Programação Vetorial Tipos primitivos de dados e funções

Prof. Edson Alves

Faculdade UnB Gama

Sumário

1. Tipos primitivos de dados

2. Arrays

Expressões em APL

► APL pode ser vista como uma notação matemática que também é executável por máquina

Expressões em APL

 APL pode ser vista como uma notação matemática que também é executável por máquina

A linguagem é composta por funções, operadores, arrays e atribuições

Expressões em APL

- APL pode ser vista como uma notação matemática que também é executável por máquina
- A linguagem é composta por funções, operadores, arrays e atribuições
- Qualquer código que pode ser aplicado a dados é chamado função

Expressões em APL

 APL pode ser vista como uma notação matemática que também é executável por máquina

- A linguagem é composta por funções, operadores, arrays e atribuições
- Qualquer código que pode ser aplicado a dados é chamado função
- Dois exemplos de funções seriam a adição (+) e subtração (-)

Expressões em APL

 APL pode ser vista como uma notação matemática que também é executável por máquina

- A linguagem é composta por funções, operadores, *arrays* e atribuições
- Qualquer código que pode ser aplicado a dados é chamado função
- Dois exemplos de funções seriam a adição (+) e subtração (-)
- As funções de APL podem ser aplicadas monadicamente (prefixada, um operando) ou diadicamente (infixada, dois operando, um à esquerda e outro à direita)

Expressões em APL

 APL pode ser vista como uma notação matemática que também é executável por máquina

- A linguagem é composta por funções, operadores, *arrays* e atribuições
- Qualquer código que pode ser aplicado a dados é chamado função
- Dois exemplos de funções seriam a adição (+) e subtração (-)
- As funções de APL podem ser aplicadas monadicamente (prefixada, um operando) ou diadicamente (infixada, dois operando, um à esquerda e outro à direita)
- O tipo de dados mais elementar é o escalar (array de dimensão zero)

Inteiros

Números são tratados internamente pela APL quanto ao tamanho e tipo e podem ser misturados sem problemas

Inteiros

5

6

Números são tratados internamente pela APL quanto ao tamanho e tipo e podem ser misturados sem problemas

- ► Em APL os números podem ser inteiros, reais (em ponto flutuante) e números complexos
- ▶ Um escalar inteiro pode ser grafado usando a notação decimal padrão:

```
2 + 3
2 × 3 A a multiplição é realizada pela função ×
```

Inteiros

5

6

Números são tratados internamente pela APL quanto ao tamanho e tipo e podem ser misturados sem problemas

- ► Em APL os números podem ser inteiros, reais (em ponto flutuante) e números complexos
- ▶ Um escalar inteiro pode ser grafado usando a notação decimal padrão:

```
2 + 3
2 × 3 A a multiplição é realizada pela função ×
```

Comentários são precedidos pelo símbolo A

Inteiros

- Números são tratados internamente pela APL quanto ao tamanho e tipo e podem ser misturados sem problemas
- ► Em APL os números podem ser inteiros, reais (em ponto flutuante) e números complexos
- Um escalar inteiro pode ser grafado usando a notação decimal padrão:

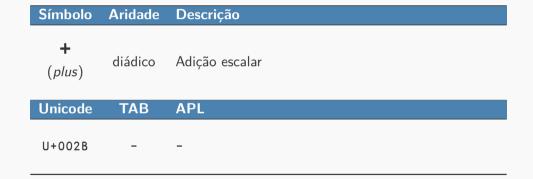
```
2 + 3
2 × 3 A a multiplição é realizada pela função ×
```

- Comentários são precedidos pelo símbolo A
- ▶ Números negativos são precedidos pelo símbolo (macron)

5

6

Novo símbolo



Novo símbolo

| Símbolo | Aridade | Descrição |
|--------------|---------|-------------------|
| — (minus) | diádico | Subtração escalar |
| Unicode | TAB | APL |
| U+002D | - | - |

Novo símbolo

| Símbolo | Aridade | Descrição |
|--------------|-----------------|-----------------------|
| × (times) | diádico | Multiplicação escalar |
| Unicode | TAB | APL |
| U+00D7 | x x <tab></tab> | APL + - |

