

# APL Code Golf Autumn Tournament

Adám Brudzewsky  
October 26<sup>th</sup>, 2017



# Background

- What is Code Golf?

# Background

- What is Code Golf?
- Tradition – known to have been popular with earlier APL hackers. [Wikipedia]

# Background

- What is Code Golf?
- Tradition – known to have been popular with earlier APL hackers. [Wikipedia]
- Dyalog '17 workshop

# Do Code Golf!

- Encourages exploration – achieve excellence
- Opens mind – think out-of-the-box!

# Types of Code

# Types of Code

- Readable

# Types of Code

- Readable
- Fast

# Types of Code

- Readable
- Fast
- Short

# Types of Code

- Readable
- Fast
- Short
- Balanced

# Selected Problems

- Conway's Game of Life
- Interval Index
- Complete Teams

# Conway's Game of Life



# Conway's Game of Life

Given a Boolean matrix world, generate the next generation where

- any live cell with fewer than two live neighbours dies, as if caused by underpopulation.
- any live cell with two or three live neighbours lives on to the next generation.
- any live cell with more than three live neighbours dies, as if by overpopulation.
- any dead cell with exactly three live neighbours becomes a live cell, as if by reproduction.
- there are hard walls (equivalent to always dead cells) surrounding the world.

0	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
1	1	1	1	0	0	1	0	0	1	0	0	0	1	1	0	0	0	0
0	0	0	0	1	→	0	0	0	0	1	→	0	1	1	0	1	0	1
1	1	1	1	1	0	0	1	1	0	1	0	1	1	0	1	1	0	1
0	0	0	1	0	0	0	1	0	1	1	0	1	0	1	1	0	1	1

# Conway's Game of Life

Given a Boolean matrix world, generate the next generation where

- any live cell with fewer than two live neighbours dies, as if caused by underpopulation.
- any live cell with two or three live neighbours lives on to the next generation.
- any live cell with more than three live neighbours dies, as if by overpopulation.
- any dead cell with exactly three live neighbours becomes a live cell, as if by reproduction.
- there are hard walls (equivalent to always dead cells) surrounding the world.

0	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0
1	1	1	1	0	0	1	0	0	1	0	0	0	1	1	
0	0	0	0	1	0	0	0	1	0	1	1	0	1		
1	1	1	1	1	0	1	1	0	1	0	1	1	0	1	
0	0	0	1	0	0	1	0	1	1	0	1	0	1	1	

# Conway's Game of Life

Given a Boolean matrix world, generate the next generation where

- any live cell with fewer than two live neighbours dies, as if caused by underpopulation.
- any live cell with two or three live neighbours lives on to the next generation.
- any live cell with more than three live neighbours dies, as if by overpopulation.
- any dead cell with exactly three live neighbours becomes a live cell, as if by reproduction.
- there are hard walls (equivalent to always dead cells) surrounding the world.

0	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	0	0	1	0	0	1	0	0	0	1	1	0	0	0	1	1
0	0	0	0	1	0	0	0	0	1	0	1	1	0	1	0	1	1	0	1
1	1	1	1	1	0	1	1	0	1	0	1	1	0	1	1	0	1	1	1
0	0	0	1	0	0	1	0	1	1	0	1	0	1	1	1	0	1	1	1

# Conway's Game of Life

Given a Boolean matrix world, generate the next generation where

- any live cell with fewer than two live neighbours dies, as if caused by underpopulation.
- any live cell with two or three live neighbours lives on to the next generation.
- any live cell with more than three live neighbours dies, as if by overpopulation.
- any dead cell with exactly three live neighbours becomes a live cell, as if by reproduction.
- there are hard walls (equivalent to always dead cells) surrounding the world.

0	0	1	1	0	0	0	0	1	0	0	0	0	0	0
1	1	1	1	0	0	1	0	0	1	0	0	0	1	1
0	0	0	0	1	0	0	0	1	0	1	1	0	1	0
1	1	1	1	1	0	1	1	0	1	0	1	1	0	1
0	0	0	1	0	0	1	0	1	1	0	1	0	1	1

# Conway's Game of Life

Given a Boolean matrix world, generate the next generation where

- any live cell with fewer than two live neighbours dies, as if caused by underpopulation.
- any live cell with two or three live neighbours lives on to the next generation.
- any live cell with more than three live neighbours dies, as if by overpopulation.
- any dead cell with exactly three live neighbours becomes a live cell, as if by reproduction.
- there are hard walls (equivalent to always dead cells) surrounding the world.

