.NET Framework

# Data structures:

## List

* Instantiation requires datatype value
* Iterable
* List of single values of the same datatype

## Dictionary

* Instantiation requires datatype values
* Iterable
* List of two paired values
* The datatype of the first value is the same for each entry
* The datatype of the second value is the same for each entry

## Array List

* Instantiation does not require datatype value
* Not iterable
* List of single values
* Values are boxed into Objects

## Hash table

* Instantiation does not require datatype value
* Not iterable
* List of paired values
* Values are boxed into Objects

## Concurrent Dictionary

* Dictionary that must be used if threading is involved

## Bit Array

* Values are bits and are set by Booleans
* Instantiation requires size

## Tuple

* N-paired values
* Each value in an entry has a specified datatype
* Iterable
* Like dictionary with n values

## Stack

* Last in first out order
* Instantiation requires datatype value
* Iterable

## Queue

* First in first out
* Instantiation requires datatype
* Iterable

## Hash set

* No duplicates
* Instantiation requires datatype
* Overlap – see common values between two hash sets

# Asynchronous Programming

Task.Run() creates new thread

Useful for web stuff

Delegate – a function that works asynchronously

“Delegates enable you to call a synchronous method in an asynchronous manner.”

Callback – a function that the delegate calls upon to report exit or state

# Multithreading

It’s a thing

# Null Coaslescing

var default = in1 ?? “default”;

Set default value for a variable given a null input

int? -> nullable int

# Null Conditional Operators

if (obj != null)

Console.WriteLine(obj.Name);

same as

Console.WriteLine(obj?.Name);

## Nesting

You can nest this.

Console.WriteLine(obj?.Name?.Length);

# String Interpolation

var world = “world”;

var s1 = “hello” + world;

var s2 = $”hello {world}”;

# String Concatenation

Console.WriteLine(“Hello “ + planet.Name + “! Have a great “ + dayOfWeek + “!”);

Each time we concatenate strings, a new string object is created.

# String Formatting

Console.WriteLine(“Hello {0}! Have a great {1}!”, planet.Name, dayOfWeek);

Does not create wasteful data like concatenation.