DYALOG

SimCorp

Workshop:
Advent of Code – day 2

Adám Brudzewsky
Richard Park
Rodrigo Girão Serrão

Day 1

- Array-oriented techniques
- Bingo and Strings

Day 2

- Reading and parsing data from files
- Mathematical insights for array-oriented programming

Problems

- 15.2 Wrapping Presents
- 15.17 Filling Containers
- 16.3 Triangles

Topics

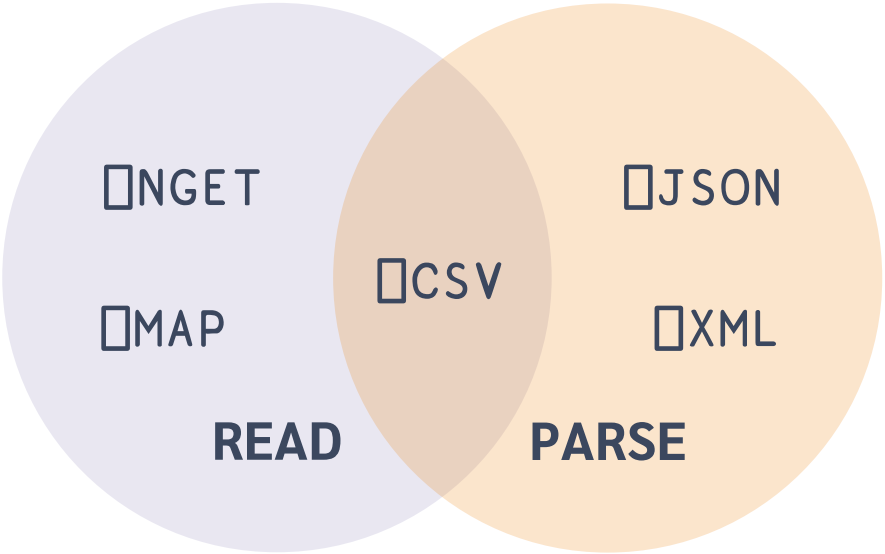
- Reading and parsing data from files
- Mathematical insights for array-oriented programming

Reading and Parsing Input

System Functions

Reading and Parsing Input

System Functions



⍞NGET

- Vector of characters:

```
      ⍞←⍞NGET 'filename'
Hello,
World!
```

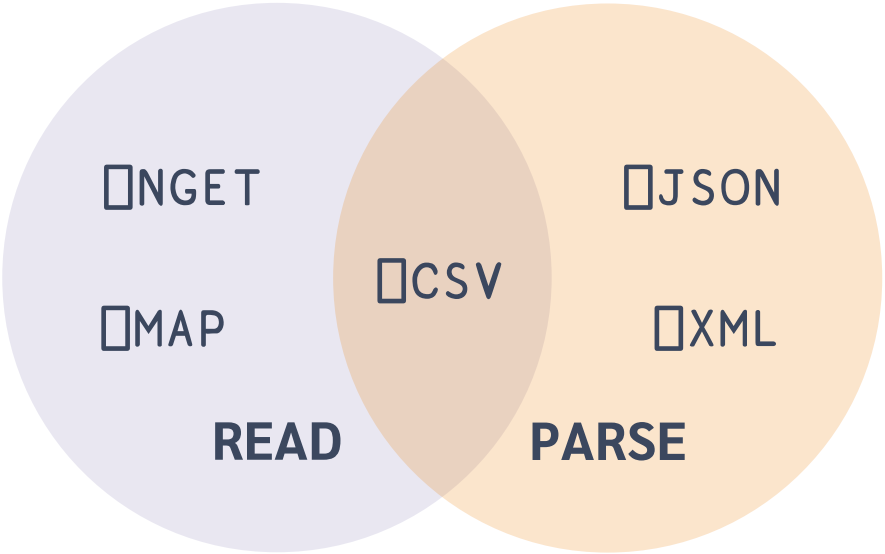
Line separator is always ⎕UCS 10

- Vector of character vectors:

```
      ⍞←⍞NGET 'filename' 1
┌→──────────┐
│ ┌→──────┐ ┌→──────┐ │
│ │Hello, │ │World! │ │
└∊──────────┘ └───────┘ ┘
```

