Basic C# data structures cheat sheet

Lists
- Multiple items of the **same type**.
- Number of items **can grow and shrink**.
- Items are accessed with the **bracket notation [ ]**.

int	same type
[0]	"Dave"
[1]	"Bernard"
[2]	"Razor"
[3]	"Wendy"
…	…

```
var names = new List<string>();
```

Arrays
- Multiple items of the **same type**.
- Number of items is **fixed when instantiated**.
- Can have **multiple dimensions**.
- Items are accessed with the **bracket notation with commas [,]**.

int	same type
[0]	3.14159
[1]	2.71828
[2]	299.7e6
[3]	1.61803

int	[, 0]	[, 1]	[, 2]	[, 3]
[0,]	1	0	0	14
[1,]	0	1	0	23
[2,]	0	0	1	42
	same type			

```
var numbers = new float[4];

var matrix = new int[3, 4];
```

Jagged arrays
- Multiple items of the **same type**.
- Number of items is **fixed when instantiated**.
- Has **multiple dimensions**.
- Sub-arrays can have **different lengths**.
- Items are accessed with the **bracket notation repeated [ ][ ]**.

int	[ ][0]	[ ][1]	[ ][2]	[ ][3]	[ ][4]	[ ][5]
[0][ ]	'P'	'I'	'X'	'E'	'L'	'S'
[1][ ]	'A'	'R'	'E'			
[2][ ]	'G'	'R'	'E'	'A'	'T'	
	same type					

```
var letters = new char[3][];
letters[0] = new char[6];
letters[1] = new char[3];
letters[2] = new char[5];
```

Classes
- Multiple items of **different types**.
- Number of items is **fixed when defined**.
- Items are accessed with the **dot notation a.b**.

field name	different types
.name	"Lara"
.age	25
.height	1.68f
.location	Country.England

```
public class Person {
  public string name;
  public int age;
  public float height;
  public Country location;
}

var lara = new Person();
```

Dictionaries
- Multiple items of the **same type**.
- Number of items **can grow and shrink**.
- Items are accessed with the **bracket notation [ ]**.
- Items are accessed with **keys of defined type**.

same type key	same type value
["pi"]	3.14159
["e"]	2.71828
["speed of light"]	299.7e6
["golden ratio"]	1.61803
…	…

```
var constants = new Dictionary<string, float>();
```