APL

J

R

julia

NumPy

APL

J

R

julia

NumPy

A Look at Array Languages

APL vs BQN vs J

https://github.com/codereport/Content

https://github.com/codereport/array-language-comparisons/

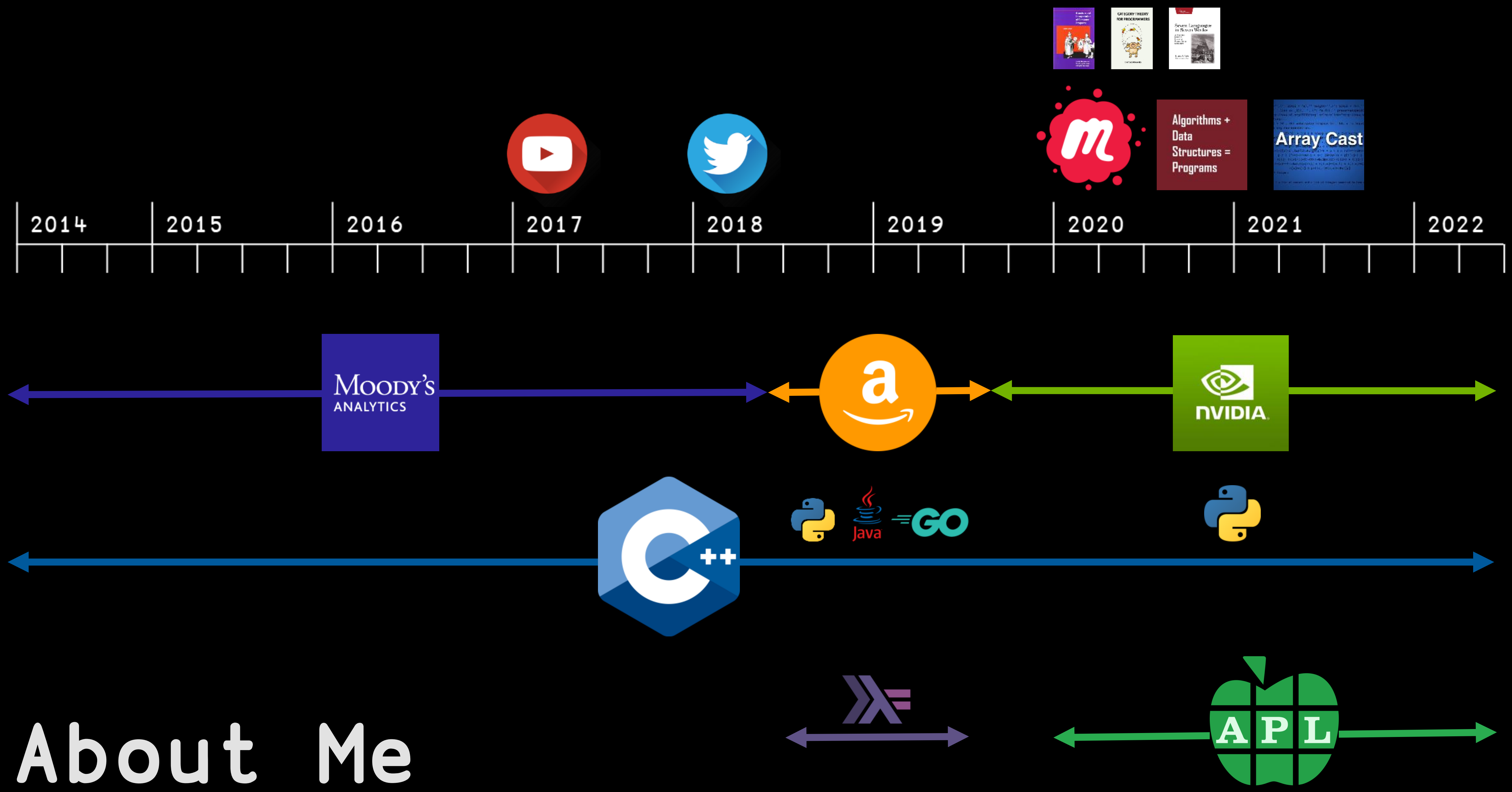
Conor Hoekstra

code_report

codereport

A Look at Array Languages

APL vs BQN vs J (vs Julia vs R vs NumPy)



About Me

Conor Hoekstra / @code_report

2319. Check if Matrix Is X-Matrix

A square matrix is said to be an X-Matrix if **both** of the following conditions hold:

1. All the elements in the diagonals of the matrix are non-zero.
1. All other elements are 0.

Given a square matrix, return **true** if grid is an X-Matrix. Otherwise, return **false**.

https://leetcode.com/problems/check-if-matrix

2	0	0	1
0	3	1	0
0	5	2	0
4	0	0	2

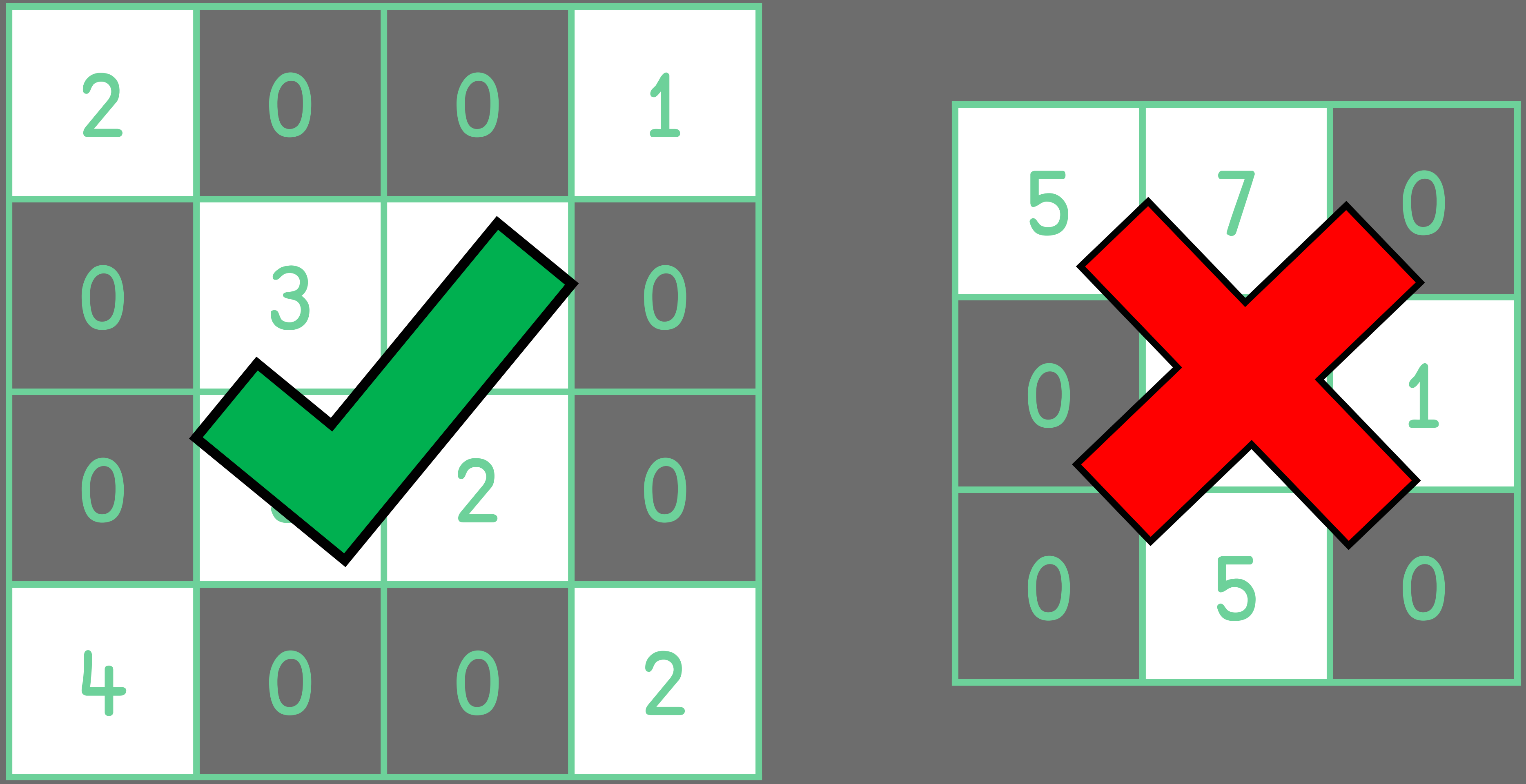
5	7	0
0	3	1
0	5	0

2	0	0	1
0	3	1	0
0	5	2	0
4	0	0	2

5	7	0
0	3	1
0	5	0

2	0	0	1
0	3	1	0
0	5	2	0
4	0	0	2

5	7	0
0	3	1
0	5	0


2 0 0 1
0 3 0
0 2 0
4 0 0 2
5 7 0
0 1
0 5 0

```
mat ← 4 4⍴2 0 0 1 0 3 1 0 0 5 2 0 4 0 0 2
```

```
mat ← 4 4⍴2 0 0 1 0 3 1 0 0 5 2 0 4 0 0 2
```

```
2 0 0 1
0 3 1 0
0 5 2 0
4 0 0 2
```

```
mat

2 0 0 1
0 3 1 0
0 5 2 0
4 0 0 2
```

```
≢mat

4
```

```
ι≢mat

1 2 3 4
```

```
(⍳≢mat)∘.,⍳≢mat
```

1 1	1 2	1 3	1 4
2 1	2 2	2 3	2 4
3 1	3 2	3 3	3 4
4 1	4 2	4 3	4 4

```
(⍳≢mat)∘.=⍳≢mat
```

```
1 0 0 0
0 1 0 0
0 0 1 0
0 0 0 1
```

```
∘.=⍨⍳≢mat
```

```
1 0 0 0
0 1 0 0
0 0 1 0
0 0 0 1
```

```
⌽∘.=⍨⍳≢mat

0 0 0 1
0 0 1 0
0 1 0 0
1 0 0 0
```

```
⌽∘.=⍨⍳≢mat    ∘.=⍨⍳≢mat

0 0 0 1        1 0 0 0
0 0 1 0        0 1 0 0
0 1 0 0        0 0 1 0
