
Warton
Borwick
Capernwray
Cantsfield
Burton in Lonsdale
Ingleton
Gressingham
Wennington
Tatham
Bentham
Newby Cote
Newby
Clapham
Austwick
Bolton-le-Sands
Morecambe Bay
Claughton
Caton Green
Halton
Caton
Lowgill
Mewith Head
Keasden
Lawkland
Botton Head
Oxcliffe Hill
Aldcliffe
Quernmore
Brayshaw
Forest of Bowland AONB
Bailrigg
Ellel
Glasson Dock
Galgate
Tarnbrook
Tosside
Lower Green Bank
Abbeystead
Wyresdale Tower
Marshaw
Slaidburn
Cockerham
Forton
Sykes
Dunsop Bridge
Meanley
Bolton by Bowland
Forest Beck
Scorton
Fleetwood
Cabus
Oakenclough
Whitewell
Google
Map
3D
Imagery ©2021 Google, Data SIO, NOAA, U.S. Navy, NGA, GEBCO, Imagery ©2021 TerraMetrics, Map data ©2021
United Kingdom
Terms
Privacy
Send feedback
1 mi

Code Demo

Route Numbers

The route number identifies the route

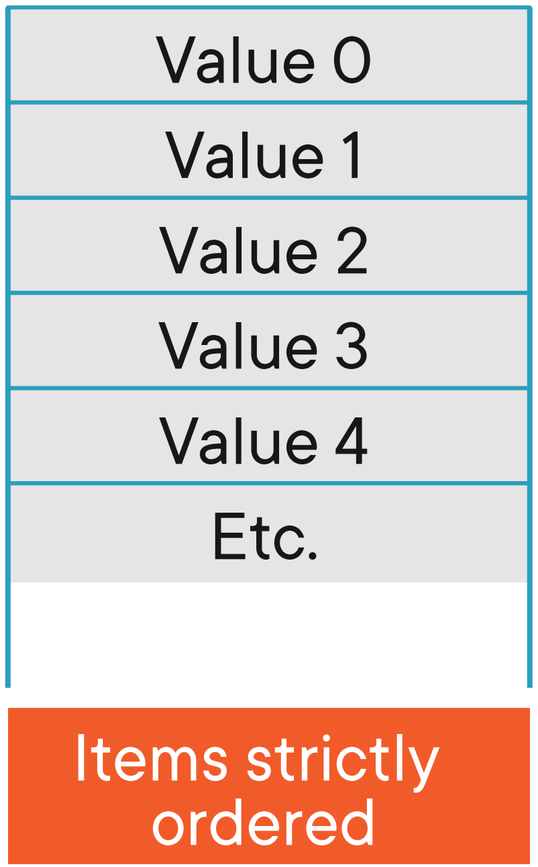
It’s unique (at least for our data)