Novo símbolo

| Símbolo | Aridade | Descrição |
|----------|-------------|----------------------------|
| (macron) | monádico | Prefixa um número negativo |
| Unicode | TAB | APL |
| U+00AF | <tab></tab> | APL + 2 |

Novo símbolo

| Símbolo | Aridade | Descrição |
|----------------|-----------------|--|
| A (comment) | monádico | Inicia um comentário. Tudo que o sucede até o fim da linha será considerado comentário |
| Unicode | TAB | APL |
| U+235D | o n <tab></tab> | APL + , |

Arrays

Números reais

► Em escalares reais, a parte inteira é separada das casas decimais por meio do ponto final

Números reais

Em escalares reais, a parte inteira é separada das casas decimais por meio do ponto final

```
0.2 ÷ 3.5
0.05714285714
```

▶ APL também trata problemas de precisão de forma transparente ao usuário

```
2÷3 ◊ 6 × (2 ÷ 3)
0.6666666667
4
```

Números reais

Em escalares reais, a parte inteira é separada das casas decimais por meio do ponto final

```
0.2 ÷ 3.5 0.05714285714
```

▶ APL também trata problemas de precisão de forma transparente ao usuário

```
2÷3 	 6 × (2 ÷ 3)
0.6666666667
4
```

O símbolo (diamond) separa duas expressões em uma mesma linha

Números reais

Em escalares reais, a parte inteira é separada das casas decimais por meio do ponto final

```
0.2 ÷ 3.5 0.05714285714
```

APL também trata problemas de precisão de forma transparente ao usuário

```
2÷3 	 6 × (2 ÷ 3)
0.6666666667
```

- ▶ O símbolo ◊ (diamond) separa duas expressões em uma mesma linha
- Notação científica pode representar números muito pequenos ou grandes

```
2E<sup>-</sup>3 ♦ 5e7 A O E pode ser maiúsculo ou minúsculo
0.002
50000000
```

Novo símbolo

| Símbolo | Aridade | Descrição |
|---------|-----------------|--|
| divide) | diádico | Divisão escalar. Divisão por zero resulta em um erro |
| Unicode | TAB | APL |
| U+00F7 | : - <tab></tab> | APL + = |

Novo símbolo



Constantes booleanas

► Em APL: falso é igual a 0 (zero) e verdadeiro é igual a 1 (um)

Constantes booleanas

► Em APL: falso é igual a 0 (zero) e verdadeiro é igual a 1 (um)

```
2 = 3
5 = 5.0
```

▶ Os operadores relacionais retornam valores booleanos

```
2 ≠ 3
0
    5 < 7
    11 > 13
    17 ≤ 19 ♦ 23 ≥ 27
```

Programação Vetorial Prof Edson Alves

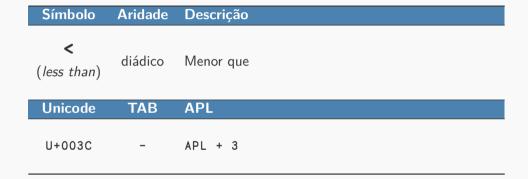
Novo símbolo

| Símbolo | Aridade | Descrição |
|--------------|---------|-----------|
| = (equal) | diádico | lgual a |
| Unicode | TAB | APL |
| U+003D | - | APL + 5 |

Novo símbolo

| Símbolo | Aridade | Descrição |
|----------------------|-----------------|--------------|
| ≠ (not equal) | diádico | Diferente de |
| Unicode | TAB | APL |
| U+2260 | = / <tab></tab> | APL + 8 |

Novo símbolo



Novo símbolo

| Simbolo | Aridade | Descrição |
|------------------|---------|-----------|
| > (greater than) | diádico | Maior que |
| Unicode | TAB | APL |
| U+003E | - | APL + 7 |

Novo símbolo

| Símbolo | Aridade | Descrição |
|----------------------------------|-----------------|------------------|
| ≤ (less than or equal to) | diádico | Menor ou igual a |
| Unicode | TAB | APL |
| U+2264 | < = <tab></tab> | APL + 4 |