1 0 0 0        0 0 0 1
```

```
(⌽∘.=⍨⍳≢mat) ⌈ ∘.=⍨⍳≢mat
```

```
1 0 0 1
0 1 1 0
0 1 1 0
1 0 0 1
```

```
(⌽∘.=⍨⍳≢mat)⌈(⊢∘.=⍨⍳≢mat)
```

```
1 0 0 1
0 1 1 0
0 1 1 0
1 0 0 1
```

```
(⌽⌈⊢)∘.=⍨⍳≢mat
```

```
1 0 0 1
0 1 1 0
0 1 1 0
1 0 0 1
```

```
mat

2 0 0 1
0 3 1 0
0 5 2 0
4 0 0 2
```

```
1⌊mat

1 0 0 1
0 1 1 0
0 1 1 0
1 0 0 1
```

```
1⌊mat    (⌽⌈⊢)∘.=⍨⍳≢mat

1 0 0 1    1 0 0 1
0 1 1 0    0 1 1 0
0 1 1 0    0 1 1 0
1 0 0 1    1 0 0 1
```

```
(1⌊mat)≡(⌽⌈⊢)∘.=⍨⍳≢mat
```

```
1
```

```
(1⌊mat)≡(⌽⌈⊢)∘.=⍨⍳≢mat
```

```
{(1⌊⍵)≡(⌽⌈⊢)∘.=⍨⍳≢⍵}
```

```
checkMatrix ← {(1⌊⍵)≡(⌽⌈⊢)∘.=⍨⍳≢⍵}
```

```
CheckMatrix ← {(1⌊𝕩)≡⌽⟜⌈=⌜˜↕≠𝕩}
```

```
checkMatrix =. {{ (1<.y)-:(>.|.)=/~i.#y }}
```

```
checkMatrix ← {(1⌊⍵)≡(⌽⌈⊢)∘.=⍨⍳≢⍵}

CheckMatrix ← {(1⌊𝕩)≡⌽⊸⌈=⌜˜↕≠𝕩}

checkMatrix =. {{ (1<.y)-:(>.|.)=/~i.#y }}
```

```
{(1⌊⍵)≡(⌽⌈⊢)∘.=⍨⍳≢⍵}
```

```
{(1⌊𝕩)≡⌽⊸⌈=⌜˜↕≠𝕩}
```

```
{{ (1<.y)-:(>.|.)=/~i.#y }}
```

```
{(1⌊⍵)≡(⌽⌈⊢)∘.=⍨⍳≢⍵}
{(1⌊𝕩)≡⌽⟜⌈=⌜˜↕≠𝕩}
{{(1<.y)-:(>.|.)=/~i.#y}}
```

```
{(1⌊ ω)≡ (⌽⌈⊢ )∘.=⍨⍳ ≢ω}
{(1⌊ 𝕩)≡  ⌽⟜⌈  =⌜˜ ↕ ≠𝕩}
{{(1<.y)-:(>.|.)=/~ i.#y}}
```