0	0	1	1	0	0	0	0	0	0	0
1	1	1	1	0	0	1	0	0	1	1
0	0	0	0	1	0	0	0	1	1	0
1	1	1	1	1	0	1	1	0	1	1
0	0	0	1	0	0	1	0	1	0	1

→

0	0	0	1	0	0	0	0	0	0	0
0	1	0	1	1	0	1	1	0	1	1
0	0	0	0	0	1	1	1	0	1	1
0	1	0	1	1	1	1	1	0	1	1
0	1	0	1	1	1	1	1	0	1	1

# Conway's Game of Life

Given a Boolean matrix world, generate the next generation where

- any live cell with fewer than two live neighbours dies, as if caused by underpopulation.
- any live cell with two or three live neighbours lives on to the next generation.
- any live cell with more than three live neighbours dies, as if by overpopulation.
- any dead cell with exactly three live neighbours becomes a live cell, as if by reproduction.
- there are hard walls (equivalent to always dead cells) surrounding the world.

0	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
1	1	1	1	0	0	1	0	0	1	0	0	0	1	1	0	0	0	0
0	0	0	0	1	0	0	0	0	1	0	1	1	0	1	0	1	1	0
1	1	1	1	1	0	0	1	1	0	1	0	1	1	0	1	1	0	1
0	0	0	1	0	0	0	1	0	1	1	1	0	1	1	0	1	1	1

# Conway's Game of Life

Given a Boolean matrix world, generate the next generation where

- any live cell with fewer than two live neighbours dies, as if caused by underpopulation.
- any live cell with two or three live neighbours lives on to the next generation.
- any live cell with more than three live neighbours dies, as if by overpopulation.
- any dead cell with exactly three live neighbours becomes a live cell, as if by reproduction.
- there are hard walls (equivalent to always dead cells) surrounding the world.

0	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
1	1	1	1	0	0	1	0	0	1	0	0	0	1	1	0	0	0	0
0	0	0	0	1	→	0	0	0	0	1	→	0	1	1	0	1	0	1
1	1	1	1	1	0	0	1	1	0	1	0	1	1	0	1	1	0	1
0	0	0	1	0	0	0	1	0	1	1	0	1	0	1	1	0	1	1

# Conway's Game of Life

Given a Boolean matrix world, generate the next generation where a cell is alive if

- it was dead but had three alive neighbours (3 "including itself")
- it was alive and had two alive neighbours (3 including itself)
- it was alive and had three alive neighbours (4 including itself)

0 0 1 1 0	0 0 0 1 0	0 0 0 0 0
1 1 1 1 0	0 1 0 0 1	0 0 0 1 1
0 0 0 0 1	→ 0 0 0 0 1	→ 0 1 1 0 1
1 1 1 1 1	0 1 1 0 1	0 1 1 0 1
0 0 0 1 0	0 1 0 1 1	0 1 0 1 1

# Conway's Game of Life

Given a Boolean matrix world, generate the next generation where a cell is alive if

- it was dead but had three alive neighbours (3 "including itself")
- it was alive and had two alive neighbours (3 including itself)
- it was alive and had three alive neighbours (4 including itself)

0 0 1 1 0	0 0 0 1 0	0 0 0 0 0
1 1 1 1 0	0 1 0 0 1	0 0 0 1 1
0 0 0 0 1	0 0 0 0 1	0 1 1 0 1
1 1 1 1 1	0 1 1 0 1	0 1 1 0 1
0 0 0 1 0	1 0 1 1 1	0 1 0 1 1

→      →

1 1 1	1 1	0 1 1
0 0 0	1 0	1 1 1

# Conway's Game of Life

Given a Boolean matrix world, generate the next generation where a cell is alive if

- it was dead but had three alive neighbours (3 "including itself")
- it was alive and had two alive neighbours (3 including itself)
- it was alive and had three alive neighbours (4 including itself)

0	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0
1	1	1	1	0	0	1	0	0	1	0	0	0	1	1	1
0	0	0	0	1	0	0	0	0	1	0	1	1	0	1	1
1	1	1	1	1	1	0	1	1	0	1	1	1	0	1	1
0	0	0	1	0	0	1	1	1	1	0	1	0	1	1	1

# Conway's Game of Life

Given a Boolean matrix world, generate the next generation where a cell is alive if

- it was dead but had three alive neighbours (3 "including itself")
- it was alive and had two alive neighbours (3 including itself)
- it was alive and had three alive neighbours (4 including itself)

0	0	1	1	0	0	0	0	0	0
1	1	1	1	0	0	1	0	0	1
0	0	0	0	1	0	0	0	0	0
1	1	1	1	1	0	1	0	0	1
0	0	0	1	0	1	0	1	1	1

