UFG

Introdução à Linguagem de Programação Julia

Gracielle, Karlo & Hélio

Goiânia, abril de 2019

Revisando...

Variáveis simples:

- ✓ Inteiro: Int64
- ✓ Real: Float64
- ✓ Verdadeiro/Falso: Bool # true/false
- ✓ Caracteres: Char

- ✓ Não precisa declarar variável
- ✓ Julia é case sensitive: a ≠ A

Mais sobre Arrays

- ✓ Criando e preenchendo com valores

```
julia> A = zeros(7,3)

julia> B = ones(4,6)

julia> C = trues(2,4)

julia> D = falses(5,3)

julia> E = fill(NaN,4,3)

julia> F = fill(7,4,3)
```

Mais sobre Arrays

```
julia> G = rand(8,6,3)

julia> H = rand(10:20,6,3)
```

✓ Operador ellipsis (para vetores: 1 dimensão)

```
julia> a = [1:5...]

julia> b = [1:2:9...]

julia> c = [1.2:2.2:15.0...]
```

Propriedades dos Arrays

- ✓ Tamanho do vetor

```
julia> length(c)
```

- ✓ Dimensões da matriz

```
julia> size(A)
julia> (n,m) = size(A) # guarda valores
```

Arrays

- ✓ Encontrar itens no Array

```
julia> 12 in H
julia> in(15,H)
```

- ✓ Listando operações com elementos

```
julia> [b[i]^2 for i=1:length(b)]
julia> [n^2 for n in b]
```

Array

- ✓ Repetir elementos do Array

Repete elemento a elemento:

```
julia> repeat(b,inner=2)
```

Repete todos os elementos:

```
julia> repeat(b,outer=2)
```

Strings

- ✓ Char (caracter)

```
julia> a1 = 'a'
```

- ✓ String

```
julia> a2 = "a"
```

```
julia> typeof(a1)
julia> typeof(a2)
```

```
julia> a3 = 'blablabla'
```

String

- ✓ Concatenação

```
julia> string("Tudo ", "junto")

julia> a4 = "Hello "

julia> a5 = "Hello "

julia> string(a4,a5) # 1ª forma

julia> a4*a5 # 2ª forma

julia> *(a4,a5) # 3ª forma
```

Interpolação

```
julia> total = 250.0
julia> "A soma é $total."

julia> d = rand(1:9,8)
julia> ["Número $i" for i in d]

# no script
for i in d
     println("Número $i")
end
```

Interpolação

```
julia> a6 = "Olá"
julia> a7 = "mundo"
julia> esp = " "
julia> "$a6$esp$a6“
```

E quando queremos imprimir $?

```
julia> saldo = "100"
julia> print("Tenho \$$saldo em conta.")
```

String como Array

```
julia> str = "Você foi avisado!"

julia> str[end]

julia> str[1:4]

julia> str[10:end]

julia> length(str)
```

Funções de String

```
julia> split(str)

julia> repeat("Oi ",3)
julia> "Oi "^3
julia> ^("Oi ",3)

julia> uppercase("Machado de Assis")
julia> lowercase("Machado de Assis")
```

Funções de String

```
julia> reverse("Bentinho")


# elimina espaços no início e fim
julia> lstrip(" Machado de Assis ")
julia> rstrip(" Machado de Assis ")
julia> strip(" Machado de Assis ")
```

Comparações lexicográficas

```
julia> "Lucas" < "Luan"

julia> "g" < "G"

julia> "13 April" < "13 May"

julia> "super" == "Super"
```

Funções

- ✓ Função hipotenusa: $h^2 = a^2 + b^2$

```
# no script salve como "hipotenusa.jl"
function hipotenusa(a,b)
        h = sqrt(a^2 + b^2)
        return h
end
# Run ou no console: include("hipotenusa.jl")
# no console:
julia> hipotenusa(3,4)
```

Funções

- ✓ **Função Fibonacci:** imprime o n-ésimo elemento da sequência de Fibonacci

- ✓ Sequência de números inteiros:
  $F_n = F_{n-1} + F_{n-2}$, com $F_0 = 0$, $F_1 = 1$

- ✓ 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89...

✓ Função Fibonacci

```
# no script salve como “fibonacci.jl"
function fibo(n)
        if n < 2
                return n
        else
                return fibo(n-1) + fibo(n-2)
        end
end
# Run ou no console: include("fibonacci.jl")
# no console
julia> fibo(10)
julia> @time fibo(40)
```

Desafio: como imprimir os 20 primeiros termos de Fibonacci?

Gráficos básicos

- ✓ Adicione os pacotes no console:

```
julia> Pkg.add("Plots")
julia> Pkg.add("GR")

julia> using Plots
```

Gráficos básicos

- ✓ **Exemplo 1**

```
> plot(rand(5,5), linewidth=2.0, title="Aleatório")

> savefig("figura1.pdf") # só no plotpane

> plotly() # web backend
> gr() # plotpane backend
```

Gráficos básicos

- ✓ **Exemplo 2**

```
> x = range(0, stop=2*pi, length=1000);

> y = sin.(3*x+4*cos.(2*x));

> plot(x, y, color="red", linewidth=3.0,
linestyle=:solid)
```

- ✓ linestyle = :auto, :solid, :dash, :dot, :dashdot, :dashdotdot

Gráficos básicos

- ✓ **Exemplo 3**

```
> x = range(0, stop=2*pi, length=1000);

> y = sin.(3*x);

> plot(x, y, color="green", linewidth=2.0,
linestyle=:dash, title="Gráfico 1",
xlabel="valor de x", ylabel="seno 3x",
label="função 1")
```

Gráficos básicos

- ✓ **Exemplo 4 – Histograma**

```
> x = randn(1000);
> histogram(x)
```

Referência

- Nagar, S. Beginning Julia Programming for Engineers and Scientists. New York: Apress, 2017.

Desafios finais:

1. Crie um Array com os 5 principais personagens do seu seriado ou filme favorito.
2. Crie um Array com os 5 primeiros números da sequência de Fibonacci.
3. Crie um vetor misto com 5 elementos, utilizando elementos dos vetores anteriores.
4. Dado x = -10:10, plote y vs. x para $y=x^2$
5. Execute os comandos:

```
p1 = plot(x, x)
p2 = plot(x, x.^2)
p3 = plot(x, x.^3)
p4 = plot(x, x.^4)
plot(p1, p2, p3, p4, layout = (2, 2), legend = false)
```

   1. Agora crie um plot 4x1 com p1, p2, p3 e p4