Rank Polymorphism

Operators

(Modifiers | Adverbs / Conjunctions)

Combinators

Leading Axis Theory

Rank Polymorphism

Operators

(Modifiers | Adverbs / Conjunctions)

Combinators

Leading Axis Theory

```
{(1⌊ ⍵)≡ (⌽⌈⊢ )∘.=⍨⍳ ≢⍵}
{(1⌊ 𝕩)≡  ⌽⊸⌈  =⌜˜ ↕ ≠𝕩}
{{(1<.y)-:(>.|.)=/~ i.#y}}
```

Scalar (R0) ⌊ Matrix (R2)

```
{(1⌊ ⍵)≡ (⌽⌈⊢ )∘.=⍨⍳ ≢⍵}

{(1⌊ 𝕩)≡ ⌽⊸⌈ =⌜˜ ↕ ≠𝕩}

{{(1<.y)-:(>.|.)=/~ i.#y}}
```

Matrix (R2) ⌈ Matrix (R2)

1⌊𝕩

```
map (map (min 1)) x
```

Rank Polymorphism

Operators

(Modifiers | Adverbs / Conjunctions)

Combinators

Leading Axis Theory

Rank Polymorphism

Operators

(Modifiers | Adverbs / Conjunctions)

Combinators

Leading Axis Theory

```
{(1⌊ ⍵)≡ (⌽⌈⊢ )∘.=⍨⍳ ≢⍵}
{(1⌊ 𝕩)≡  ⌽⟜⌈  =⌜˜ ↕ ≠𝕩}
{{(1<.y)-:(>.|.)=/~ i.#y}}
```

```
{(1⌊ ⍵)≡ (⌽⌈⊢ )∘.=⍨⍳ ≢⍵}
{(1⌊ 𝕩)≡  ⌽⟜⌈  =⌜˜ ↕ ≠𝕩}
{{(1<.y)-:(>.|.)=/~ i.#y}}
```

1	2	3	4	5	6
1	2	3	4	5	6

	1	2	3	4	5	6
1						
2						
3						
4						
5						
6						

	1	2	3	4	5	6
1	2	3	4	5	6	7
2	3	4	5	6	7	8
3	4	5	6	7	8	9
4	5	6	7	8	9	10
5	6	7	8	9	10	11
6	7	8	9	10	11	12

	Name		Reduce		Scan		Outer Product
APL	Operators Adverbs	🟢	/ (reduce) ⌿ (reduce first)	🟢	\ (scan) ⍀ (scan first)	🟢	∘. (outer product)
J	Adverbs & Conjunctions	🟢	/ (insert)	🟢	\ (prefix)	🟢	/ (table)
BQN	Modifiers	🟢	´ (fold) ˝ (insert)	🟢	` (scan)	🟢	⌜ (table)
Q	Iterators	🟢	/ over	🟢	\ scan	🟢	/:\:
Julia	Functions	🟢	reduce	🟢	accumulate	🟡	broadcast
NumPy	Functions	🟢	_.reduce()	🟢	_.accumulate()	🟢	_.outer()
R	Functions	🟢	Reduce	🟢	Reduce(accumulate=TRUE)	🟢	outer
Nial	Transformers	🟢	REDUCE	🟢	ACCUMULATE	🟢	OUTER
Futhark	Functions SOAC	🟢	foldl/r reduce(_comm)	🟢	scan	🟡	outer_product
SaC		🔴	-	🔴	-	🔴	-

Rank Polymorphism

Operators

(Modifiers | Adverbs / Conjunctions)

Combinators

Leading Axis Theory

Rank Polymorphism

Operators

(Modifiers | Adverbs / Conjunctions)

Combinators

Leading Axis Theory

```
{(1⌊ ⍵)≡ (⌽⌈⊢ )∘.=⍨⍳ ≢⍵}
{(1⌊ 𝕩)≡  ⌽⊸⌈  =⌜˜ ↕ ≠𝕩}
{{(1<.y)-:(>.|.)=/~ i.#y}}
```

```
{(1⌊ ⍵)≡ (⌽⌈⊢ )∘.=⍨⍳ ≢⍵}
{(1⌊ 𝕩)≡  ⌽⟜⌈   =⌜˜ ↕ ≠𝕩}
{{(1<.y)-:(>.|.)=/~ i.#y}}
```

```
{(1⌊ ω)≡ (⌽⌈⊢ )∘.=⍨⍳ ≢ω}     ⌽
{(1⌊ 𝕩)≡ ⌽⊸⌈ =⌜˜ ↕ ≠𝕩}     Σ
{{(1<.y)-:(>.|.)=/~ i.#y}}  S
```

```
{(1⌊ω)≡ ⌽∘⌈⍨ ∘.=⍨⍳ ≢ω} D+W
{(1⌊ω)≡ (⌽⌈⊢) ∘.=⍨⍳ ≢ω} Φ
{(1⌊𝕩)≡ ⌽⊸⌈ =⌜˜ ↕ ≠𝕩} Σ
{(1⌊𝕩)≡ ⌈⟜⌽ =⌜˜ ↕ ≠𝕩} S
{{(1<.y)-:(>.|.)=/~ i.#y}} S
```

