User-Defined Operators

Adám Brudzewsky

Overview

Why?

What?

How?

Examples

Why define your own operators?

Apply multiple functions in similar ways

Add a parameter to tweak behaviour

Achieve nicer looking expressions

Amend primitives to your needs

Avoid repeating yourself

Why define your own operators?

Apply multiple functions in similar ways

Add a parameter to tweak behaviour

Achieve nicer looking expressions

Amend primitives to your needs

Avoid repeating yourself

... in ways that isn't part of the core language

What is an operator?

MONADIC OPERATOR

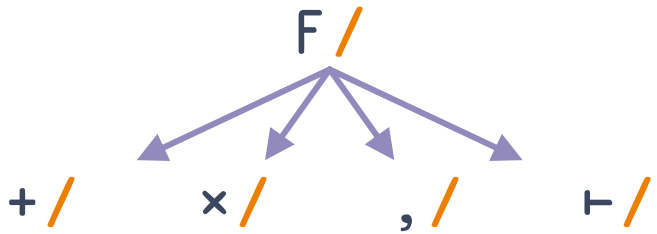
```
F/
```

DYADIC OPERATOR

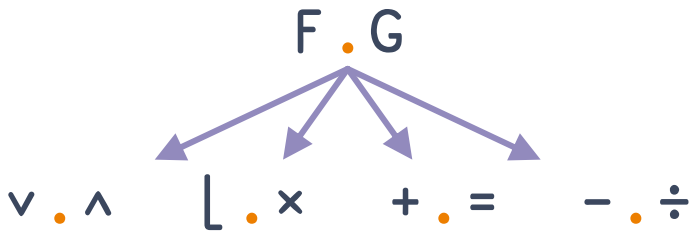
```
F.G
```

What is an operator?

MONADIC OPERATOR

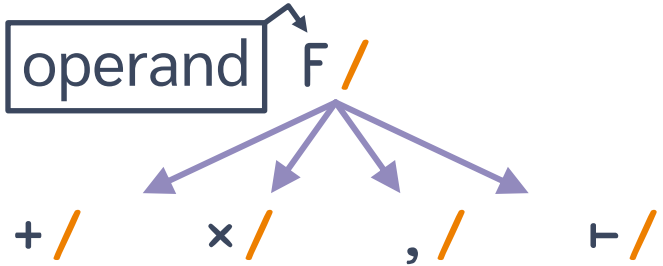


DYADIC OPERATOR

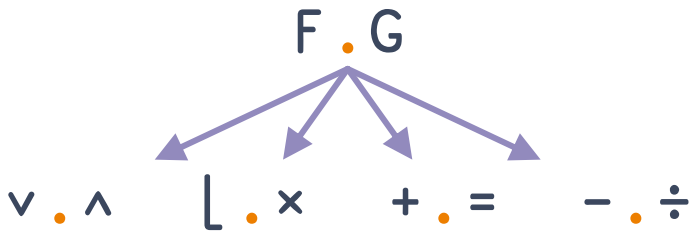


What is an operator?

MONADIC OPERATOR

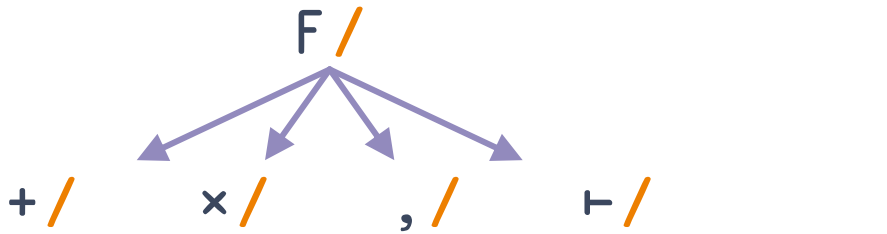


DYADIC OPERATOR

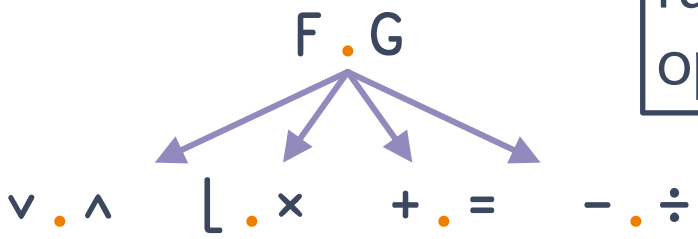


What is an operator?

MONADIC OPERATOR



DYADIC OPERATOR



What is an operator?

MONADIC OPERATOR

`+/` `×⍨` `,\` `⊢⌸`

function operand

DYADIC OPERATOR

`∨.∧` `⌊@×` `+⍤⍟=` `-⍤⌹÷`

What is an operator?