Reading and Parsing Input

System Functions



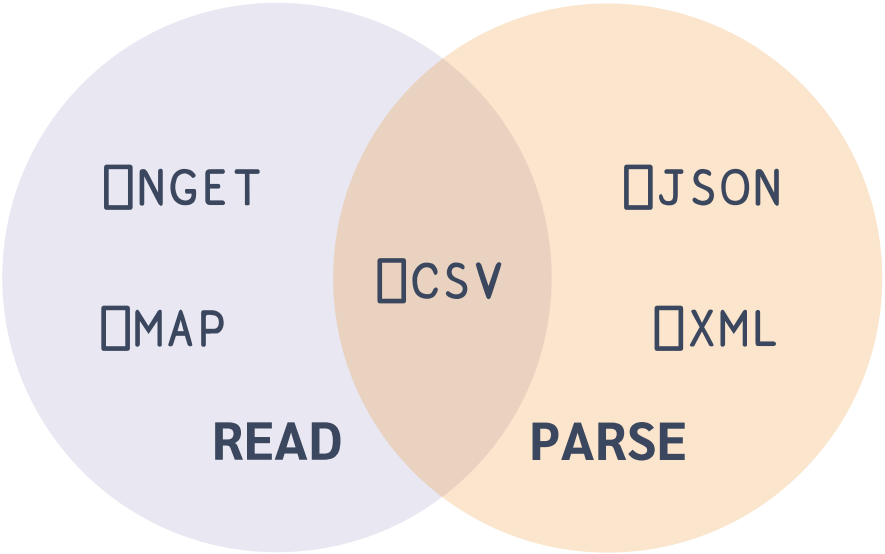
□MAP

- Vector of characters: `80 ¯1 □MAP filename`
- n-row matrix: `80 n ¯1 □MAP filename`
- n-column matrix: `80 ¯1 n □MAP filename`
- n-column m-row array: `80 ¯1 n m □MAP filename`

etc.

Reading and Parsing Input

System Functions



□CSV

- From array: `□CSV ('1,2,3' '4,5,6') ⍬ 4`

```
┌→──────┐
↓1 2 3  │
 4 5 6  │
└~──────┘
```

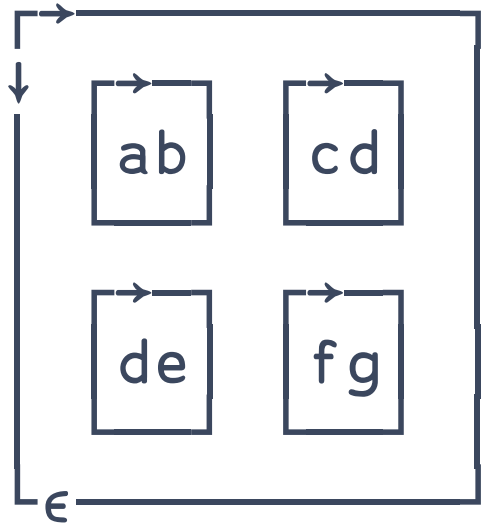
- From file: `□CSV 'foo.csv' ⍬ 4`

```
┌→──────┐
↓1 2 3  │
 4 5 6  │
└~──────┘
```