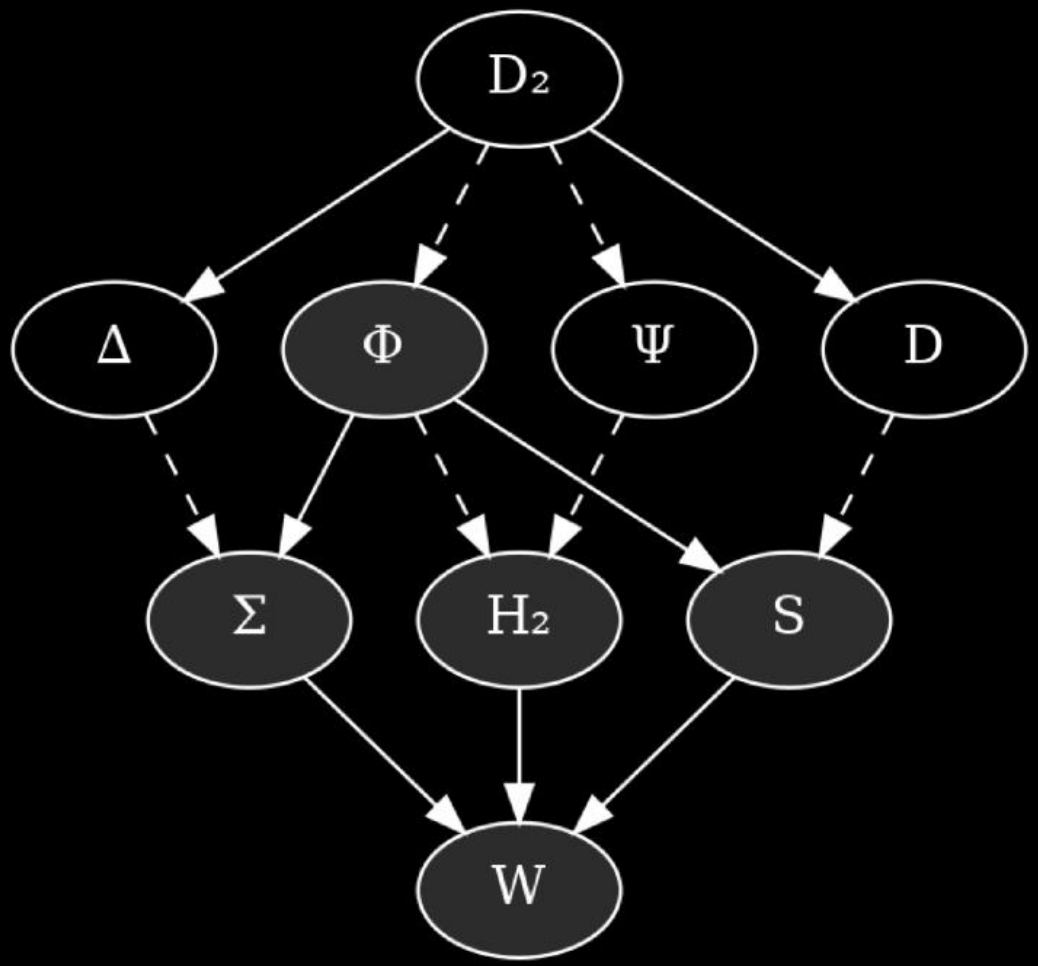


Figure 4.2: D_2 combinator hierarchy with Δ, Σ and H_2.

