2012年5月26日

Ruby初級者向けレッスン 42回
— Array と Hash —

ひがきまさる@Ruby関西

もくじ

- Array とは
  - Array オブジェクトの作り方
- Hash とは
  - Hash オブジェクトの作り方
- Array の初期化・Hash のデフォルト値
- 繰り返し
  - 繰り返しと多重代入
- Array のコピー

Array とは

- 配列クラス
- 任意のオブジェクトを持つことができる

```
[1, 1, 2, 3]
```

```
[1, "two", [3, "3"], 4.0, :five]
```

Array とは (2)

```
a = [1, "two", [3, "3"], 4.0, :five]


a[0]        # => 1
a[-1]       # => :five
a[1] = "2nd"
a[3, 2]     # => [4.0, :five]
a[1..-2]    # => ["2nd", [3, "3"], 4.0]
a[5]        # => nil
```

Array オブジェクトの作り方

```
["a", "b", "c"] # => ["a", "b", "c"]
("a".."c").to_a # => ["a", "b", "c"]
[*"a".."c"]     # => ["a", "b", "c"]
%w[a b c]       # => ["a", "b", "c"]
```

Array オブジェクトの作り方 (2)

```
"No Ruby, No Life.".scan(/\w+/)
      # => ["No", "Ruby", "No", "Life"]

"1,1,2,3,5,8".split(/,/)
      # => ["1", "1", "2", "3", "5", "8"]
```

Hash とは

- 連想配列クラス
- 任意のオブジェクトを持つことができる
- 任意のオブジェクトをキーにできる

```
{:AAPL=>566.71, :GOOG=>605.23}


{AAPL: 566.71, GOOG: 605.23}
     # => {:AAPL=>566.71, :GOOG=>605.23}
```

Hash とは (2)

```
h = {AAPL: 566.71, GOOG: 605.23}


h[:AAPL]       # => 566.71
h[:MSFT] = 31.16
h[:FB]         # => nil
```

Hash オブジェクトの作り方

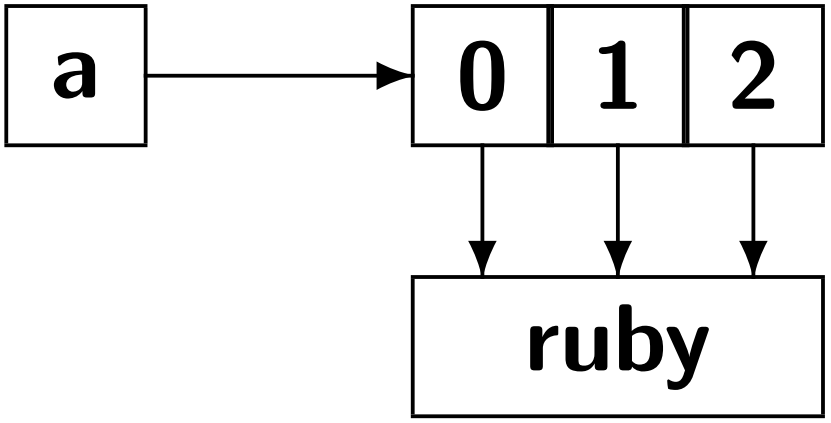
```
a = [:AAPL, 566.71, :GOOG, 605.23]

Hash[*a]
     # => {:AAPL=>566.71, :GOOG=>605.23}
```

Array の初期化

```
Array.new(4, 0)      # => [0, 0, 0, 0]

a = Array.new(3, "ruby")
a    # => ["ruby", "ruby", "ruby"]

a[0].upcase!     # => "RUBY"

a    # => ["RUBY", "RUBY", "RUBY"]
```