2 to error on non-numbers

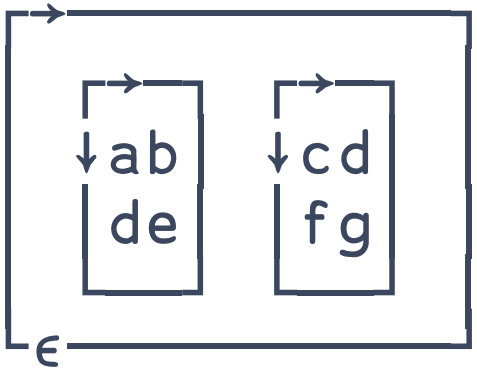
⍞CSV

```
⍞CSV⊂'ab,cd' 'de,fg'
```



⎕CSV

```
⎕CSV⍠'Invert'1⊂'ab,cd' 'de,fg'
```

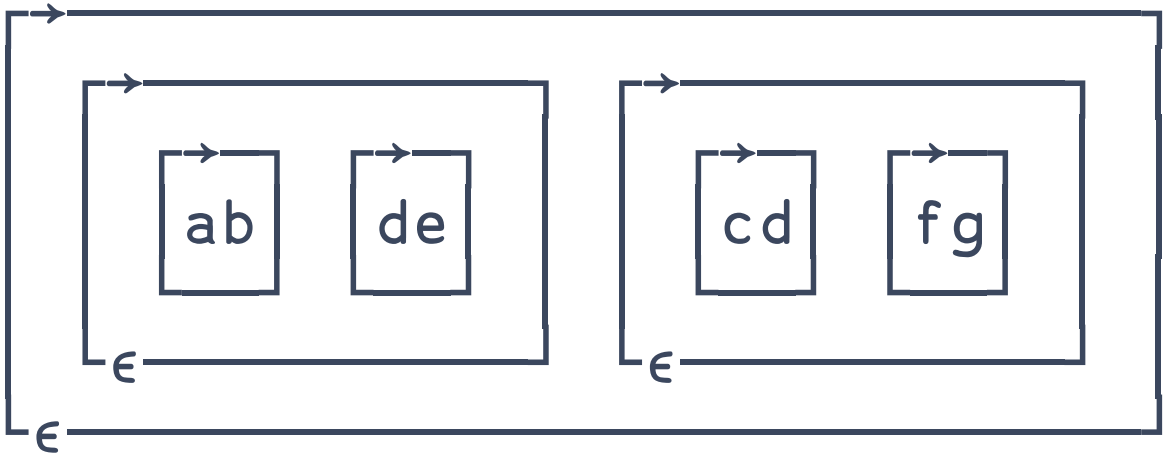


⍠ ⇔ ⎕OPT
is in Classic

⎕CSV

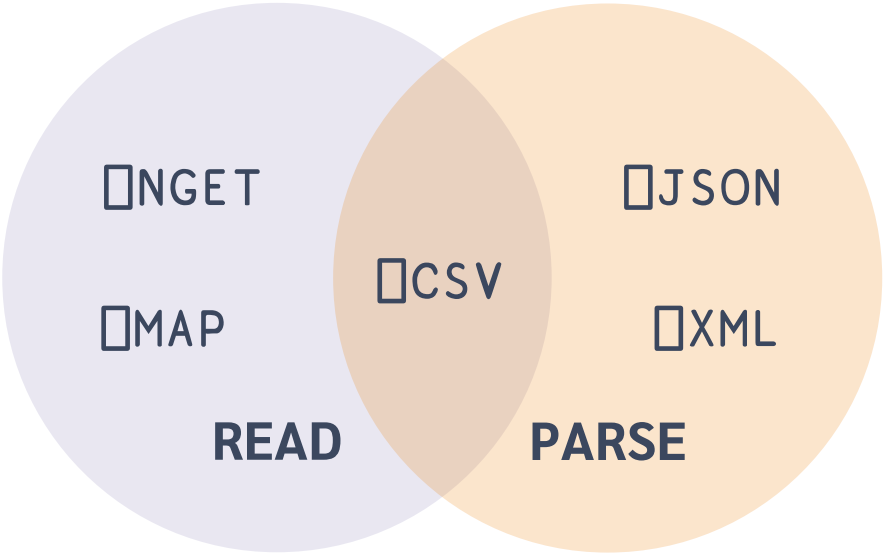
For more techniques, watch is.gd/dyalog_csv

```
⎕CSV⍠'Invert'2⊂'ab,cd' 'de,fg'
```



