Ruby 3 and destruction with hash and array

In Ruby 3 we have rightward assignment operator `=>`. This flips the script and lets you write an expression before assigning it to a variable. Instead of `x = :y` we can write `:x = y`.

```
 => {:foo=>1, :bar=>2}
3.1.0 :009 > hash => { foo:, bar: }
 => nil
3.1.0 :010 > foo
 => 1
3.1.0 :011 > bar
 => 2
3.1.0 :012 >
```

What happen if we have a **rest?

```
3.1.0 :012 > hash = { foo: 1, bar: 2, more: 3, andmore: 4 }
 => {:foo=>1, :bar=>2, :more=>3, :andmore=>4}
3.1.0 :013 > hash => { foo:, bar:, **rest }
 => nil
3.1.0 :014 > foo
 => 1
3.1.0 :015 > bar
 => 2
3.1.0 :016 > rest
 => {:more=>3, :andmore=>4}
3.1.0 :017 >
```

Rightward assignment and pattern matching works with arrays as well:

```
3.1.0 :017 > array = [1, 2, 3, 4, 5, 6]
 => [1, 2, 3, 4, 5, 6]
3.1.0 :018 > array => [a, b, *rest]
 => nil
3.1.0 :019 > a
 => 1
3.1.0 :020 > b
 => 2
3.1.0 :021 > rest
 => [3, 4, 5, 6]
3.1.0 :022 >
```

and check this out:

```
3.1.0 :022 > array = ['a', 'b', 1, 2, 3, 'c', 'd']
 => ["a", "b", 1, 2, 3, "c", "d"]
3.1.0 :023 > array => [*left, 1, 2, 3, *right]
 => nil
3.1.0 :024 > left
 => ["a", "b"]
3.1.0 :025 > 1
 => 1
3.1.0 :026 > 2
 => 2
3.1.0 :027 > 3
 => 3
3.1.0 :028 > right
 => ["c", "d"]
```

or this one:

```
3.1.0 :033 > array = [1, 2, 'this is a string', 3, 4]
 => [1, 2, "this is a string", 3, 4]
3.1.0 :034 > array => [*left, String => my_string, *right]
 => nil
3.1.0 :035 > left
 => [1, 2]
3.1.0 :036 > my_string
 => "this is a string"
3.1.0 :037 > right
 => [3, 4]
3.1.0 :038 >
```

and now lets find a specific integer:

```
3.1.0 :045 > int = 2
 => 2
3.1.0 :046 > array = [-2, -1, 1, 2, 3, 4, 5]
 => [-2, -1, 1, 2, 3, 4, 5]
3.1.0 :047 > array => [*left, ^int, *right]
 => nil
3.1.0 :048 > left
 => [-2, -1, 1]
3.1.0 :049 > int
 => 2
3.1.0 :050 > right
 => [3, 4, 5]
3.1.0 :051 >
```