Array の初期化

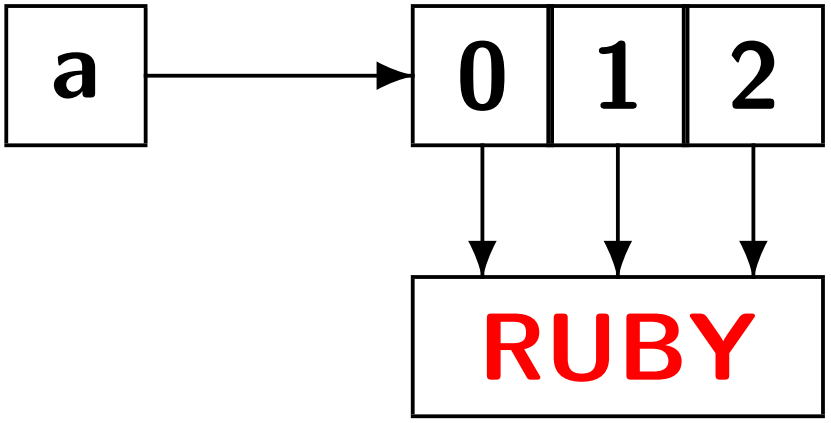
```
a = Array.new(3, "ruby")
```




```
a[0].upcase!
```

Array の初期化

```
a = Array.new(3, "ruby")
```




```
a[0].upcase!
```

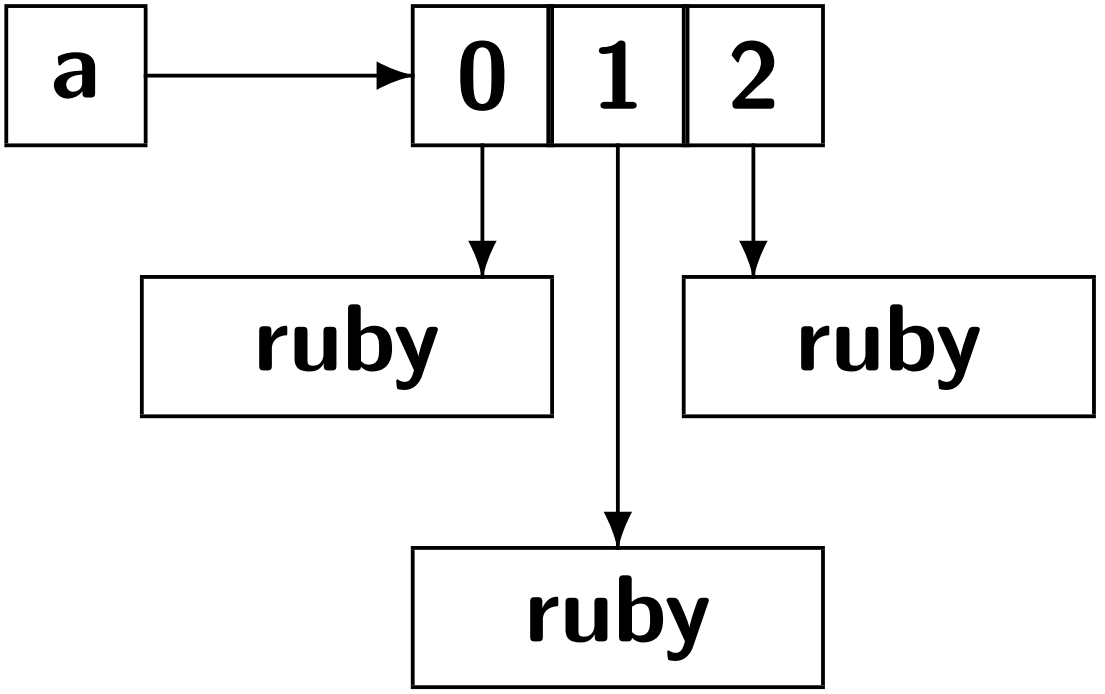
Array の初期化 (2)

```
a = Array.new(3){"ruby"}
a   # => ["ruby", "ruby", "ruby"]

a[0].upcase!    # => "RUBY"
a   # => ["RUBY", "ruby", "ruby"]
```