→

0	0	0	1	0	0	0	0	0	0
0	1	0	0	1	0	0	0	1	1
0	0	0	0	1	0	0	0	1	1
1	1	1	1	0	1	1	0	0	1
0	1	0	1	1	0	1	1	1	1

→

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	1
0	1	1	0	0	0	0	0	1	1
0	1	1	0	0	0	0	1	1	0
0	1	0	1	1	0	1	1	1	1

# Conway's Game of Life

- Traditional solution

```
{↑1 ω∨.^3 4=+/ , -1 0 1∘.⊖ -1 0 1∘.φ⊂ω}
```

See <https://youtu.be/a9xAKttWgP4>

# Conway's Game of Life

- Traditional solution

```
{↑1 ⍷ v. ^ 3 4 = + / , -1 0 1 o. ⓥ -1 0 1 o. Ⓛ ⊂ w}
```

See <https://youtu.be/a9xAKttWgP4>

- Golfed using version 16.0 features

```
{≠_l w} Ⓣ 3 3 ∈ `` 3 + 0 , `` ←
```

# Conway's Game of Life

## ➤ Traditional solution\*

criteria      neighbourhood  
 $\{\uparrow 1 \ \omega \vee .^3 \ 4 = + / , - 1 \ 0 \ 1 \circ . \ominus - 1 \ 0 \ 1 \circ . \phi \subset \omega\}$

See <https://youtu.be/a9xAKttWgP4>

## ➤ Golfed using version 16.0 features\*\*

neighbourhood      criteria  
 $\{\not\equiv \underline{\omega}\} \bowtie 3 \ 3 \in \cdot \cdot 3 + 0 , \cdot \cdot \vdash$

\* Life on a torus

\*\* Life on a plane

# Conway's Game of Life

{≠lω}⊗3 3∈..3+0,..+

# Conway's Game of Life

$$(\{\not\equiv_{\omega}\} \boxtimes 3 \ 3) \quad \in^{\cdots} \ (3+0, \cdots)$$

# Conway's Game of Life

$$( \{ \not\equiv \underline{\omega} \} \boxtimes 3 \ 3 )$$

# Conway's Game of Life

```
( {≠_ω}⊗3 3 )  
( {≤ω}⊗3 3 ) W
```

# Conway's Game of Life

```
( {≠_ω}⊗3 3 )
```

```
( {<ω}⊗3 3 ) W
```

W	0	0	1	1	0
1	1	1	1	1	0
0	0	0	0	0	1
1	1	1	1	1	1
0	0	0	1	0	

# Conway's Game of Life

$$(\{\not\models_{\mathcal{L}} \omega\} \boxdot 3 \quad 3)$$

(  $\{<\omega\}$   $\boxtimes$  3 3 ) W

W	0	1	1	0
0	0	1	1	0
1	1	1	1	0
0	0	0	0	1
1	1	1	1	1
0	0	0	1	0

# Conway's Game of Life

$$( \{ \not\equiv \underline{\omega} \} \bowtie 3 \ 3 )$$
$$( \{ \subset \omega \} \bowtie 3 \ 3 ) \ W$$

0 0 0	0 0 0	0 0 0	0 0 0	0 0 0
0 0 0	0 0 1	0 1 1	1 1 0	1 0 0
0 1 1	1 1 1	1 1 1	1 1 0	1 0 0
0 0 0	0 0 1	0 1 1	1 1 0	1 0 0
0 1 1	1 1 1	1 1 1	1 1 0	1 0 0
0 0 0	0 0 0	0 0 0	0 0 1	0 1 0

0	0	1	1	0
1	1	1	1	0
0	0	0	0	1
1	1	1	1	1
0	0	0	1	0

# Conway's Game of Life

$$( \{ \not\equiv \underline{\omega} \} \bowtie 3 \ 3 )$$
$$( \{ \subset \omega \} \bowtie 3 \ 3 ) \ W$$

0 0 0	0 0 0	0 0 0	0 0 0	0 0 0
0 0 0	0 0 1	0 1 1	1 1 0	1 0 0
1 1 1	1 1 1	1 1 1	1 1 0	1 0 0
0 0 0	0 0 1	0 1 1	1 1 0	1 0 0
0 1 1	1 1 1	1 1 1	1 1 0	1 0 0
0 0 0	0 0 0	0 0 0	0 0 1	0 1 0