Novo símbolo

| | Símbolo | Aridade | Descrição |
|---------------------------------|---------|-----------------|------------------|
| ≥ (greater than or equal to) | | diádico | Maior ou igual a |
| | Unicode | TAB | APL |
| | U+2265 | > = <tab></tab> | APL + 6 |

Números complexos

O caractere 'J' separa a parte real da parte imaginária em números complexos

$$2J3 \times 5j^{-7}$$
 A O J também pode ser minúsculo 31J1

Números complexos

▶ O caractere 'J' separa a parte real da parte imaginária em números complexos

```
2J3 \times 5j^{-7} A O J também pode ser minúsculo 31J1
```

► Lembre-se de que o argumento à direita de uma função diádica é o resultado de toda a expressão à direita do símbolo

Novo símbolo

| Símbolo | Aridade | Descrição |
|------------------|---------|--|
| ★ (power) | diádico | Eleva o argumento à esquerda a potência indicada no argumento à direita |
| Unicode | TAB | APL |
| U+002A | - | APL + p |

Caracteres e strings

► Em APL, strings são vetores de caracteres

Caracteres e strings

- ► Em APL, strings são vetores de caracteres
- ► Tanto caracteres quanto strings são delimitadas por aspas simples

```
'c' A um caractere
c
'uma string'
uma string
```

Caracteres e strings

- ► Em APL, strings são vetores de caracteres
- Tanto caracteres quanto strings são delimitadas por aspas simples

```
'c' A um caractere
c
'uma string'
uma string
```

Atribuições podem ser feitas por meio do símbolo +

Novo símbolo

| Símbolo | Aridade | Descrição |
|---------------|-----------------|--|
| ← (assign) | diádico | Atribui o argumento à direta ao argumento à esquerda |
| Unicode | TAB | APL |
| U+2190 | < - <tab></tab> | APL + ' |

Funções aritméticas monádicas

As funções aritméticas apresentadas até o momento tem versões monádicas

Funções aritméticas monádicas

As funções aritméticas apresentadas até o momento tem versões monádicas

 Quando aplicada a números reais, a função monádica × corresponde à função signum() de muitas linguagens

Novo símbolo



Novo símbolo

| Símbolo | Aridade | Descrição |
|---------------|----------|-------------------|
| - (negate) | monádico | Simétrico aditivo |
| Unicode | TAB | APL |
| U+002D | - | - |

Novo símbolo

| Símbolo | Aridade | Descrição |
|---------------|-----------------|-------------------------------------|
| × (direction) | monádico | Vetor unitário na direção do número |
| Unicode | TAB | APL |
| U+00D7 | x x <tab></tab> | APL + - |

Novo símbolo



Novo símbolo

| Símbolo | Aridade | Descrição |
|------------------------|----------|----------------------------------|
| ★ (exponential) | monádico | e elevado ao argumento à direita |
| Unicode | TAB | APL |
| U+002A | - | APL + p |

Arrays Arrays

Referências

- 1. APL Wiki. Quad name, acesso em 23/09/2021.
- 2. **BROCKLEBANK**, Daniel. *APL The Language*, John Hopkins APL Technical Digest, vol. 5, number 3, 1984.
- 3. Dyalog. Dyalog APL Version 12.0, acesso em 23/09/2021.
- 4. Dyalog. Try APL, acesso em 23/09/2021.
- 5. IVERSON, Kenneth E. A Programming Language, John Wiley and Sons, 1962.
- **6.** Stack Overflow. Add APL Keyboard Layout On Linux 20.04, acesso em 23/09/2021.
- **7. ULMANN**, Bernd. *APL One of the Greatest Programming Languages Ever*, Vintage Computer Festival Europe 2007.
- 8. Xah Lee. Unicode APL Symbols, acesso em 23/09/2021.
- 9. Wikipédia. APL (programming language), acesso em 24/09/2021.
- 10. Wikipédia. Array programming, acesso em 24/09/2021.

Prof. Edson Alves Programação Vetorial