Array の初期化 (2)

```
a = Array.new(3){"ruby"}
```

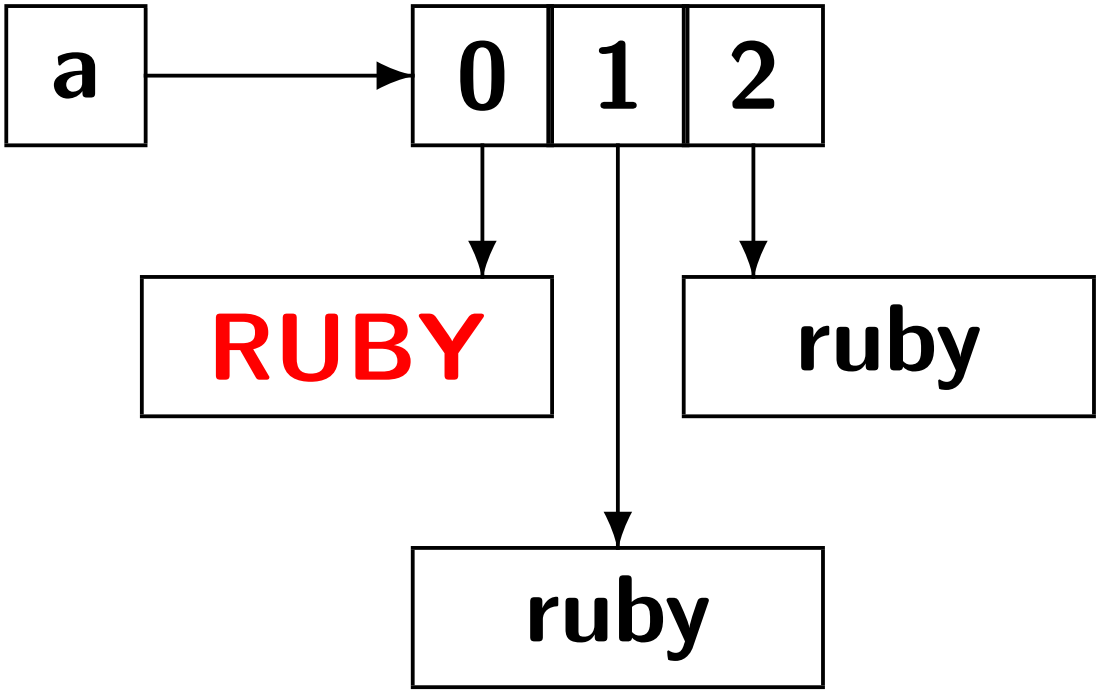
```
a[0].upcase!
```



Array の初期化 (2)

```
a = Array.new(3){"ruby"}
```

```
a[0].upcase!
```



Hash のデフォルト値

```
hash = Hash.new(0.0)   # => {}
hash[:AAPL]            # => 0.0

hash = Hash.new{|h, k| h[k] = ""}
hash                   # => {}
hash[:GOOG]            # => ""
hash[:IBM]             # => ""
```

※ キーの破壊

繰り返し each

```
[0, 1, 2].each{|i| puts i}

[0, 1, 2].each do |i|
  puts i
end

# >> 0
# >> 1
# >> 2
```

繰り返し　　Enumerable

```
Array.ancestors
    # => [Array, Enumerable, Object, Kerne
Hash.ancestors
    # => [Hash, Enumerable, Object, Kernel
```

- **Enumerable**
  - 繰り返しを行なうクラスのための Mix-in
  - クラスには each メソッドが必要

繰り返し　Enumerable　(2)

```
a = [2, 3, 5, 7]  # => [2, 3, 5, 7]

a.map{|i| i * i}  # => [4, 9, 25, 49]
a.select{|i| i.even?}   # => [2]

a.inject{|s, i| s += i} # => 17

a.all?{|n| n.prime?}    # => true
```

繰り返しと多重代入

```
a = [[:matz, 47], [:dhh, 32]]

a.each{|i| puts "#{i[0]}(#{i[1]})"}

a.each{|name, age|puts "#{name}(#{age})"}

# >> matz(47)
# >> dhh(32)
```

繰り返しと多重代入 (2)

```
a = [[1, [:matz, 47]], [2, [:dhh, 32]]]
a.size       # => 2

a.each do |id, (name, age)|
  puts "#{id}: #{name}(#{age})"
end

# >> 1: matz(47)
# >> 2: dhh(32)
```

繰り返しと多重代入 (3)