Reading and Parsing Input

System Functions

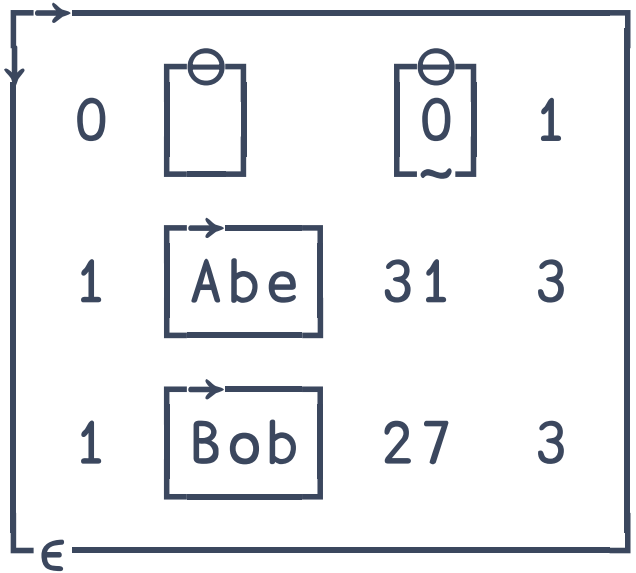


⎕JSON

```
      r←⎕JSON '{"Abe":31,"Bob":27}'
      ]names r
r.Abe r.Bob
      r.Abe
31
```

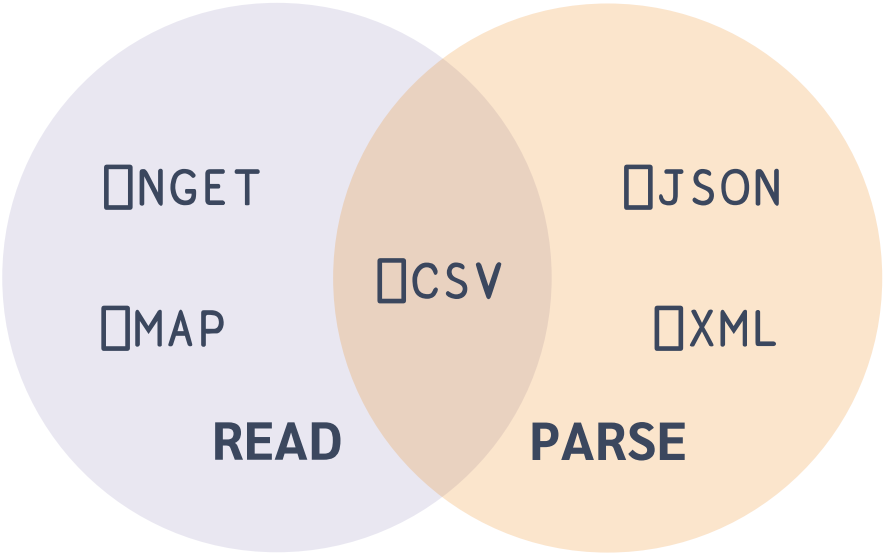
⎕JSON

```
⎕JSON⍠'M'⊢'{"Abe":31,"Bob":27}'
```