0 0	1 1	1 1	0
1 1	0 0	0 0	1
0 0	1 1	1 1	1
0 0	0 0	0 1	0

# Conway's Game of Life

$$( \{ \not\equiv \_ \omega \} \bowtie 3 \ 3 )$$
$$( \{ \subset \omega \} \bowtie 3 \ 3 ) \ W$$

0 0 0	0 0 0	0 0 0	0 0 0	0 0 0
0 0 0	0 0 1	0 1 1	1 1 0	1 0 0
0 1 1	1 1 1	1 1 1	1 1 0	1 0 0
0 0 0	0 0 1	0 1 1	1 1 0	1 0 0
0 1 1	1 1 1	1 1 1	1 1 0	1 0 0
0 0 0	0 0 0	0 0 0	0 0 1	0 1 0

0 0 1	1 1 0	0 0 1	1 1 0	0 0 1
1 1 1	1 1 1	0 0 0	0 0 1	1 1 1
0 0 0	0 0 0	0 0 0	0 0 1	0 1 0
1 1 1	1 1 1	1 1 0	1 0 0	1 1 1
0 0 0	0 0 0	0 0 0	1 0 0	0 1 0

# Conway's Game of Life

$$( \{ \not\equiv \underline{\omega} \} \bowtie 3 \ 3 )$$
$$( \{ \subset \omega \} \bowtie 3 \ 3 ) \ W$$

0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	1	1	0	1	0
0	1	1	1	1	1	1	1	1	0	1	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	1	1	0	1	0	0
0	1	1	1	1	1	1	1	0	1	0	0
0	0	0	0	0	0	0	0	1	0	1	0

W	0	0	1	1	0
0	1	1	1	1	0
1	0	0	0	0	1
0	1	1	1	1	1
0	0	0	1	0	0

# Conway's Game of Life

```
( {≠_ω}⊗3 3)  
( {<ω}⊗3 3) W
```

W	0	0	1	1	0
1	1	1	1	1	0
0	0	0	0	0	1
1	1	1	1	1	1
0	0	0	1	0	

# Conway's Game of Life

```
( {≠_ω}⊗3 3)  
⇒ ( {≤ω}⊗3 3) W
```

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

				W
0	0	1	1	0
1	1	1	1	0
0	0	0	0	1
1	1	1	1	1
0	0	0	1	0

# Conway's Game of Life

$$\begin{aligned} & (\{\not\equiv \underline{\omega}\} \boxtimes 3 \ 3) \\ \Rightarrow \quad & (\{\subset \omega\} \boxtimes 3 \ 3) \ W \end{aligned}$$

0 0 0  
0 0 0  
0 1 1

$\underline{\omega}$   $\Rightarrow (\{\subset \omega\} \boxtimes 3 \ 3) \ W$

3	2	3	3
---	---	---	---

			W		
0	0	1	1	1	0
1	1	1	1	1	0
0	0	0	0	0	1
1	1	1	1	1	1
0	0	0	1	0	0

# Conway's Game of Life

$$\begin{aligned} & (\{\not\equiv \underline{\omega}\} \boxtimes 3 \ 3) \\ \Rightarrow & (\{\subset \omega\} \boxtimes 3 \ 3) \ W \end{aligned}$$

0 0 0  
0 0 0  
0 1 1

$\underline{\omega}$   $\Rightarrow (\{\subset \omega\} \boxtimes 3 \ 3) \ W$

3	2	3	3
---	---	---	---

$\not\equiv \underline{\omega}$   $\Rightarrow (\{\subset \omega\} \boxtimes 3 \ 3) \ W$

W	0	0	1	1	0
1	1	1	1	1	0
0	0	0	0	0	1
1	1	1	1	1	1
0	0	0	1	0	

# Conway's Game of Life

$$\begin{aligned} & (\{\not\equiv \underline{\omega}\} \boxtimes 3 \ 3) \\ \Rightarrow & (\{\subset \omega\} \boxtimes 3 \ 3) \ W \end{aligned}$$

0 0 0  
0 0 0  
0 1 1

$\underline{l}$   $\Rightarrow (\{\subset \omega\} \boxtimes 3 \ 3) \ W$

3	2	3	3
---	---	---	---

+/,  $\Rightarrow (\{\subset \omega\} \boxtimes 3 \ 3) \ W$   
2

W	0	0	1	1	0
1	1	1	1	1	0
0	0	0	0	0	1
1	1	1	1	1	1
0	0	0	1	0	0

# Conway's Game of Life

$$(\{\not\equiv_{\omega}\} \boxtimes 3 \ 3) \quad \in^{\cdots} \ (3+0, \cdots)$$

# Conway's Game of Life

( 3+0 , ``- )

