

## TEMA: GEOMETRIA

## SUBTEMAS: TRIGONOMETRIA. PRODUTO ESCALAR

Tarefa “Geometria em Python”

## PARTE 1 – Memorando de linguagem Python

## VARIÁVEIS

- Distinguem-se dois tipos de números: inteiros (**int**) e decimais (**float**).
- Atribuição de um número inteiro: **n = 1** → o número 1 é atribuído à variável **n** (tipo **int**)
- Atribuição de um número real: **x = 0.5** → o número 0.5 é atribuído à variável **x** (tipo **float**)

## LEITURA E ESCRITA

- Leitura de um número inteiro: **n = int(input("n=?"))**.
- Leitura de um número real: **x = float(input("x=?"))**.
- Leitura de texto: **t = input("mensagem")**.
- Escrita de texto: **print("texto")** → escreve **texto** no Shell e muda de linha.
- Escrita de um valor numérico: **print(x)** → escreve o valor de **x** no Shell e muda de linha.
- **print(..., ..., ...)** → escreve os vários argumentos, separados por espaços, e muda de linha.
- **\n** → muda de linha

```

exemplo.py 005/005
n=int(input("n=?"))
x=float(input("x=?"))
print("n*x")
print(n*x)
print("n*x =", n*x)

```

```

MicroPython v1.9.4
|CASIO COMPUTER CO.,
>>>from exemplo impor
n=?

```

```

>>>from exemplo impor
n=?2
x=?3.5
n*x
7.0
n*x = 7.0
>>>

```

## OPERAÇÕES COM VARIÁVEIS NUMÉRICAS

- **x+y**, **x-y**, **x \* y**, **x/y** → adição (+), subtração (-), multiplicação (X) e divisão (÷)
- **x // y** → quociente da divisão inteira (⌊ ⌋)
- **x % y** → resto da divisão inteira (F4 CHAR)
- **a \*\* n** → potência  $a^n$  (∧)
- **round(x, d)** → valor arredondado de **x** com **d** casas decimais
- **abs()** → valor absoluto
- **min(a, b)**, **max(a, b)** → menor/maior valor entre **a** e **b**

```

Character Select
!"#$%&'()*+,-./0123
456789:;<=>?@ABCDEF
GHIJKLMNOPQRSTUVWXYZ
[\]^_`abcdefgijkl
mnopqrstuvwxyz{|}~

```

```

exemplo.py 007/007
print(7/3)
print(7//3)
print(7%3)
print(round(7/3,2))
print(5**2)
print(abs(-7))
print(min(-3,-7))

```

```

2.3333333333333333
2
1
2.33
25
7
-7

```

Algumas operações e constantes (como  $\pi$ ) são ficam disponíveis importando módulos (*bibliotecas*).

### MÓDULOS MATH E RANDOM

- Importar o módulo math: `from math import*` (catálogo: **SHIFT** **4**).
- `pi` →  $\pi$
- `sqrt()` → raiz quadrada
- `cos()`, `sin()`, `tan()` → funções trigonométricas (em radianos)
- `acos()`, `asin()`, `atan()` → funções trigonométricas inversas ( $\cos^{-1}$ ,  $\sin^{-1}$ ,  $\tan^{-1}$ )
- Importar o módulo random: `from random import*` (catálogo: **SHIFT** **4**).
- `randint(a,b)` → número inteiro aleatório no intervalo  $[a, b]$
- `uniform(a,b)` → número real aleatório no intervalo  $[a, b]$

```
Catalog [F ]
from
from casioplot impo~
from math import *
from random import *
.from_bytes(,)
.fromkeys()
[INPUT] [CAT]
```

```
exemplo.py 007/007
from math import *
print(sqrt(16))
print(cos(pi))
print(acos(-1))
from random import *
print(randint(1,6))
print(uniform(1,2))
[FILE] [RUN] [SYMBOL] [CHAR] [A↔a] [▶]
```

```
>>>from exemplo impor
4.0
-1.0
3.141592653589793
5
1.069638672090014
>>>
[RUN] [A↔a] [CHAR]
```

### CONDIÇÕES

- `x == y` → `x` igual a `y` (condição verdadeira se `x` for igual a `y` e falsa se `x` for diferente de `y`).
- `x != y` → `x` diferente de `y` (**F4** CHAR)
- `x > y` → `x` superior a `y` (**F4** CHAR)
- `x < y` → `x` inferior a `y`
- `x >= y` → `x` superior ou igual a `y`
- `x <= y` → `x` inferior ou igual a `y`

Para criar opções, usam-se estruturas condicionais:

### ESTRUTURAS CONDICIONAIS: **F6** (▶) **F1** (COMMAND) e **F1** (if) ou **F2** (if-else) ou **F3** (if-elif)

- `if` condição 1: → se a condição 1 for verdadeira.  
---| instruções
- `elif` condição 2: → se a condição 1 for falsa e a condição 2 for verdadeira.  
---| instruções
- `else`: → se nenhuma das condições anteriores for verdadeira.  
---| instruções

