Ruby For Programmers

This cheat-sheet accompanies the Bitwise Courses course on *Ruby For Programmers* by Huw Collingbourne.

Arrays

An Array is a sequential collection of items in which each item can be indexed. In Ruby, (unlike many other languages) a single Array can hold items of mixed data types such as strings, integers and floats or even a method-call which returns some value:

```
a1 = [1,'two', 3.0, array_length( a0 ) ]
```

The first item in an array has the index 0, which means that the final item has an index equal to the total number of items in the array minus 1. Given the array `a1` shown above, this is how to obtain the values of the first and last items:

```
a1[0]           # returns 1st item (at index 0)
a1[3]           # returns 4th item (at index 3)
```

Multi-dimensional arrays can be created by nesting arrays inside another array, like this:

```
a = [[1,2,3,4],
    [5,6,7,8],
    [9,10,11,12],
    [13,14,15,16]]
```

Hashes

A Ruby Hash is the equivalent of what some other languages call an associative-array or a ‘Dictionary’. Just like a real dictionary, the entries are indexed by some unique *key* (in a dictionary, this would be a word) and a *value* (in a dictionary, this would be the definition of the word).

A hash-literal can be defined by placing a comma-delimited list of key-value pairs beytween curly brackets, with each key associated with a value like this:

```
h1 = {    'room1'=>'The Treasure Room',
    'room2'=>'The Throne Room',
    'loc1'=>'A Forest Glade',
    'loc2'=>'A Mountain Stream' }
```

Note that each key must be unique. Values may, however, be duplicated.

Indexing Into A Hash

Hashes, unlike arrays, should not be assumed to have any sequential ‘order’. To access a value, place its key between square brackets like this:

```
puts(h1['room2'])
```

This displays:

```
The Throne Room’
```