					W
0	0	1	1	1	0
1	1	1	1	1	0
0	0	0	0	0	1
1	1	1	1	1	1
0	0	0	1	0	

# Conway's Game of Life

```
( 3+0 , ``⊣ )  
( 3 + ( 0 , ``⊣ ) )
```

W	0	0	1	1	0
1	1	1	1	1	0
0	0	0	0	0	1
1	1	1	1	1	1
0	0	0	1	0	

# Conway's Game of Life

```
( 3+0 , ``` )  
( 3 + ( 0 , ``` ) )  
( 0 , ``` ) W
```

					W
0	0	1	1	1	0
1	1	1	1	1	0
0	0	0	0	0	1
1	1	1	1	1	1
0	0	0	1	1	0

# Conway's Game of Life

( 3+0 , ``⊣ )

( 3 + ( 0 , ``⊣ ) )

( 0 , ``⊣ ) W

0	0	1	1	0
1	1	1	1	0
0	0	0	0	1
1	1	1	1	1
0	0	0	1	0

0	0	0	0	0	1	0	1	0	0
0	1	0	1	0	1	0	1	0	0
0	0	0	0	0	0	0	0	0	1

# Conway's Game of Life

( 3+0 , ``⊣ )

( 3 + ( 0 , ``⊣ ) )

( 3 + 0 , ``⊣ ) W

0	0	1	1	0
1	1	1	1	0
0	0	0	0	1
1	1	1	1	1
0	0	0	1	0

3	3	3	3	3	4	3	4	3	3
3	4	3	4	3	4	3	4	3	3
3	3	3	3	3	3	3	3	3	4

# Conway's Game of Life

$$(\{\not\equiv \omega\} \bowtie 3 \ 3) \in^{\cdot\cdot} (3+0, \cdot\cdot\vdash)$$

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

3	3	3	3	3	4	3	4	3	3
3	4	3	4	3	4	3	4	3	3
3	3	3	3	3	3	3	3	3	4
3	4	3	4	3	4	3	4	3	4
3	3	3	3	3	3	3	4	3	3

0	0	1	1	0
1	1	1	1	0
0	0	0	0	1
1	1	1	1	1
0	0	0	1	0

# Conway's Game of Life

( {≠\_ω}⊗3 3 ∈ ∘∘3+0 , ∘∘⊤ ) ⍵

0	0	0	1	0
0	1	0	0	1
0	0	0	0	1
0	1	1	0	1
0	1	0	1	1

W	0	0	1	1	0
1	1	1	1	1	0
0	0	0	0	0	1
1	1	1	1	1	1
0	0	0	1	0	0

# Conway's Game of Life

```
({≠_ω}⊗3 3 ∈ ∘∘3+0, ∘∘⊣) W
```

0	0	0	1	0
0	1	0	0	1
0	0	0	0	1
0	1	1	0	1
0	1	0	1	1

W	0	0	1	1	0
	1	1	1	1	0
	0	0	0	0	1
	1	1	1	1	1
	0	0	0	1	0

• ONLY 17 CHARS •

# Interval Index

## using pre-16.0 primitives



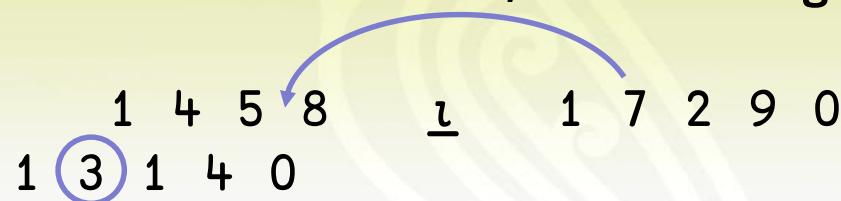
# Interval Index

Implement 16.0's  $\alpha \underline{\imath} \omega$  for vectors, using pre-16.0 Dyalog APL.  
For each element of  $\omega$ , find which "gap" of  $\alpha$  it belongs in.