Numa estrutura `if-elif-else`, é possível colocar tantos `elif` quantos forem necessários.

```
exemplo.py 001/007
if :
elif:
else:
[if ifelseifelif for for-range while]
```

```
exemplo.py 007/008
n=float(input("n=?"))
if n==0:
print("nulo")
elif n>0:
print("positivo")
else:
print("negativo")
[if if-elseif-elif for for-range while]
```

```
n=?3.12
positivo
* SHELL Initialized *
>>>from exemplo impor
n=?-4.5
negativo
>>>
[RUN] [A↔a] [CHAR]
```

PARTE 2 – Explorar programas de Geometria em Python

Exercício 1

1.1 Observe o programa seguinte e, sem o executar, descubra a sua função.

Linguagem natural	Linguagem Python
ler $x$	01 <code>x=int(input("x=? "))</code>
$n \leftarrow$ quociente de $x/360$	02 <code>n=x//360</code>
$\text{alfa} \leftarrow$ resto de $x/360$	03 <code>alfa=x%360</code>
escrever $\text{alfa} + n \times 360$	04 <code>print(alfa, "+", n, "*360")</code>

1.2 Escreva o programa no editor do Menu Python, com o nome **alfa.py** e experimente-o.

Qual é o resultado obtido para  $x = 1000$ ?.....

Exercício 2

2.1 Observe o programa seguinte e, sem o executar, descubra a sua função.

Linguagem natural	Linguagem Python
importar módulo math	01 <code>from math import*</code>
ler $x$	02 <code>x=float(input("x=? "))</code>
$y \leftarrow x \times \pi/180$	03 <code>y=x*pi/180</code>
escrever $y$	04 <code>print(y)</code>

2.2 Escreva o programa no editor do Menu Python, com o nome **conv.py** e experimente-o.

Qual é o resultado obtido para  $x = 180$ ? .....

Exercício 3

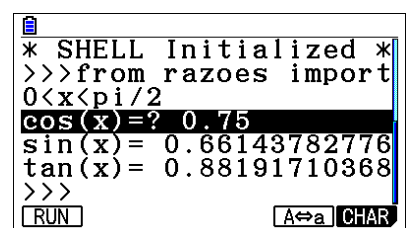
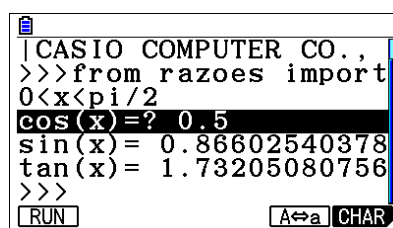
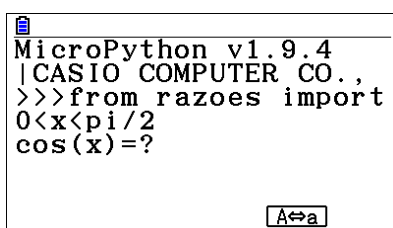
Nota: Em Python, as funções trigonométricas funcionam unicamente em radianos.

3.1 Complete o programa seguinte de modo a apresentar os valores de  $\sin(x)$  e  $\tan(x)$ .

```

from math import*
print("0<x<pi/2")
cosseno = float(input("cos(x)=? "))
seno = ...
print("sin(x)=",...)
tg = ...
print("tan(x)=",...)
    
```

3.2 Escreva o programa no editor do Menu Python, com o nome **razoes.py**, e experimente-o.



#### Exercício 4

4.1 Observe o programa seguinte e, sem o executar, descubra a sua função.

Linguagem natural	Linguagem Python
escrever u(a,b)	01 <code>print("u(a,b)")</code>
ler a	02 <code>a=float(input("a=? "))</code>
ler b	03 <code>b=float(input("b=? "))</code>
escrever v(c,d)	04 <code>print("v(c,d)")</code>
ler c	05 <code>c=float(input("c=? "))</code>
ler d	06 <code>d=float(input("d=? "))</code>
$pe \leftarrow a \times c + b \times d$	07 <code>pe=a*c+b*d</code>
escrever pe	08 <code>print("pe=",pe)</code>

4.2 Escreva o programa no editor do Menu Python, com o nome vetores.py, e experimente-o.

4.3 Modifique o programa de modo a o classificar o ângulo formado pelos vetores  $\vec{u}$  e  $\vec{v}$ .

```
a=? 2
b=? 4
v(c,d)
c=? -1.5
d=? 3
pe= 9.0
Angulo agudo
RUN A⇌a CHAR
```

```
a=? 7
b=? 2
v(c,d)
c=? -2
d=? 7
pe= 0.0
Angulo reto
RUN A⇌a CHAR
```

```
a=? 3
b=? -5
v(c,d)
c=? -2.5
d=? 1
pe= -12.5
Angulo obtuso
RUN A⇌a CHAR
```

### PARTE 3 – Criar novos programas de Geometria em Python

1. Em grupo, criem programas em Python sobre trigonometria, funções trigonométricas ou produto escalar.

2. Apresentem os programas à turma.