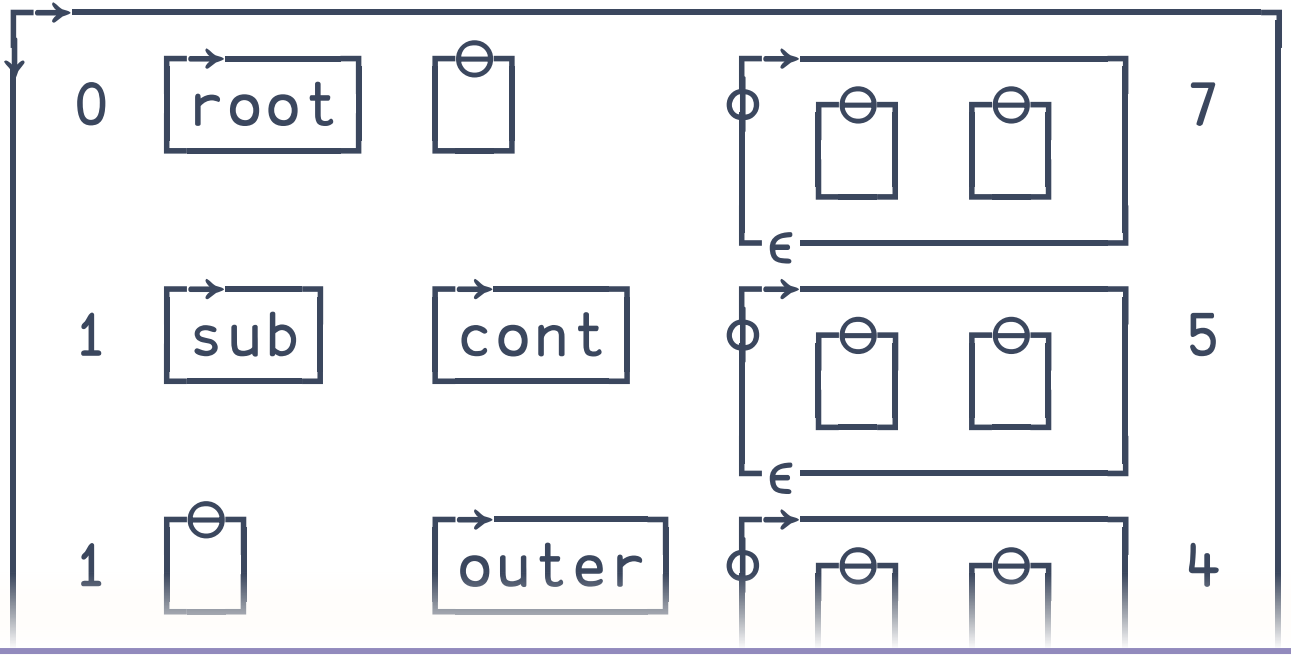
Reading and Parsing Input

System Functions



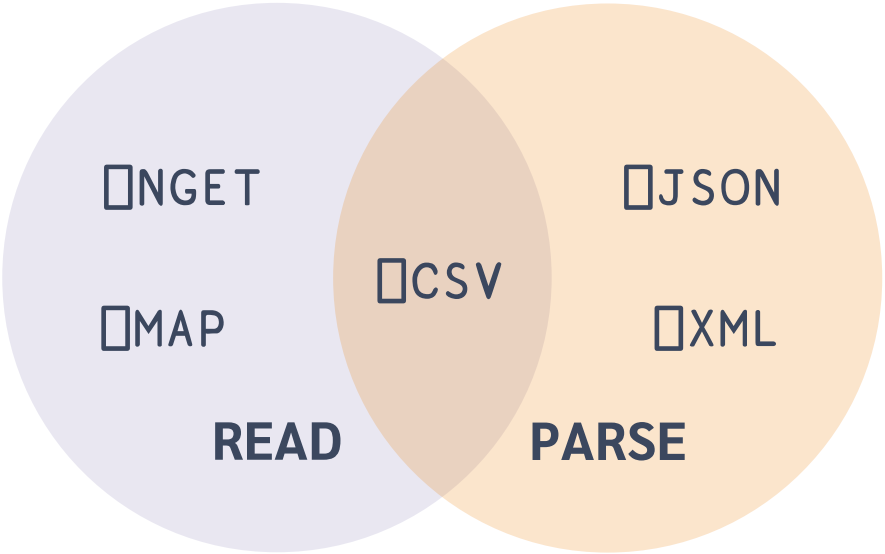
⍰XML

⍰XML '<root>_{cont}outer</root>'



Reading and Parsing Input

System Functions



Reading and Parsing Input

"x"-delimited tuples to numeric matrix (15.2)

```
↑{⎕JSON'[',⍵,']'}¨'x'⎕R','⊃⎕NGET path 1
```

```
↑'x'(0 ⎕JSON¨≠⊆⊢)¨⊃⎕NGET path 1
```

```
⎕CSV⍠'Separator' 'x'⊂path
```

Reading and Parsing Input

Lines of same length to character matrix (15.5, 16.6)

```
↑⊃⎕NGET path 1

⊃⎕CSV⍠'Invert' 1⊢path

{0 ¯1↓⍵⍴⍨n,⍨n÷⍨≢⍵}⊃⎕NGET path

0 ¯1↓80 ¯1 n ⎕MAP path
```

n is line length + 1

Reading and Parsing Input

Fixed-width fields to numeric matrix (16.3)