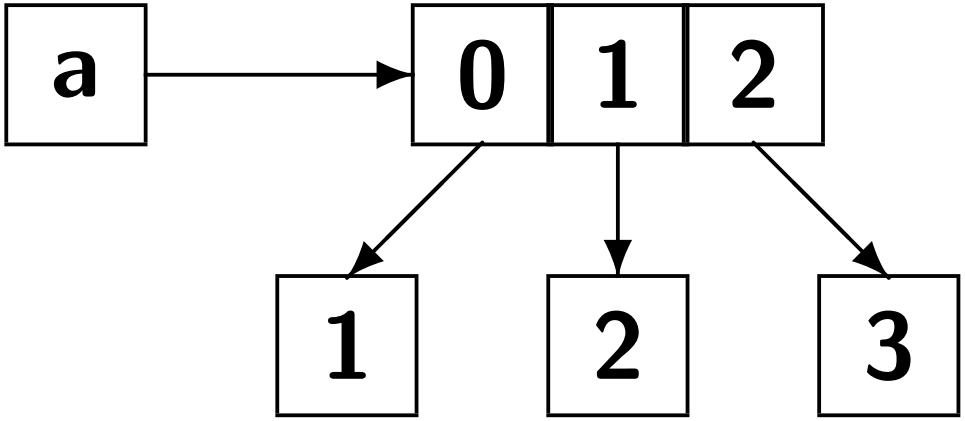
```
h = {matz: 47, dhh: 32}

h.each{|i| puts "#{i[0]}(#{i[1]})"}

h.each{|name, age|puts "#{name}(#{age})"}

# >> matz(47)
# >> dhh(32)
```

Array のコピー

```
a = [1, 2, 3]

b = a

a[0] = 0

a    # => [0, 2, 3]
b    # => [0, 2, 3]
```