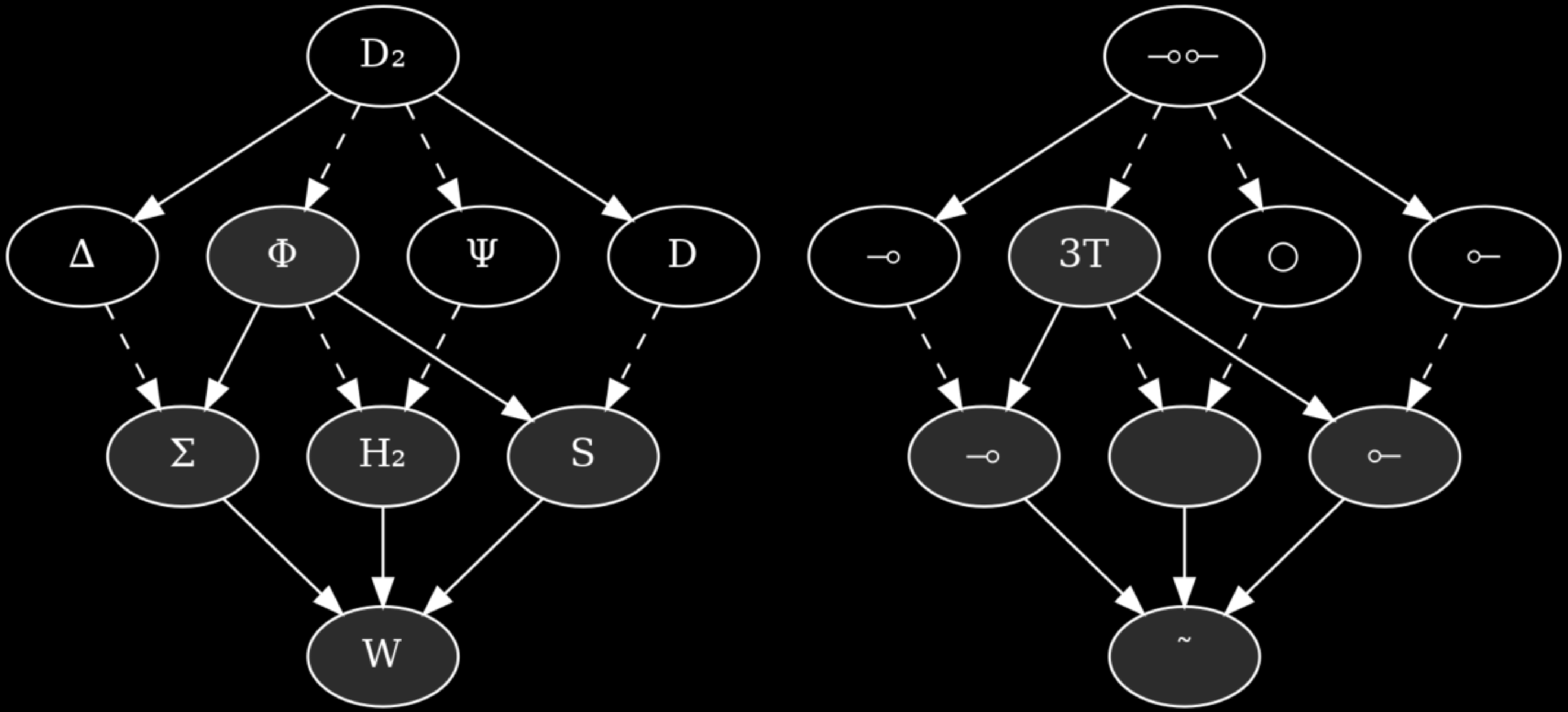


Figure 4.3: D_2 combinator hierarchy vs BQN spellings.

Rank Polymorphism

Operators

(Modifiers | Adverbs / Conjunctions)

Combinators

Leading Axis Theory

Rank Polymorphism

Operators

(Modifiers | Adverbs / Conjunctions)

Combinators

Leading Axis Theory

```
{(1⌊ ω)≡ (⌽⌈⊢ )∘.=⍨⍳ ≢ω}
{(1⌊ x)≡  ⌽⟜⌈  =⌜˜ ↕ ≠x}
{{(1<.y)-:(>.|.)=/~ i.#y}}
```

```
{(1⌊ω)≡(⌽⌈⊢)∘.=⍨⍳≢ω}
{(1⌊𝕩)≡⌽∘⌈=⌜˜↕≠𝕩}
{{(1<.y)-:(>.|.)=/~i.#y}}
```

