

```
echo "Olá mundo!"
```

[Hello world](#)

Nim é uma linguagem eficiente, expressiva, e elegante, criada por Andreas Rumpf e lançada em 2008. É uma linguagem de programação compilada e estaticamente tipada. Nim combina conceitos de linguagens como Python, Ada, e Modula.

```
42, -4      # int - Inteiro
42, 4       # uint - Inteiro (não negativo)
true, false # bool - Booleano
"42"        # string - String
'a', '4'    # char - Caractere
42.0        # float - Flutuante
```

[Tipo Básico](#)

```
# Este é um comentário
echo "Olá!" # outro comentário

#[ Este é um comentário
Multilinha
]#
```

[Comentários](#)

```
var minhaVar1:string = "Minha variável" # Variável declarada como string.
var minhaVar2 = "Minha variável" # Variável declarada implicitamente como string.
let minhaConstante = "Valor constante" # Constante. Valor imutável.
var salario:float = 9_000.00 # Variável float.
                                # _ é ignorado. Útil para visualizar números grandes.
const PI = 3.14 # Constante. Valor imutável. Atribuição em tempo de compilação.
```

[Variável](#) e [Constante](#)

```
import os # Importa uma lib.
import strutils
import json, outmod # Importa lib e outro módulo.
include "pasta/arq.nim" # Inclui arquivo. Uso: dividir
include "arq1.nimf".    # um arquivo grande em partes.
```

[Import](#) e [Include](#)

```
5 + 2 # Adição
5 - 2 # Subtração
5 * 2 # Multiplicação
5 / 2 # Divisão
# Divisão inteira # Resto da divisão inteira
5 div 2 # retorna: 2 5 mod 2 # retorna: 1
```

[Operadores](#)

```
# Table.
# Estrutura tipo dicionário.
import tables
var anoDeNascimento: Table[string, uint]
anoDeNascimento["Mike"] = 1995
anoDeNascimento["John"] = 1961
echo anoDeNascimento["Mike"]
1995
# Array.
# Estrutura tipo vetor ou matriz.
var meuArray: array[0 .. 2, bool] = [false, true, false]
echo meuArray[1]
true
```

[Tipo Avançado](#)
[1/2](#)

```
# Sequence.
# Estrutura tipo lista de valores.

let minhaSequencia: seq[string] = @["abc", "def"]
echo minhaSequencia[1]
def
# Tuple.
# Útil para retornar vários valores de uma
# func ou proc.
let pos: tuple[x, y, z: int] = (x: 100, y: 50, z: 30)
echo pos          echo pos.x      echo pos[1]
(x: 100, y: 50, z: 30) 100          50

# Tuple também suporta unpacking:
let (a,b,_) = pos # Aqui criamos as variáveis a, b.
                  # a = 100, b = 50.
                  # A variável z foi desprezada.
```

[Tipo Avançado](#)
[2/2](#)

```
# Concatenação
echo "con" & "catena"
concatena

# Aspas dentro de aspas
echo """Nim e' uma linguagem "expressiva"""
Nim e' uma linguagem "expressiva"
```

[Operações com](#)
[String 1/2](#)

```
# Interpolação de strings
import strformat
let versaoNim = $nimVersion
echo fmt"""Nim versao em uso: {versaoNim}"""
Estou usando Nim versao: 1.6.8
```

```
echo fmt"""Resultado de 4 x 2 = {4*2}"""
Resultado de 4 x 2 = 8
```

```
# Remover parte de uma string
echo "Nim-lang".strip[0 .. 2]
Nim
```

```
# String multilinha
let multilinha = """linha1
linha2"""
echo multilinha
linha1
linha2
```

[Operações com](#)
[String 2/2](#)

```
import strutils
echo "A_B_C".normalize()
abc

echo "A_B_C".normalize().toUpper()
ABC

echo " ABC ".strip().len()
3

echo len(" ABC ") == " ABC ".len()
true

echo "ABC".toLowerCase()
abc

echo "a b c".split()
@["a", "b", "c"]

echo ["a", "b", "c"].join(",")
a,b,c

echo "".repeat(20)
*****

echo "  a".replace(" ", " ")
***a
```