Array のコピー

```
a = [1, 2, 3]
```

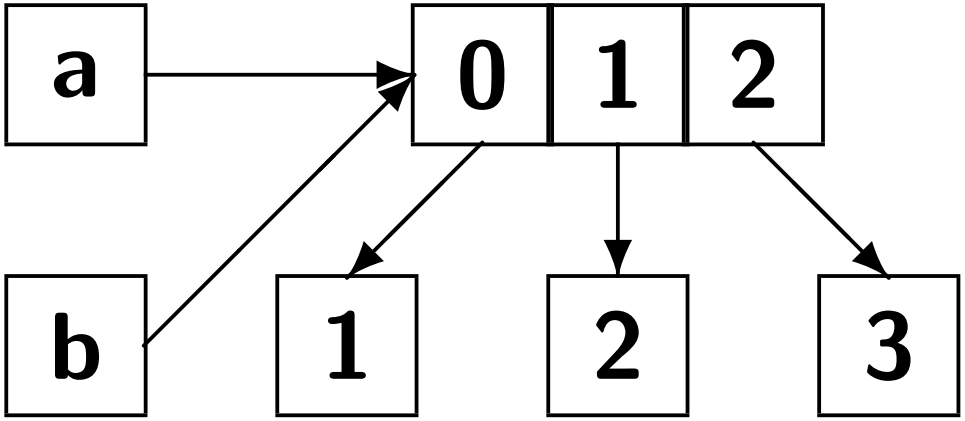



```
b = a
```

```
a[0] = 0
```

Array のコピー

```
a = [1, 2, 3]
```

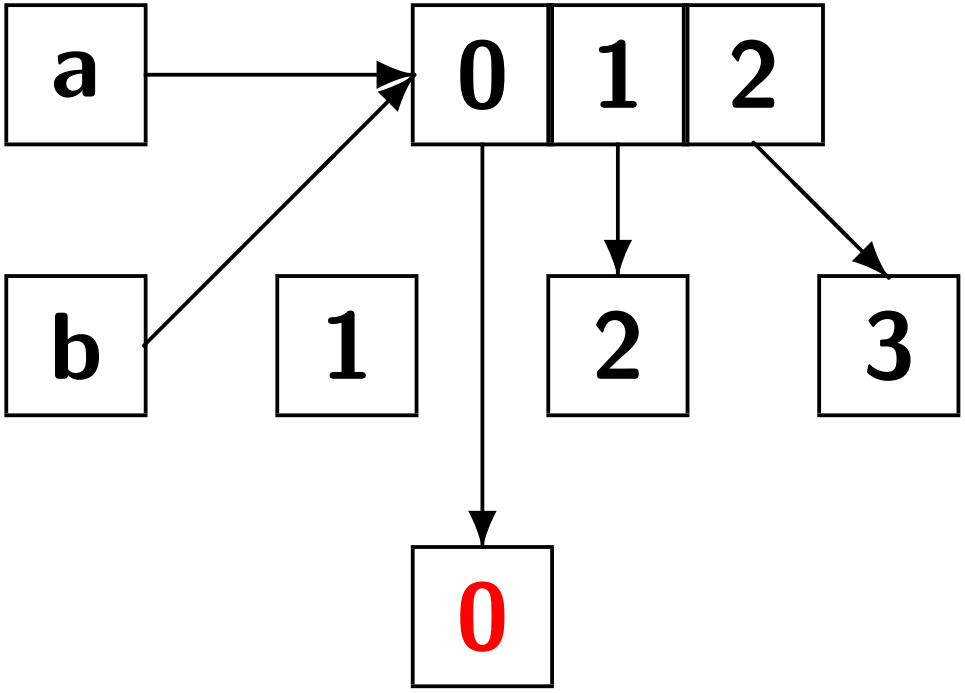



```
b = a
```

```
a[0] = 0
```

Array のコピー

```
a = [1, 2, 3]

b = a

a[0] = 0
```



Array のコピー (2)

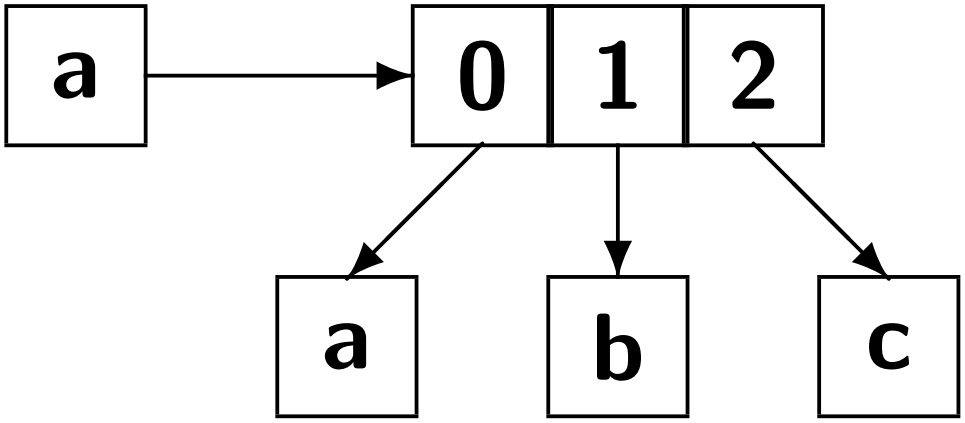
```
a = ["a", "b", "c"]

b = a.dup

a[0] = "A"

a    # => ["A", "b", "c"]
b    # => ["a", "b", "c"]
```

Array のコピー (3)

```
a = ["a", "b", "c"]

b = a.dup

a[1].upcase!

a    # => ["a", "B", "c"]
b    # => ["a", "B", "c"]
```

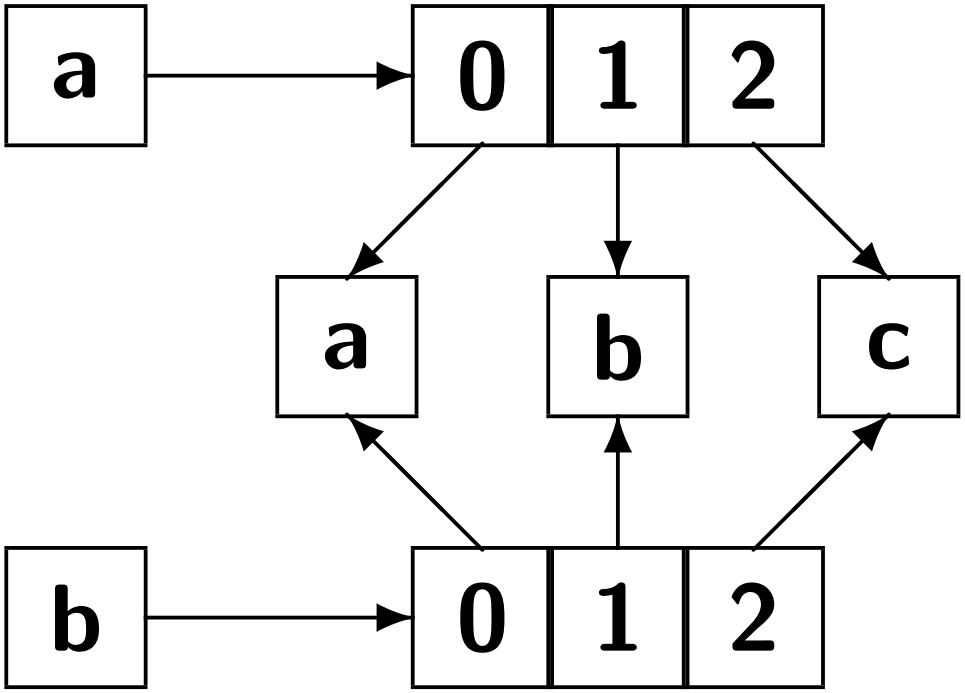
Array のコピー (2) (3)

```
a = ["a", "b", "c"]
b = a.dup
a[0] = "A"
a[1].upcase!
```