1	4	5	8	<u>  </u>	1	7	2	9	0
1	3	1	4	0					

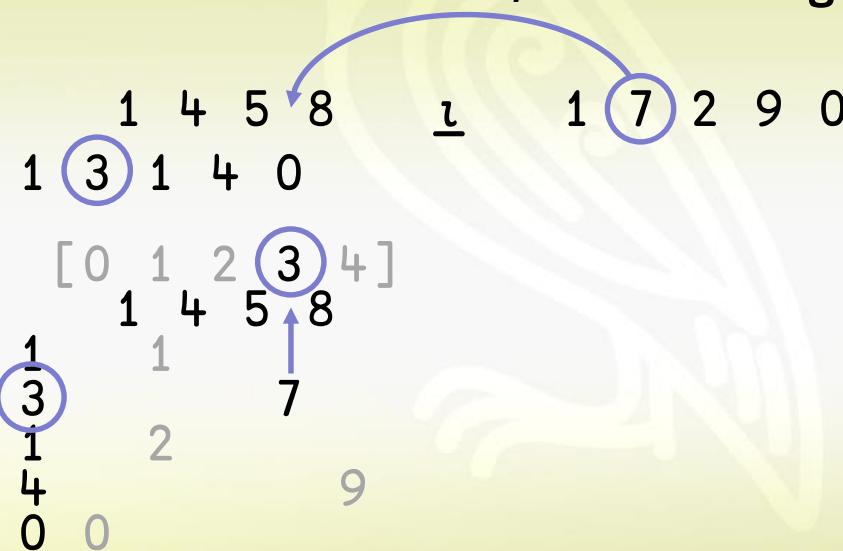
# Interval Index

Implement 16.0's  $\alpha \underline{\imath} \omega$  for vectors, using pre-16.0 Dyalog APL.  
For each element of  $\omega$ , find which "gap" of  $\alpha$  it belongs in.



# Interval Index

Implement 16.0's  $\alpha \underline{\imath} \omega$  for vectors, using pre-16.0 Dyalog APL.  
For each element of  $\omega$ , find which "gap" of  $\alpha$  it belongs in.



# Interval Index

Implement 16.0's  $\alpha \underline{\imath} \omega$  for vectors, using pre-16.0 Dyalog APL.  
For each element of  $\omega$ , find which "gap" of  $\alpha$  it belongs in.

	1	4	5	8	<u>  </u>	1	7	2	9	0
1	3	1	4	0						
	[0	1	2	3	4]					
	1	4	5	8						
1										
3										
1										
4										
0	0									

# Interval Index

## ➤ Traditional solution

```
{  
    a←⍋α,ω  
    i←(ι≠α),(#ω)ρ0  
    g←⌈\i[ a]  
    g[aι(≠α)+ι≠ω]  
}
```

A indices which will sort ( $\alpha, \omega$ ) ascending  
A indices of  $\alpha$ , and one zero per item of  $\omega$   
A the index of  $\alpha$  which the sorted ( $\alpha, \omega$ ) follows (= gap number)  
A gap numbers where items of  $\omega$  are found in the sorted array

# Interval Index

- Traditional solution

```
{  
    a←⍋α,ω           ⌈A indices which will sort (α,ω) ascending  
    i←(ι≠α), (≠ω)ρ0 ⌈A indices of α, and one zero per item of ω  
    g←⌈\i[ a]         ⌈A the index of α which the sorted (α,ω) follows (= gap number)  
    g[ aι(≠α)+ι≠ω]   ⌈A gap numbers where items of ω are found in the sorted array  
}
```

- Golfed

```
+⌿∘.(</∘⍋,)
```

# Interval Index

- Traditional solution

```
{  
    a←⍋α,ω           ⌈ A indices which will sort (α,ω) ascending  
    i←(ι≠α), (≠ω)ρ0 ⌈ A indices of α, and one zero per item of ω  
    g←⌈\i[ a]         ⌈ the index of α which the sorted (α,ω) follows (= gap number)  
    g[ aι(≠α)+ι≠ω]   ⌈ gap numbers where items of ω are found in the sorted array  
}
```

- Golfed  
 $+/\circ.(</\circ⍋,)$

- In version 16.0  
l

# Interval Index

4 7 (+ ≠ ∘ . <) 5 2 8

# Interval Index

4 7 (+≠∘. <) 5 2 8

4 7 ⋄. < 5 2 8

# Interval Index

4 7 (+≠∘. <) 5 2 8

4 7 ∘. < 5 2 8

< 5 2 8

4 1 0 1

7 0 0 1

# Interval Index

```
    4 7  (+-/ . <) 5 2 8  
[0 1 2]  
    4 7      . <      5 2 8  
  
< 5 2 8  
4 1 0 1  
7 0 0 1  
+/  
1 0 2
```

# Interval Index

4 7 (+≠∘. <) 5 2 8

1 0 2

[0 1 2]

4 7

1 0 2  
2 5  
0 2 8

# Interval Index

'D' 'G' (+≠∘.⟨) 'E' 'B' 'H'

# Interval Index

'D' 'G' (+≠◦.<>) 'E' 'B' 'H'