MONADIC OPERATOR

```
42⍨   1200⌶
```

array operand

DYADIC OPERATOR

```
⊢⍤3   1∘+   'Hi'⍤R'Hello'
```

What is an operator?

derived functions

MONADIC OPERATOR

```
42⍨  1200⌶
```

DYADIC OPERATOR

```
⊢⍠3  1∘+  'Hi'⍞R'Hello'
```

What is an operator?

derived functions

MONADIC OPERATOR

```
Answer ← 42⍤
Format ← 1200⍪
```

DYADIC OPERATOR

```
Replace ← 'Hi'⍠R'Hello'
Increment ← 1∘+
Windows ← ⊢⌺3
```

What is an operator?

derived functions

MONADIC OPERATOR

```
Answer ← 42⍨          ⍝ ambivalent
Format ← 1200⌶        ⍝ dyadic
```

DYADIC OPERATOR

```
Replace ← 'Hi'⎕R'Hello'   ⍝ ambivalent
Increment ← 1∘+           ⍝ monadic
Windows ← ⊢⌺3             ⍝ monadic
```

What is an operator?

derived functions

MONADIC OPERATOR

```
Answer ← 42⍤⍤
Format ← 1200I
```

DYADIC OPERATOR

```
Replace ← 'Hi'⎕R'He
Increment ← 1∘+
Windows ← ⊢⌺3
```

⍝ ambivalent

```
      'a' Answer 'b'
42
      Answer 'b'
42
```

⍝ monadic

⍝ monadic

What is an operator?

derived functions

MONADIC OPERATOR

```
Answer ← 42⍨                ⍝ ambivalent
Format ← 1200⌶              ⍝ dyadic
```

```
      Format 123
SYNTAX ERROR: The function requires a left argument
      Format 123
      ∧
```

DYADIC OPER

```
Windows ← ⊢⌺3               ⍝ monadic
```

What is an operator?

derived functions

```
      4 Increment 2
SYNTAX ERROR: The function does not take a left argument
      4 Increment 2
        ^
```

DYADIC OPERATOR

```
Replace ← 'Hi'⍠R'Hello'      ⍝ ambivalent
Increment ← 1∘+              ⍝ monadic
Windows ← ⊢⌺3                ⍝ monadic
```

What is an operator?

OPERATOR	⌜ *deriving a* →	FUNCTION *monadic*	*dyadic*	*ambivalent*
	monadic	`+\`	`1200⌶`	`42⍨`
	dyadic	`⊢⌺3`	`+.×`	`,⍤2`

How are they defined/identified?

TRADFN

Calling syntax in header

```
∇ res←x Name y
  res←x y
∇
```

DFN

Presence of ⍺ and ⍵ in body

```
Name←{
     ⍺ ⍵
}
```

dyadic function

How are they defined/identified?

TRADOP

Calling syntax in header

```
∇ res←(F Name)y
  res←F y
∇
```

DOP

Presence of ⍺⍺ or ⍵⍵ in body

```
Name←{
     ⍺⍺ ⍵
}
```

monadic operator deriving a monadic function

How are they defined/identified?

TRADOP

Calling syntax in header

```
∇ res←x(F Name)y
  res←x F y
∇
```

DOP

Presence of ⍺⍺ or ⍵⍵ in body

```
Name←{
     ⍺ ⍺⍺ ⍵
}
```

monadic operator deriving a dyadic function

How are they defined/identified?

TRADOP

Calling syntax in header

```
∇ res←x(F Name G)y
  res←x F G y
∇
```

DOP

Presence of ⍺⍺ or ⍵⍵ in body

```
Name←{
     ⍺ ⍺⍺ ⍵⍵ ⍵
}
```

dyadic operator deriving a dyadic function

How are they defined/identified?

TRADOP

Calling syntax in header

```
∇ res←(F Name G)y
  res←F G y
∇
```

DOP

Presence of ⍺⍺ or ⍵⍵ in body

```
Name←{
     ⍺⍺ ⍵⍵ ⍵
}
```

dyadic operator deriving a monadic function

Apply multiple functions in similar ways

Add a parameter to tweak behaviour

Achieve nicer looking expressions

Amend primitives to your needs

Avoid repeating yourself

Demo: utilities

```
_S←{α←⊢ ⋄ ↑⍵(α ⍺⍺ ⍵)}  ⍝ Stack
Vowel←∈∘'AEIOU'
Vowel _S □A
'AEIOU'⍳_S□A

_T←{α←⊢ ⋄ r←⍺ ⍺⍺ ⍵ ⋄ □←⍺(62□ATX '⍺⍺')⍵ '→'r ⋄ r}  ⍝ Trace
-_T/3 1 4 1 5
```

Demo: combinators/compositions

```
A_←{α(⍵⍵ ⍺⍺ ⍵⍵⍨⍨)⍵}  ⍝ Across
'HELLO'∪_A_~'APL'

_H_←{(⍺⍺ ⍵)⍵⍵ ⍵}  ⍝ Hook
⌽_H_≡¨'hello' 'racecar' 'APL' 'ABBA' 'max'
⌽_H_≡¨_S'hello' 'racecar' 'APL' 'ABBA' 'max'
⌽_H_≡¨_H_/'hello' 'racecar' 'APL' 'ABBA' 'max'
```

Demo: thought concepts

```
⍝ param Fn _W_ Cond initArg
_W_←{α←⊢ ⋄ ⍵⍵ ⍵:α ∇ α ⍺⍺ ⍵ ⋄ ⍵}  ⍝ While
2×_W_{⍵<100}4
2×_T _W_{⍵<100}4

_E_←{α←⊢ ⋄ 0::α ⍵⍵ ⍵ ⋄ α ⍺⍺ ⍵}  ⍝ ErrorElse

_D←{r←⍵ ⋄ (∊r)←⍺⍺¨∊r ⋄ r}  ⍝ Depth D
(×/⍳)_D 4 5(2 3)
```

Next Webinar

June 23: TBA

Remember: BAA webinars every other week
britishaplassociation.org/webinar-schedule-2022
June 16th, 30th; July 14th, 28th; etc.

More at: apl.wiki/activities