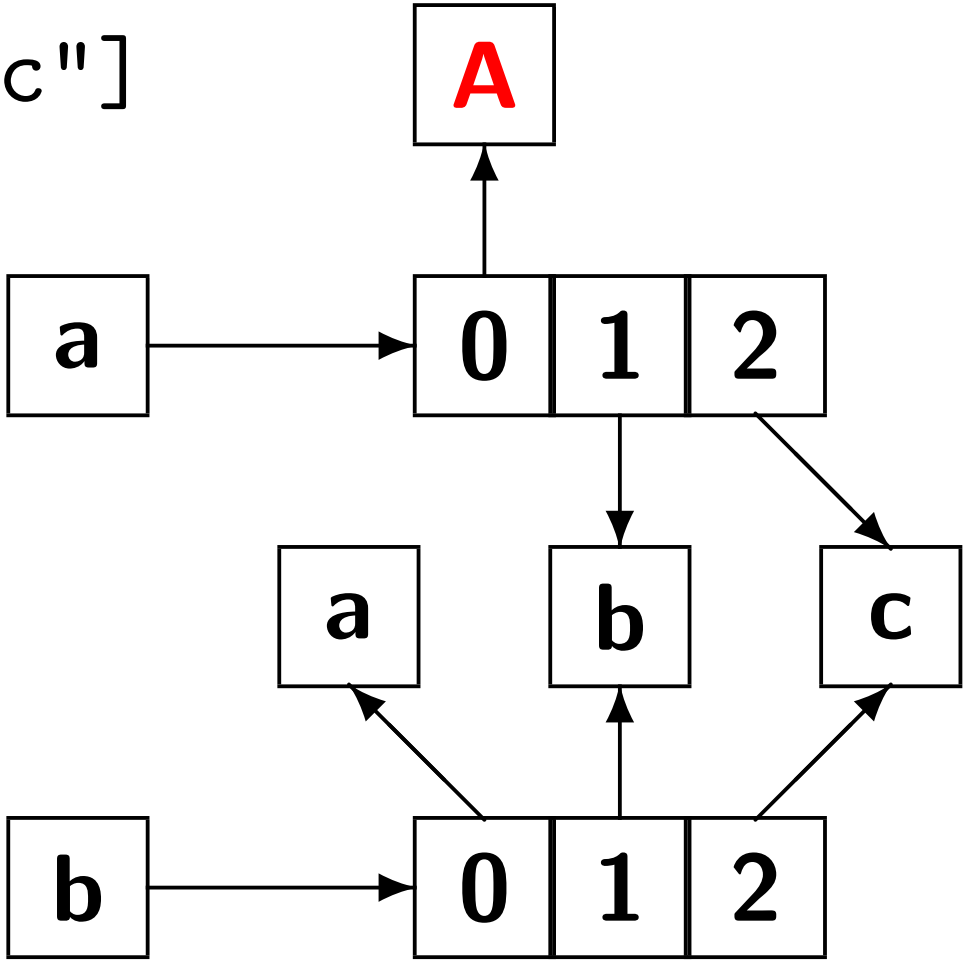
Array のコピー (2) (3)

```
a = ["a", "b", "c"]
b = a.dup
a[0] = "A"
a[1].upcase!
```