DOMAIN ERROR

# Interval Index

'D' 'G' (+≠◦.<<) 'E' 'B' 'H'

DOMAIN ERROR

'D' 'G' ◦.< 'E' 'B' 'H'

DOMAIN ERROR

# Interval Index

'D' 'G' (+-/ .<) 'E' 'B' 'H'

DOMAIN ERROR

'D' .< 'E'

# Interval Index

'D' 'G' (+-/ . <) 'E' 'B' 'H'

DOMAIN ERROR

'D' < 'E'

# Interval Index

'D' 'G' (+f. .<) 'E' 'B' 'H'

DOMAIN ERROR

'D' < 'E'

DOMAIN ERROR

# Interval Index

'D' 'G' (+-/ . <) 'E' 'B' 'H'  
DOMAIN ERROR  
    ↓ 'D' , 'E'

# Interval Index

'D' 'G' (+f o . <) 'E' 'B' 'H'  
DOMAIN ERROR  
    ↑ 'D' , 'E'  
1 2

# Interval Index

'D' 'G' (+-/ .<) 'E' 'B' 'H'  
DOMAIN ERROR  
    ↑ 'D' , 'E'  
1 2  
    </ ↑ 'D' , 'E'  
1

# Interval Index

		'D'	'G'	(+/-○. <)	'E'	'B'	'H'
DOMAIN	ERROR						
		↳ 'D'			,	'E'	
1	2						
		</	↳ 'D'		,	'E'	
1							
		'D'		(</○↳ , )	'E'		
1							

# Interval Index

# Interval Index

'D' 'G'      ⋸ . ( < / ⋸ ⌂ , )      'E' 'B' 'H'

# Interval Index

	'D'	'G'	◦ . ( < / ◦ ↳ , )	'E'	'B'	'H'
< / ◦ ↳ ,	E	B	H			
D	1	0	1			
G	0	0	1			

# Interval Index

'D' 'G' (+/∘.(</∘⍋,)) 'E' 'B' 'H'

'D' 'G' ∘.(</∘⍋,) 'E' 'B' 'H'  
</∘⍋, E B H  
D 1 0 1  
G 0 0 1

# Interval Index

```
'D' 'G' (+/∘.(</∘⌺,)) 'E' 'B' 'H'  
[0 1 2]  
'D' 'G' ∘.(</∘⌺,) 'E' 'B' 'H'  
</∘⌺, E B H  
D 1 0 1  
G 0 0 1  
+/  
1 0 2
```

# Interval Index

'DG' (+≠∘.(</∘��,)) 'EBH'

1 0 2

[0 1 2]  
D G  
1 E  
0 B  
2 H

# Interval Index

' DG '

1 0 2

l

' EBH '

	0	1	2
D	G		
E			
B			H

# Complete Teams



# Complete Teams

Get the numbers that occur exactly twice.

7 1 8 2 8 1 8 2  
1 2

1 4 1 5 9 9 6 5  
1 5 9

# Complete Teams using pre-16.0 features only

- Naïve traditional solution (“Readable”)  
 $\{ \cup (2 = +/\omega \circ . = \omega) / \omega \}$  ⋄ compare all, then select those with two matches

# Complete Teams using pre-16.0 features only

- Naïve traditional solution (“Readable”)

```
{ u(2=+/ω∘.=ω)/ω} ⍝ compare all, then select those with two matches
```

- Efficient traditional solution (“Fast”)

```
{  
    s←ω[⍋ω]           ⍝ sort  
    m←1,2≠/s          ⍝ mark partitions  
    i←{ω/ιρω}m,1     ⍝ where partitions begin  
    t←(2=¬2-/i,1)/i  ⍝ ... and are length-two  
    s[t]               ⍝ get those  
}
```

# Complete Teams including using 16.0 features

- Golfed ("Short")  
 $\in \{2 = \# \omega\} \boxtimes \subseteq \cup$

# Complete Teams including using 16.0 features

- Golfed ("Short")  
 $\in \{ 2 = \# \omega \} \boxdot \subseteq \cup$
- Production efficiency ("Balanced")  
 $\{ ( 2 = \{ \# \omega \} \boxdot \omega ) / \cup \omega \}$