It’s an ID

Route number is the **key**

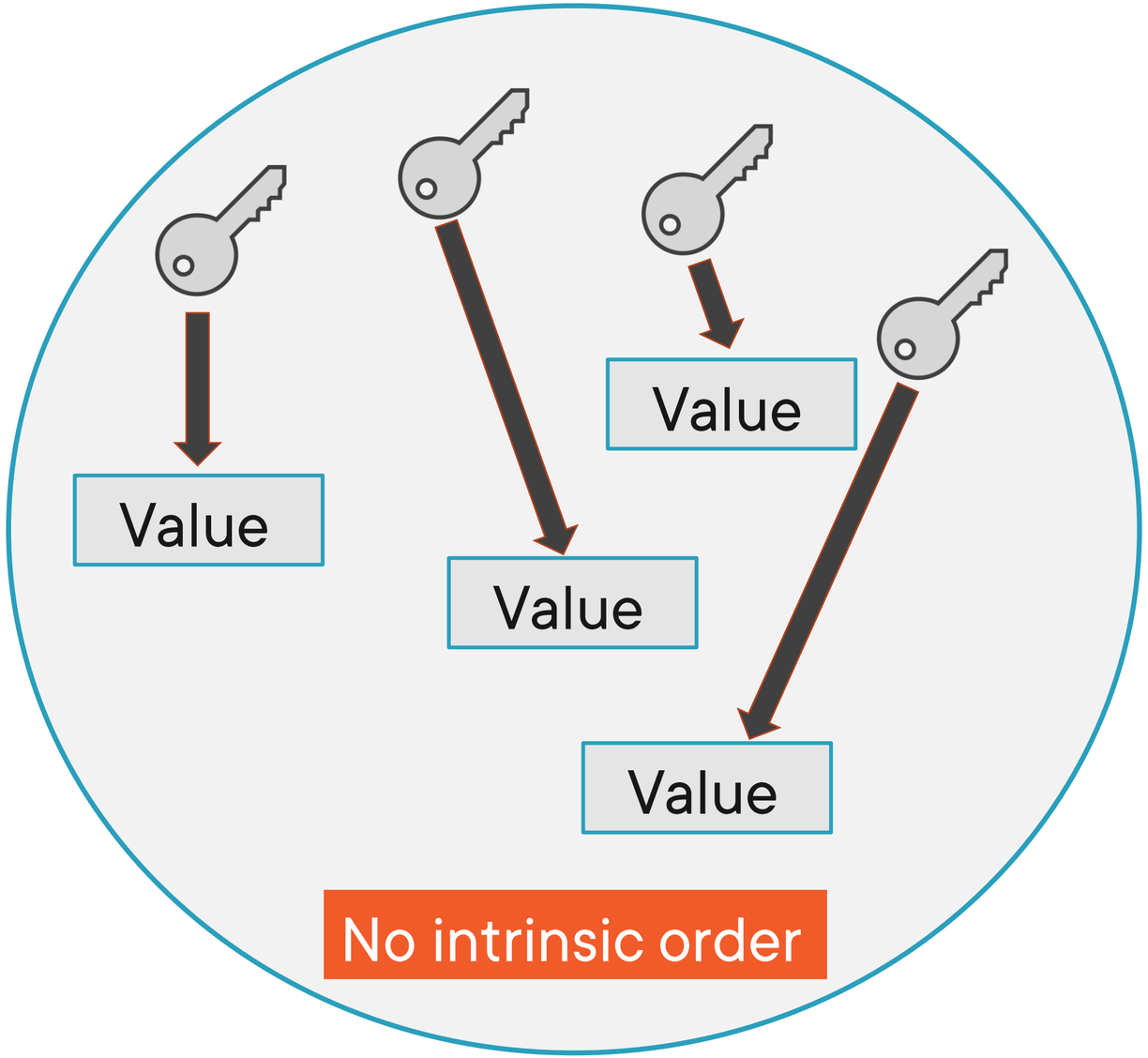
Route is the **value**

Array and list:



Items strictly ordered

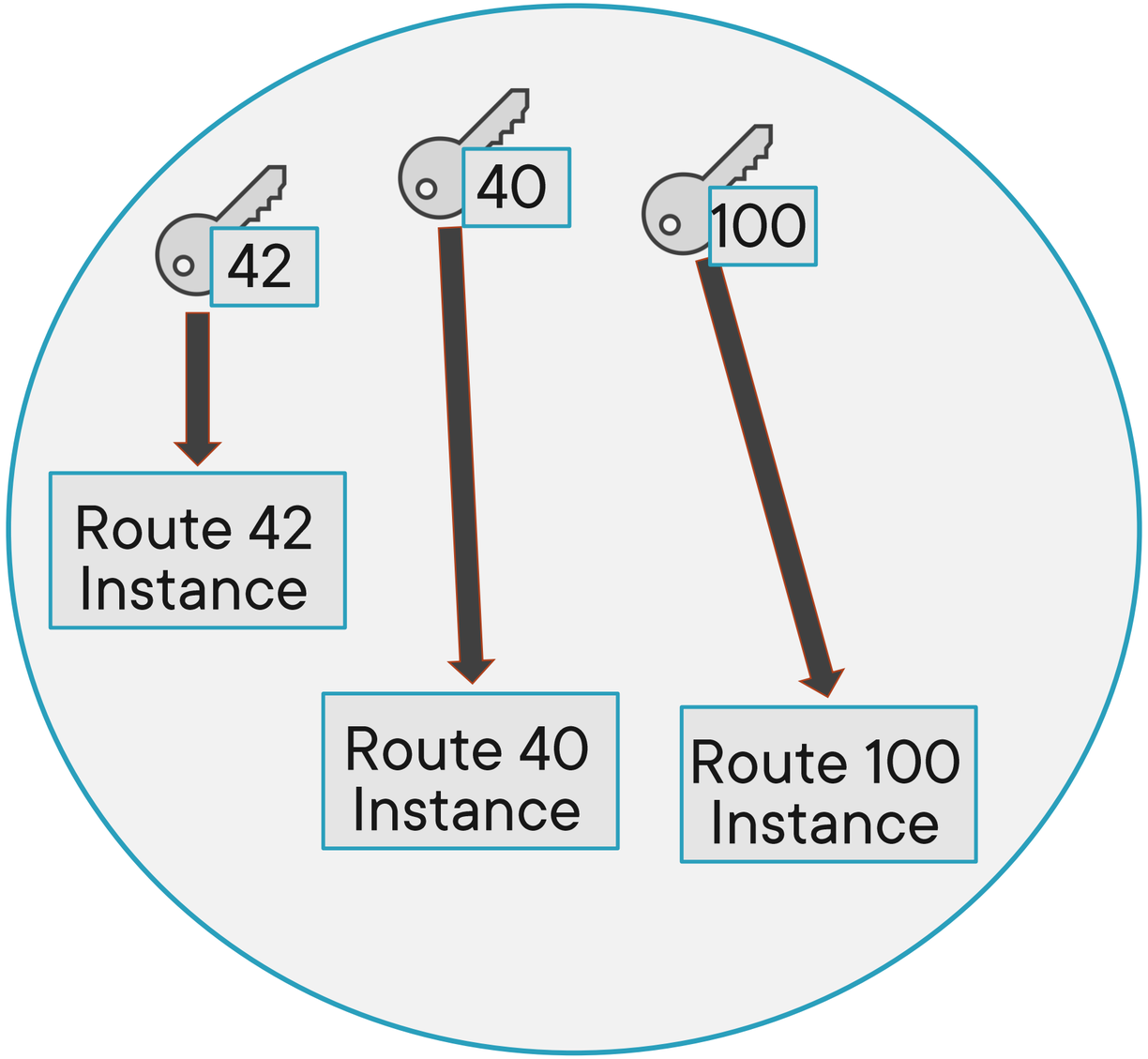
This is a Dictionary!