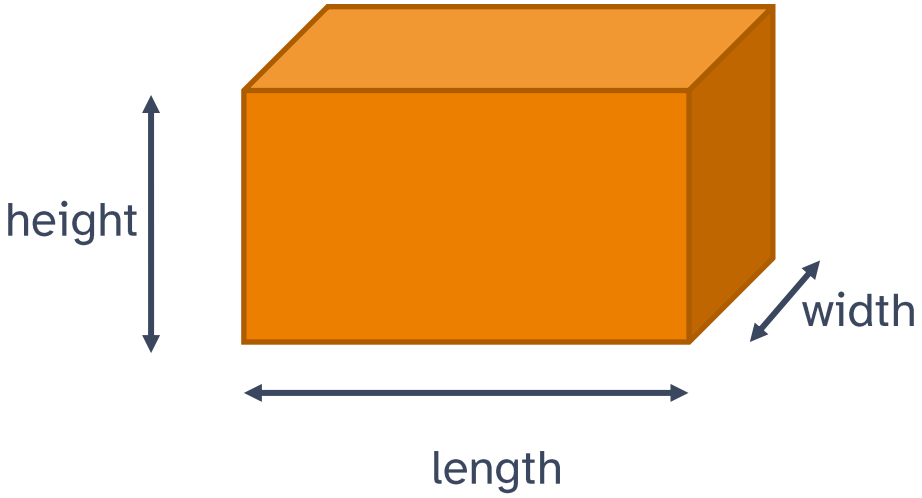
```
↑' '(1 ⎕JSON⍤≠⊆⊢)⍤⊃⎕NGET path 1
```

```
⎕CSV⍠'Widths'(5 5 5)⊢path ' ⍬ 2
```


Mathematical Insights

- 15.2 Wrapping Presents
- 15.17 Filling Containers
- 16.3 Triangles

Wrapping Presents



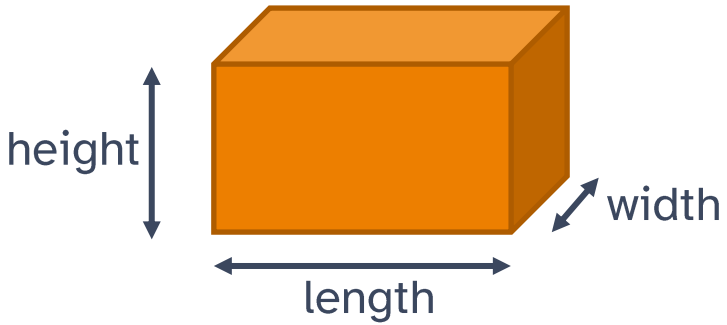
Wrapping Presents

Area of paper required:

Surface area of whole box

+

Area of smallest face

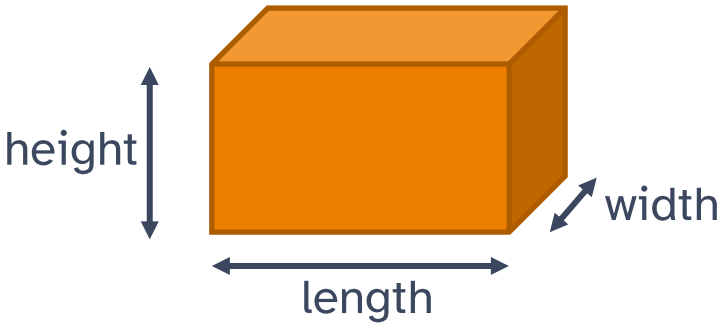


Wrapping Presents

$$(2 \times l \times w) + (2 \times w \times h) + (2 \times h \times l)$$

+

Area of smallest face



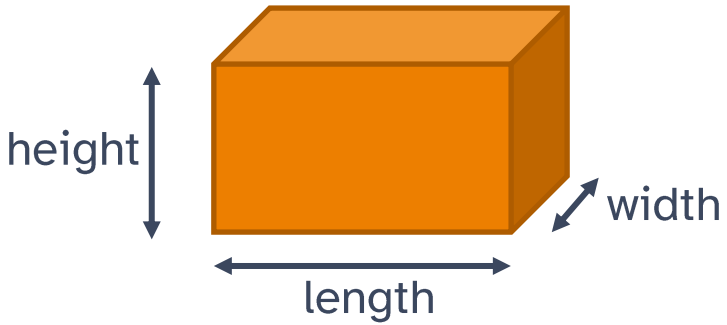
Wrapping Presents

Area of smallest face

- `{⍵[⍋⍵]}`
  - `×/2↑`
  - `×/⍵[1 2]`
- Multiply all 3 sides and divide by largest