# Complete Teams

$$\in \{2 = \# \omega\} \boxtimes \subseteq \cup$$

# Complete Teams

$$\in (\{2 = \# \omega\} \boxtimes \subseteq \cup)$$

# Complete Teams

$$( \{ 2 = \# \omega \} \boxtimes \subseteq \cup )$$

# Complete Teams

$$((\{2=\#ω\}⊐) ⊆ (\cup))$$

# Complete Teams

7 1 8 2 8 1 8 2  
N

( { 2 = ⍷ ω } ⎕ ) N

# Complete Teams

7 1 8 2 8 1 8 2  
N

({2=≠ω}目) N

({<ω}目) N

# Complete Teams

( {2=≠ω}⊐) N

( {<ω}⊐) N

1	2	6	3	5	7	4	8
---	---	---	---	---	---	---	---

N  
7 1 8 2 8 1 8 2  
[ 1 2 3 4 5 6 7 8 ]

# Complete Teams

( { $2 = \# \omega$ } ) N

( { $\subset \omega$ } ) N

1	2	6	3	5	7	4	8
---	---	---	---	---	---	---	---

1                    ( { $\# \omega$ } ) N

2                    3                    2

7    1    8    N  
[ 1    2    3    4    5    6    7    8 ]

# Complete Teams

( { $2 = \# \omega$ } ) N

( { $\subset \omega$ } ) N

1	2	6	3	5	7	4	8
---	---	---	---	---	---	---	---

7	1	8	N	2	8	1	8	2
1	2	3	4	5	6	7	8	

( { $\# \omega$ } ) N

1 2 3 2

( { $2 = \# \omega$ } ) N

0 1 0 1

# Complete Teams

$$(\ ((\{2=\# \omega\} \boxdot) \subseteq (\cup))$$

7 1 8 2 N 8 1 8 2

# Complete Teams

7	1	8	N	2	8	1	8	2
( u )	N							
7	1	8	2					

# Complete Teams

$$\begin{array}{c} ((\{2=\#ω\})\boxtimes) \subseteq (\cup)_N \\ 0 \quad 1 \quad 0 \quad 1 \subseteq 7 \quad 1 \quad 8 \quad 2 \end{array}$$

1	2
---	---

7 1 8 2 N  
8 1 8 2

# Complete Teams

$\left( \left( \{ 2 = \# \omega \} \right) \right) \subseteq \left( \cup \right) N$

$\in \quad 0 \ 1 \ 0 \ 1 \quad \subseteq \ 7 \ 1 \ 8 \ 2$

1 2

# Complete Teams

$$\in ((\{2=\# \omega\} \boxtimes) \subseteq (\cup)^N N$$

7 1 8 2 8 1 8 2  
N

1 2

# Complete Teams

7 1 8 2 N 8 1 8 2

( ∈ { 2 = ≠ ω } ⊕ ⊆ ∪ ) N

1 2

# Complete Teams performance comparison

```
N←?1000⍴1000
```

```
]Runtime -compare "Readable N" "Fast N" "Short N" "Balanced N"
```

Readable N	→ 3.0E <sup>-3</sup>	0%	██
* Fast N	→ 2.7E <sup>-5</sup>	-100%	█
Short N	→ 6.3E <sup>-4</sup>	-80%	███████████
Balanced N	→ 7.2E <sup>-5</sup>	-98%	█

# Complete Teams performance comparison

```
N←?10000⍴10000
```

```
]Runtime -compare "Readable N" "Fast N" "Short N" "Balanced N"
```

Readable N	→ 6.4E-3	0%	██
* Fast N	→ 6.3E-5	-100%	█
Short N	→ 2.7E-3	-58%	██
Balanced N	→ 1.0E-4	-99%	█

# THANKS FOR PLAYING!

1.	Nikolay Georgiev Nikolov	143
2.	Stefano Lanzavecchia	145
3.	Francesco Garue	145
4.	Michele Bellon	157
5.	Zachary Taylor	174
6.	Pierre	208
7.	Veli-Matti Jantunen	215
8.	Gilgamesh Athoraya	218
9.	Giacomo Manfredi	220
10.	Moris Zucca	222

Want more?

[codegolf.stackexchange.com](https://codegolf.stackexchange.com)

# New Features in Dyalog APL

- 14.0
  - (+-/÷1⌈≠) trains
  - ⎕ Key
  - ⚈ Rank
- 15.0
  - ⌊MKDIR ⌋NDELETE ⌋NEXISTS ⌋NGET ⌋NINFO ⌋NPARTS ⌋NPUT
- 16.0
  - ⌐ω Where ⌐⌐ω Interval Index
  - ⌐≤ω Nest ⌐⌐≤ω Partition
  - @ At
  - ⌈Stencil
  - ⌋CSV ⌋JSON

Next Webinar  
**Source Code Management  
with GitHub and APL**  
December 7<sup>th</sup>, 2017