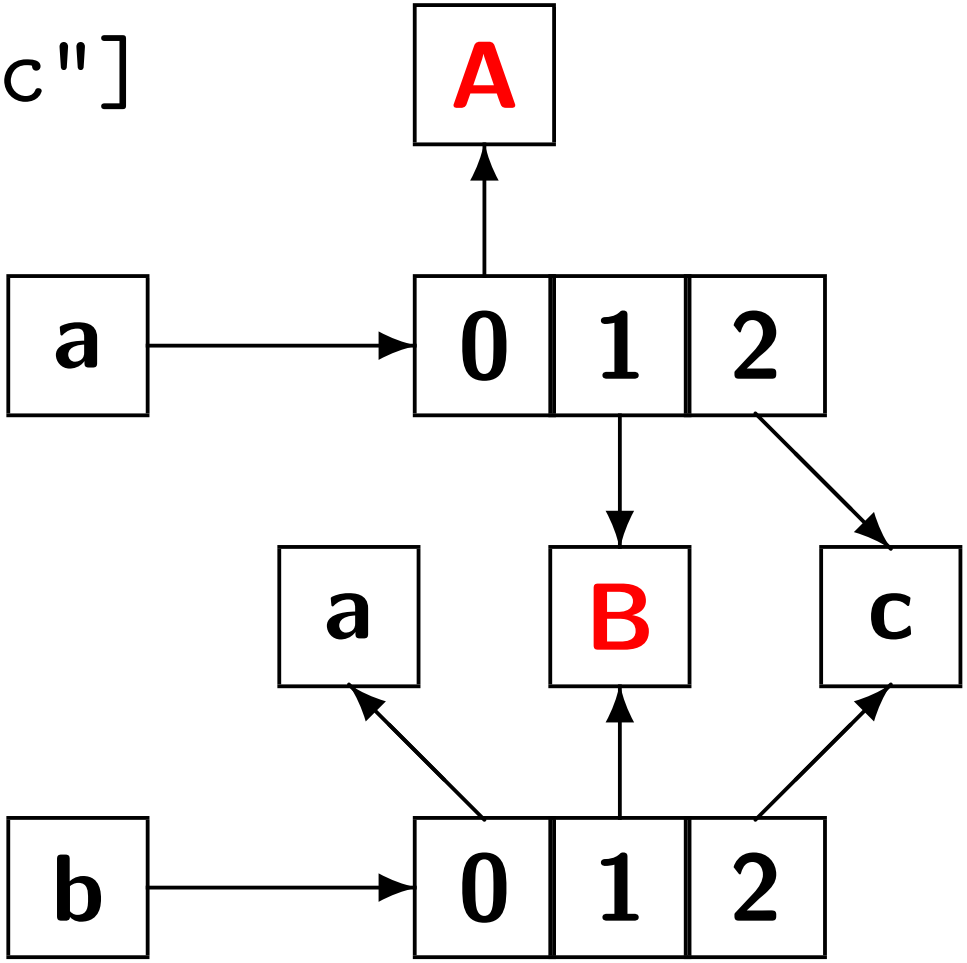
Array のコピー (2) (3)

```
a = ["a", "b", "c"]
b = a.dup
a[0] = "A"
a[1].upcase!
```



Array のコピー (2) (3)

```
a = ["a", "b", "c"]
b = a.dup
a[0] = "A"
a[1].upcase!
```



まとめ

- Array と Hash の作り方・使い方
  - 初期化・デフォルト値はブロックで
- 繰り返しはブロックで
- 浅いコピー・破壊に注意

- each はメソッド
- ブロックは Proc オブジェクト

演習問題 0

今日のレッスンで分からなかったこと、疑問に思ったことをグループで話し合ってみよう。

演習問題 1

空の Array a がある。これを [0, 1, 2, 3] にする方法をできるだけ挙げてみよう。
また、Array 以外のオブジェクト (例えば ”0,1,2,3”) から Array [0, 1, 2, 3] を作る方法をできるだけ挙げてみよう。

```
a = []
# なんとかして
a    # => [0, 1, 2, 3]
```

演習問題 2

文字列の中から、単語ごとの出現数、文字ごとの出現数を調べてみよう。

```
"No Ruby, No Life.".scan(/\w+/).
# なんとかして集計

# >>  2: No
# >>  1: Ruby
# >>  1: Life
```