  `(×/÷⌈/) l w h`
- Take smallest of products of sides

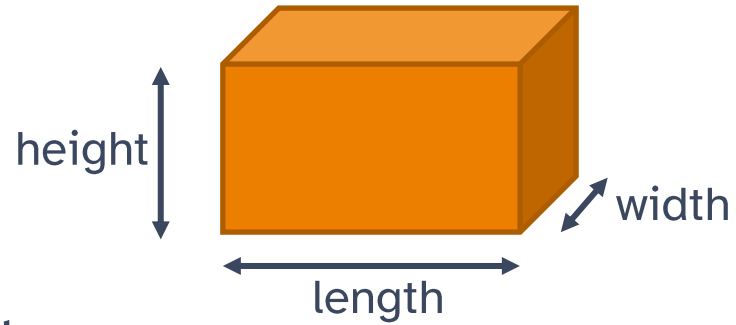
  `⌊/ l w h × w h l`



Wrapping Presents

Area of smallest face

Exercise: Using a similar insight, compute
the perimeter of the smallest face



Multiply all 3 sides and divide by largest

```
(×/÷⌈/) l w h
```

Filling Containers

```
CanFillRecursive←{
      α≤0:α=0
      0∊ρω:0
      +/(α-ω)∇¨(⍳≠ω)↓¨⊂ω
}
```


Filling Containers

Combinations are related to Boolean vectors

When doing base conversion, `⍞←⎕IO←0`

Exercise: Use `⊤` (or `⊥`) to find all Boolean numbers `0` to `⍵`

Filling Containers

Filtered arithmetic

Exercise: What is another way to write:

```
+/α/ω
```

for Boolean vector α and numeric vector ω ?

Filling Containers

Filtered arithmetic

Exercise: How can we filter numeric vector ⍵ using each row of Boolean matrix ⍺ ?

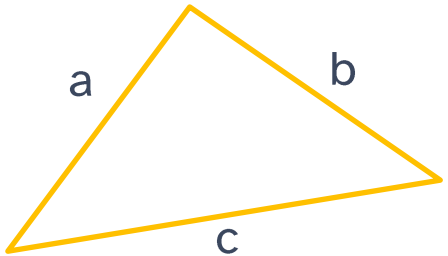
```
      ⍺: 3 5⍴1 0 0 1
1 0 0 1 1
0 0 1 1 0
0 1 1 0 0
      ⍵: 3 4 5 6 7
      ←: ⍝ result need not be nested
```

3 6 7	5 6	4 5

Triangles

Three numbers may define a triangle if

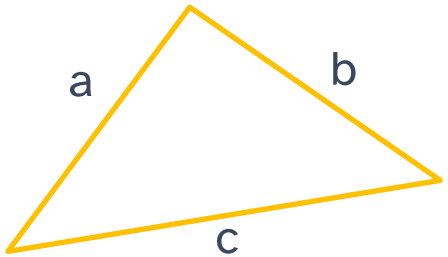
$$a + b \geq c$$



Triangles

Three numbers may define a triangle if

$$a + b \geq c$$



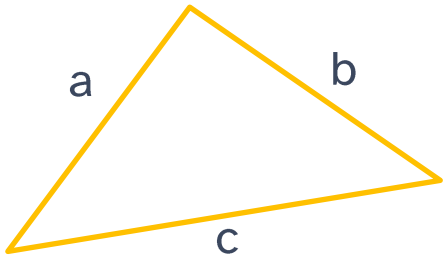
This requires finding out which two sides are shortest.

Can we find a calculation which doesn't depend on conditions?

Triangles

For part 1, each row of 3 numbers defines the sides of a triangle.

For part 2, each vertical group of 3 defines the sides of a triangle.



How does the following code achieve this redefinition?

```
(⍴⍵)⍴,⍉⍵
```


DYALOG

SimCorp

We're here for you!

General support	support@dyalog.com
Forums	forums.dyalog.com
Chat room	apl.chat
Adám Brudzewsky	adam@dyalog.com
Richard Park	rpark@dyalog.com
Rodrigo Girão Serrão	rodrigo@dyalog.com