⌽

⌽

|.

```
⌽ 1 2 3
⌽ 1 2 3
|. 1 2 3
```

```
⌽ 1 2 3 → 3 2 1
⌽ 1 2 3 → 3 2 1
|. 1 2 3 → 3 2 1
```

```
⌽ 1 2 3
  4 5 6
```

```
⌽ 1 2 3
  4 5 6
```

```
|. 1 2 3
   4 5 6
```

APL	⌽	1 2 3 4 5 6	→	3 2 1 6 5 4	⊖
	⌽	1 2 3 4 5 6	→	4 5 6 1 2 3	Leading Axis
J	|.	1 2 3 4 5 6	→	4 5 6 1 2 3	

	Reverse Columns	Reverse Rows
APL	⊖	⌽ ⊖⍤1
BQN	⌽	⌽˘ ⌽⎉1
J	|.	|."1

Rank Polymorphism

Operators

(Modifiers | Adverbs / Conjunctions)

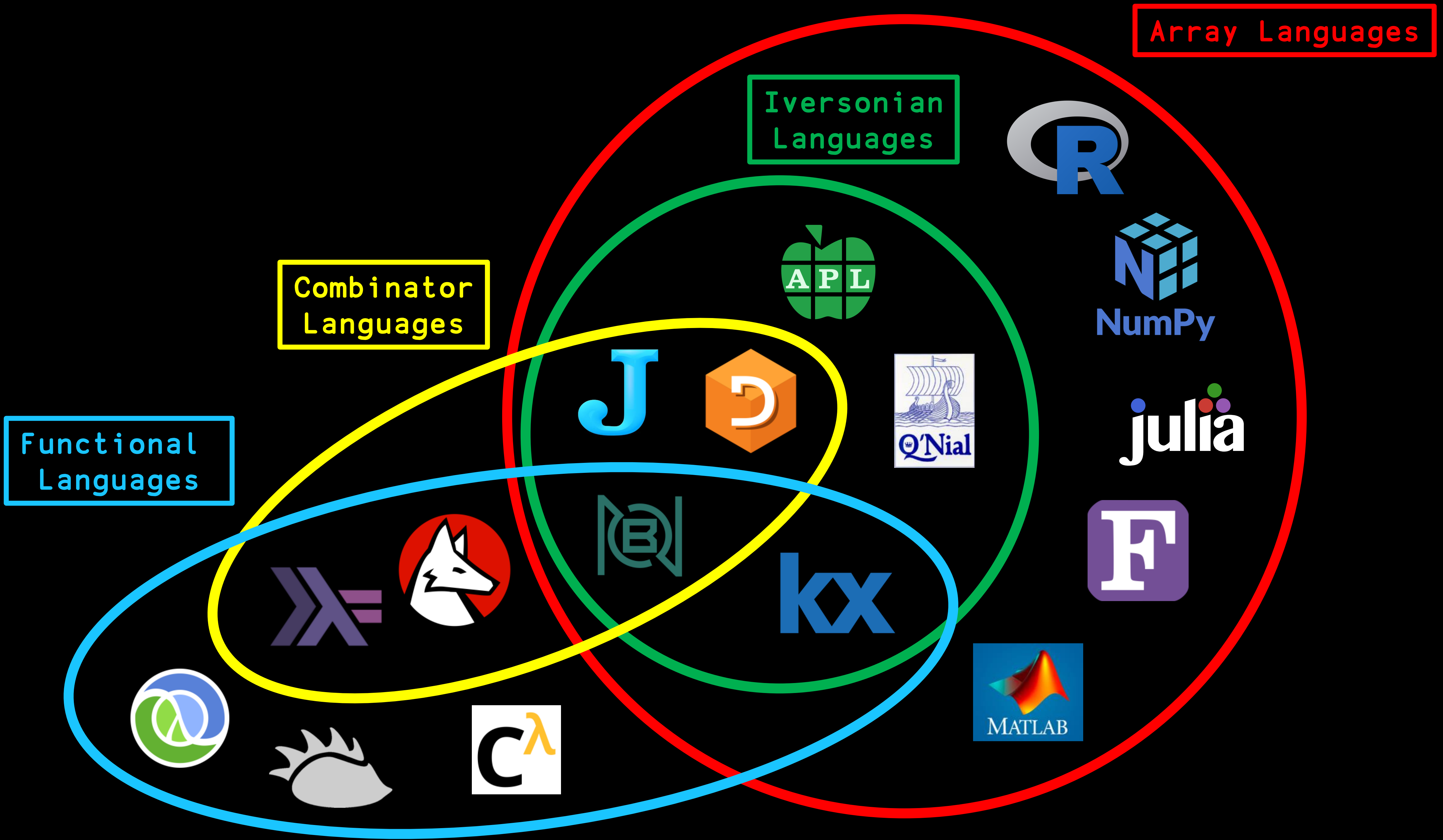
Combinators

Leading Axis Theory

```
{(1⌊ω)≡(⌽⌈⊢)∘.=⍨⍳≢ω}
{(1⌊𝕩)≡⌽⊸⌈=⌜˜↕≠𝕩}
{{(1<.y)-:(>.|.)=/~i.#y}}
```

```
checkMatrix: { a: til count x; i: a =/:\: a; m: i | reverse i; m ~ x <> 0 }
```

```
checkMatrix: {
    a: til count x;
    i: a =/:\: a;
    m: i | reverse i;
    m ~ x <> 0
}
```


Array Languages
Iversonian Languages
Combinator Languages
Functional Languages
APL
NumPy
julia
Q'Nial
MATLAB

```python
import numpy as np

def check_matrix(grid):
    i = np.identity(len(grid), int)
    return np.array_equal(np.minimum(grid, 1),
                          np.maximum(i, np.flip(i, axis=1)))
```

NumPy

```python
import numpy as np

def check_matrix(grid):
    n = len(grid)
    i = np.equal.outer(np.arange(n), np.arange(n))
    return np.array_equal(np.minimum(grid, 1),
                          np.maximum(i, np.flip(i, axis=1)))
```

```julia
function checkmatrix(grid)
  i = size(grid, 1) |> I |> Matrix
  min.(grid, 1) == max.(i, reverse(i, dims=1))
end
```

```julia
function checkmatrix(grid)
  n = size(grid, 1)
  i = 1:n .== 1:n |> transpose
  min.(grid, 1) == max.(i, reverse(i, dims=1))
end
```

```r
check_matrix <- function(grid) {
   i = diag(nrow(grid))
   all(pmin(grid, 1) == pmax(i, apply(i, 2, rev)))
}
```

```r
check_matrix <- function(grid) {
    n = nrow(grid)
    i = outer(1:n, 1:n, "==")
    all(pmin(grid, 1) == pmax(i, apply(i, 2, rev)))
}
```

Rank Polymorphism

Operators

(Modifiers | Adverbs / Conjunctions)