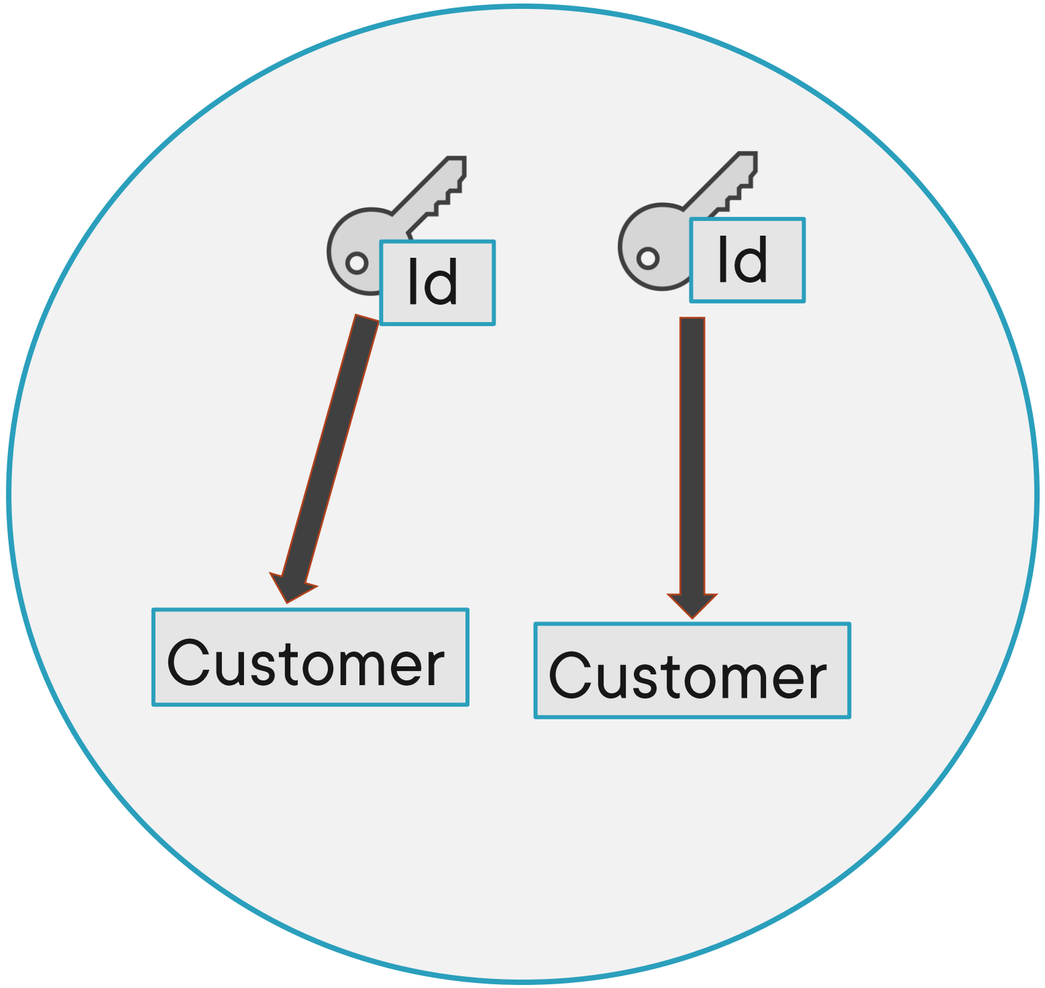
Dictionary of Bus Routes

Here the keys are integers

But the key can be any type



Example: Dictionary of Customers




```
public class Customer
{
    public string Id { get; }
    // etc.
}
```

In this case
the key would probably be a string

Code Demo

ContainsKey() vs. TryGetValue()

This only looks up the value once

```
bool success = allRoutes.TryGetValue(number, out BusRoute answer);
if (success)
{
    // etc.
```

