Nim 1.6 Cheat Sheet versão 1.2

https://nim-lang.org

Hello World

```
echo "Olá, mundo!"
```

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.

Tipo Básico

```
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
```

Comentários

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

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

Variável e Constante

```
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.
```

Import e Include

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

Operadores

```
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
```

Tipo Avançado 1/2

```
# 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 2/2

```
# 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.
```

Operações com String 1/2

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

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

# 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".toLower()
abc

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

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

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

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

Operador Booleano

```
# and
echo true and true  # retorna true somente se os
true                # dois valores são verdadeiros.
# or
echo true or false  # retorna true se pelos menos um
true                # dos dois valores é verdadeiro.
# xor
echo false xor true # retorna true se somente um
true                # dos dois valores é verdadeiro.
# not
echo not false      # retorna o inverso do valor.
true
```

Operador Relacional

```
echo 5 < 4      echo 5 <= 4     echo 5 == 4
false           false           false

echo 5 != 4     echo 5 > 4      echo 5 >= 4
true            true            true
```

Entrada de dados pelo teclado

```
echo "Entre com seu nome: "
let nome = stdin.readLine()
echo "\nOlá ",nome
Olá Sergio
```

Declaração If

```
echo "Entre com seu nome: "
let nome = stdin.readLine()
if nome == "":
  echo "Você esqueceu seu nome?"
elif nome == "nome":
  echo "Entre com seu nome!"
else:
  echo "Olá, ",nome,"!"
```

Operador Ternário

```
import random
randomize()
let n = rand(10)
let parImpar = n mod 2
echo "O numero ",n," e' "
echo if parImpar == 0: "par" else: "impar" # if ternário
```

Declaração Case

```
from std/strutils import parseInt
let validNumber = "Entre um numero de 0 a 9"
echo validNumber
let n = parseInt(readLine(stdin))
case n
  of 0..2, 4..7: echo n," faz parte de:{0,1,2,4,5,6,7}"
  of 3, 8: echo "O numero e' 3 ou 8"
  else: echo validNumber
```

Declaração For

```
echo "Contando de 1 a 5"
for i in countup(1, 5):
  stdout.write(i," ")
1 2 3 4 5
echo ""
for i in countup(1, 5): stdout.write(i," ")
1 2 3 4 5
echo ""
for i in 1 .. 5: stdout.write(i," ")
1 2 3 4 5
echo ""
for i in countup(1, 5, 2): stdout.write(i," ")
1 3 5
echo "\nContando de 2 a -2"
for i in countdown(2, -2): stdout.write(i," ")
2 1 0 -1 -2
echo "\nIterar através de cada caractere da string"
let word = "Nim Lang"
for c in word: stdout.write(c)
Nim Lang
```

Declaração While

```
var a = 1
while a*a < 10:
  stdout.write(" a = ",a)
  a = a + 1
echo "\nValor final de a = ",a
a = 1 a = 2 a = 3
Valor final de a = 4

let frase = "\nEntre seu nome. <ENTER> para sair:"
echo frase
var nome = readLine(stdin)
while nome != "":
  echo "Ola, ",nome
  echo frase
  nome = readLine(stdin)
while true: # while usando break
  echo frase
  nome = readLine(stdin)
  if nome == "": break # break para sair do while
  echo "Ola, ",nome
```

Procedure 1/3

```
# Retorna o valor da última expressão
proc olaMundo(): string =
  "Olá, mundo!"

echo olaMundo()
Olá, mundo!

# return: retorna valor da procedure
proc sinalNum(n: int): int =
  if n == 0: return n
  return if n > 0: 1 else: -1

stdout.write(sinalNum(-10),",",sinalNum(2),",",sinalNum(0))
-1, 1, 0
```

Procedure 2/3

```
# result é uma variável
# declarada implicitamente

proc dizOla(nome: string): string =
  result = "Olá, "
  if nome == "":
    result = "Qual é seu nome mesmo?"
  else:
     result &= nome

echo dizOla("Sergio")
Olá, Sergio
echo dizOla("")
Qual é seu nome mesmo?
```

Procedure 3/3

```
# Declare um argumento como var para permitir mudar seu valor dentro da procedure
proc isPossibleDivision(dividend: int, divisor: int, quotient: var float, remainder: var int): bool =
  if divisor == 0: return false
  quotient = dividend / divisor
  remainder = dividend mod divisor
  return true

var quotient: float = 0.0
var remainder: int = 0
stdout.write(isPossibleDivision(5, 2, quotient, remainder), ", ", quotient, ", ", remainder)
true, 2.5, 1

# Use uma Tuple (ou Seq) para retornar múltiplos valores de uma procedure
var divItems = (isPossible: false, quotient: 0.0, remainder: 0)
proc isPossibleDivision(dividend: int, divisor: int): (bool, float, int) =
  if divisor == 0: return (false, 0.00, 0)
  return (true, dividend / divisor, dividend mod divisor)

divItems = isPossibleDivision(5, 2)
stdout.write(divItems.isPossible, " ,", divItems.quotient, " ,", divItems.remainder)
true, 2.5, 1
```