Combinators

Leading Axis Theory

	R	julia	NumPy	kx
Rank Polymorphism	✓	✓	✓	✓
Operators (Modifiers | Adverbs / Conjunctions)	✓	✓	✓	✓
Combinators				
Leading Axis Theory				✓

Arrays

```cpp
auto check_matrix(auto grid) -> bool {
    for (int i = 0; i <= grid.size(); ++i) {
        for (int j = 0; j <= grid.size(); ++j) {
            if (i == j or (grid.size() - i == j)) {
                if (grid[i][j] == 0) {
                    return false;
                }
            } else if (grid[i][j] != 0) {
                return false;
            }
        }
    }
    return true;
}
```

https://godbolt.org/z/PGMYvM184

```cpp
auto check_matrix(auto grid) -> bool {
    for (int i = 0; i < grid.size(); ++i) {
        for (int j = 0; j < grid.size(); ++j) {
            if (i == j or (grid.size() - i == j)) {
                if (grid[i][j] == 0) {
                    return false;
                }
            } else if (grid[i][j] != 0) {
                return false;
            }
        }
    }
    return true;
}
```

https://godbolt.org/z/PGMYvM184

```cpp
auto check_matrix(auto grid) -> bool {
    for (int i = 0; i < grid.size(); ++i) {
        for (int j = 0; j < grid.size(); ++j) {
            if (i == j or (grid.size() - i == j + 1)) {
                if (grid[i][j] == 0) {
                    return false;
                }
            } else if (grid[i][j] != 0) {
                return false;
            }
        }
    }
    return true;
}
```

https://godbolt.org/z/PGMYvM184

```
{(1⌊⍵)≡(⌽⌈⊢)∘.=⍨⍳≢⍵}
{(1⌊𝕩)≡⌽⊸⌈=⌜˜↕≠𝕩}
{{(1<.y)-:(>.|.)=/~i.#y}}
```

Thank you!

https://github.com/codereport/Content

https://github.com/codereport/array-language-comparisons/

Conor Hoekstra

code_report

codereport

Questions?

https://github.com/codereport/Content

https://github.com/codereport/array-language-comparisons/

Conor Hoekstra

code_report

codereport