Arrays and Hashes

León Jaramillo

softserve

Why to use arrays and hashes?

- Variables are suitable and useful to store data in any program.
- But, since the beginning of this job, we found the need to store grouped data.
- Variables might not be that suitable when:
    - We need to represent natural (or artificial) groups of entities.
    - We need to store large datasets of similar or dissimilar information.
    - We need to process data with mechanisms such as iteration.
- Hence, Ruby includes data structures such as arrays and hashes.

Arrays

- Arrays are ordered collections of any object. They are indexed (we can get any element using its index).

- Indexes in Ruby arrays start at 0 relatively to the beginning of the array and at -1 relatively to the its end (as lists in Python).

- Unlike in languages such as Java, Ruby arrays are dynamic (grow its size while we add elements to them).

- They are similar to Java arrays (but dynamic) and to Python lists.

How to use them?

- Creating an empty array

```
        my_array = Array.new
```

- Creating an array of any size

```
        my_array = Array.new(10)
```

- Creating an array of any size with a given value for each element

```
        my_array = Array.new(5, "One message")
```

- Creating an array with an initial set of elements

```
        my_array = ["Hugo", "Paco", "Luis", 27, 50]
```

- Assigning a value to an array’s given element

```
        my_array[3] = "Another message"
```

- Getting the value of a given array’s element

```
        puts my_array[2]
```

- Getting the length of an array

```
        my_array.length
```

Hashes

- A Hash is a collection of key-value pairs (i.e., “Antioquia” => “Medellin”). Unlike in arrays, in hashes the key can be an object of any type.

- Hashes and unordered. So, when we iterate them, it’s following an arbitrary order.

- If you try to get an element using a nonexciting key, it returns *nil*.

- They are similar to Java hashtables and to Python dictionaries.

Hashes

- Creating an empty hash

```
my_hash = Hash.new
```

- Creating a hash with an initial set of elements

```
my_hash = {"Juan Pérez" => 8, "José Rodríguez" => 3 }
```

- Adding or modifying an elements with a given key

```
my_hash["Juan Pérez"] = 10
```

- Getting the value of a given hash’s element

```
 puts my_hash["José Rodríguez"]
```

- Getting the number of key-value pairs in a hash

```
my_hash.size
```

Useful Resources

- Official Array documentation: https://ruby-doc.org/core-3.1.0/Array.html
- Official Hash documentation: https://ruby-doc.org/core-3.1.0/Hash.html

softserve

Homework

- What cases do you think arrays are more suitable than hashes for?
- What cases do you think hashes are more suitable than arrays for?
- Can we have hashes as arrays' elements? Can we have arrays as hashes' elements? Can an array or a hash be the key of any hash element?
- Write an array that stores, in order, the most popular domains in Colombia, according to https://radar.cloudflare.com/co
- Write a hash that stores Colombia's departments and their corresponding capital cities.
- Write a hash that stores Colombia's autonomous systems' codes and their corresponding names, according to https://radar.cloudflare.com/co

COLOMBIA DC

Thanks! Any question?

softserve