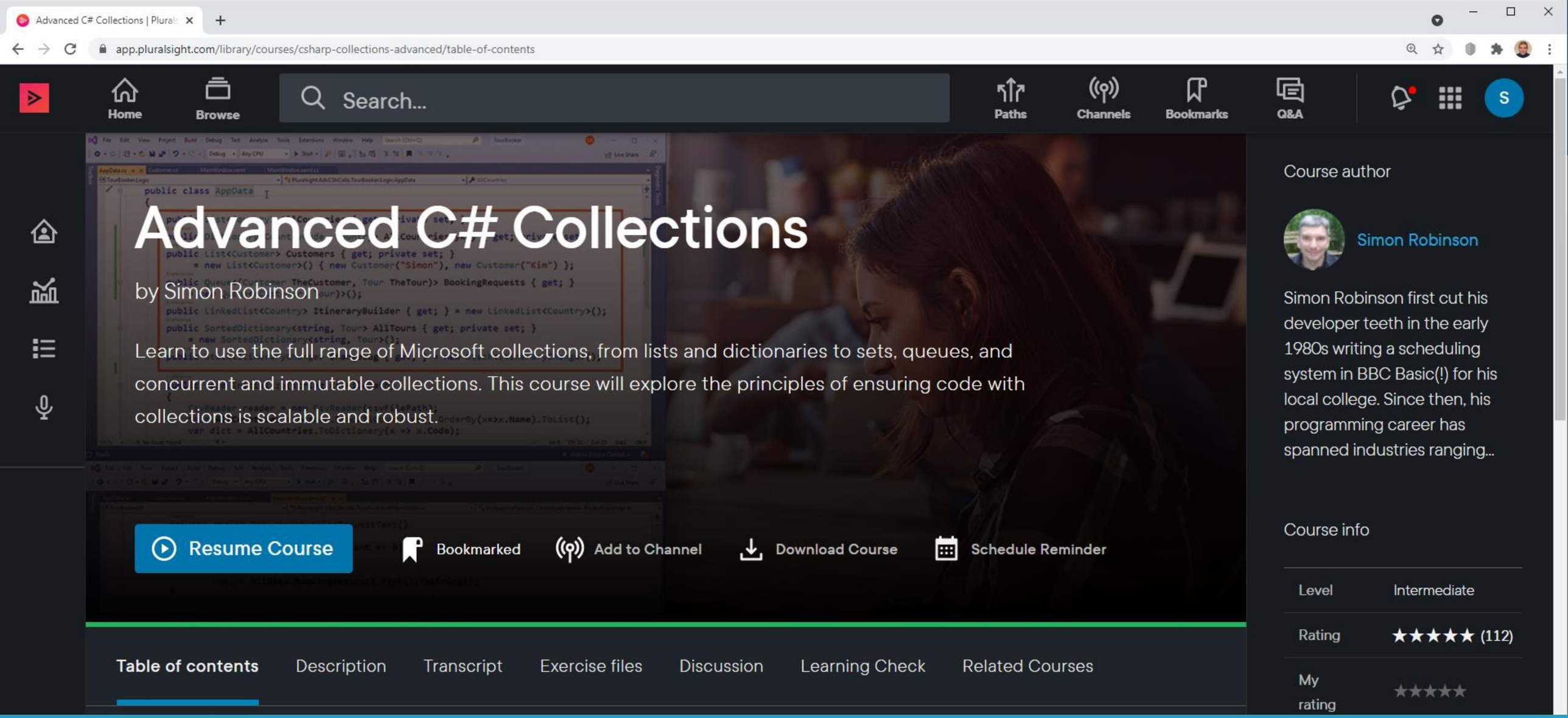
This looks up the value twice

```
bool success = allRoutes.ContainsKey(number);
if (success)
{
    BusRoute answer = allRoutes[number];
    // etc.
```

Code Demo

Dictionary or SortedDictionary

`Dictionary<TKey, TValue>`	`SortedDictionary<TKey, TValue>`
Doesn't sort	Automatically sorts by key
Key can be any type	Key can only be types that the dictionary knows how to sort
	Numbers and strings are fine
	Most other types require extra work to specify how to sort



This covers sorting in sorted dictionaries

Use an ordinary dictionary,
unless you really need sorting.

Version
.NET 5
Search
IStructuralComparable
IStructuralEquatable
Queue
ReadOnlyCollectionBase
SortedList
SortedList
Constructors
Properties
Methods
Explicit Interface Implementations
Stack
StructuralComparisons

SortedList Class

Namespace: System.Collections

Assembly: System.Collections.NonGeneric.dll

Represents a collection of key/value pairs that are sorted by the keys and are accessible by key and by index.

C# Copy

```
public class SortedList : ICloneable, System.Collections.IDictionary
```

Inheritance Object → SortedList

Implements ICollection , IDictionary , IEnumerable , ICloneable

Examples

The following code example shows how to create and initialize a SortedList object and how to print out its keys and values.

Is this page helpful?
Yes No
In this article
Definition
Examples
Remarks
Constructors
Properties
Methods
Explicit Interface Implementations
Extension Methods
Applies to
Thread Safety
See also

Code Demo

Summary

Dictionaries

- Used for looking up items with a key
- Use `ContainsKey()` or `TryGetValue()` if you don't know whether a key exists
- `SortedDictionary` to enumerate if order is important
  - Provided it knows how to sort the keys

Up Next:
Sets and HashSet<T>