

Introdução ao Python

Python para Sala de Aula de Matemática

Luis Alberto D'Afonseca

CEFET-MG

Conteúdo

Apresentação

Introdução a Programação

Introdução ao Python

Gerando Gráficos

Conteúdo

Apresentação

Introdução a Programação

Introdução ao Python

Gerando Gráficos

Algoritmo

- ▶ 2500 BC – Algoritmo da divisão usado pelos Babilônios
- ▶ Século IX – Muhammad ibn Mūsā al-Khwārizmī
- ▶ 1936 – Alan Mathison Turing

Algoritmo

Procedimento para resolver um determinado problema

- ▶ Sequência finita de **passos**
- ▶ Sem ambiguidades
- ▶ Garantia de solução

Quais são os **passos** possíveis?

Programação

Escrever um algoritmo em uma linguagem de programação

- ▶ Os **passos** do algoritmo são **instruções** definidas na linguagem
- ▶ Cada linguagem tem as suas próprias instruções
- ▶ Existem muitas linguagens

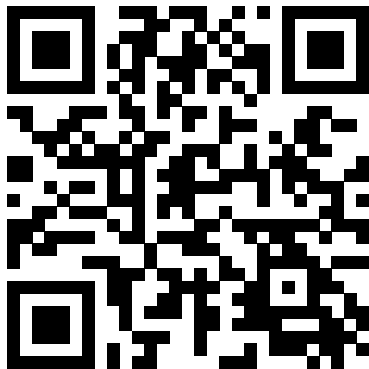
Assembler, Fortran, C, Algol, Java, R, Matlab, Lisp, Python, ...

Python

- ▶ Lançada por Guido van Rossum em 1991
- ▶ Alto nível – Próxima da linguagem humana
- ▶ Uso geral – Não é dedicada a uma única finalidade
- ▶ Interpretada – Não precisa ser compilada
- ▶ Imperativa – Cada comando é uma ordem
- ▶ Tipagem dinâmica – O Python decide o tipo das variáveis

Google Colaboratory – Colab

- ▶ IDE online para Python
- ▶ Diretamente no navegador
- ▶ Precisa de uma conta Google (gmail)
- ▶ Salva os Notebooks no Google Drive



[*https://colab.research.google.com*](https://colab.research.google.com)

Conteúdo

Apresentação

Introdução a Programação

Introdução ao Python

Gerando Gráficos

Operações Numéricas

Principais operações numéricas em Python

Função	Sintaxe
Soma	+
Diferença	-
Produto	*
Quociente	/
Potência	**
Quociente Inteiro	//
Resto da Divisão	%

Python como Calculadora

Principais Relações em Python

Função	Sintaxe
Maior que	>
Maior ou igual a	>=
Menor que	<
Menor ou igual a	<=
Igual	==
Diferente	!=

Variáveis

- ▶ Armazenam valores na memória do computador
- ▶ Alguns tipos de variáveis
 - ▶ Números inteiros `int`
 - ▶ Números reais `float`
 - ▶ Números complexos `complex`
 - ▶ Letras ou texto `string`
 - ▶ Verdadeiro ou falso `boolean`
 - ▶ Matrizes ou vetores `array`

Atribuindo Valores para Uma Variável

- ▶ Usamos o operador `=` para atribuições

```
a = 3
x = 16.89
nome = 'Luis'
```

Python-6-Objetos.lst

- ▶ Não confundir com a comparação `==`

```
4 == 3
4 == 2 + 2
```

- ▶ É perfeitamente válido fazer

```
x = 1
x = x + 1
```

Funções

Função na Programação não é a mesma coisa que na Matemática

- ▶ Implementa um algoritmo
- ▶ Conjunto de instruções
- ▶ Principal forma de organização do código
- ▶ Pode receber parâmetros ou não
- ▶ Pode retornar valores ou não
- ▶ Agrupadas em bibliotecas

Criando Funções

```
def nome_da_funcao( parametro1, parametro2 ):
    comando_1
    comando_2
    comando_3
    comando_4
    return valor_de_retorno
```

Minha Primeira Função

```
def ola_mundo():  
    print( 'Olá Mundo!' )
```


Tomando Decisões

```
if teste:  
    Comandos executados se verdadeiro  
  
else:  
    Comandos executados se falso
```

Tomando Decisões

```
def verificar_se_eh_par( N ):  
  
    if N % 2 == 0:  
        print( N, 'é par')  
  
    else:  
        print( N, 'não é par')
```

Repetições

- ▶ Muitos algoritmos envolvem repetições
- ▶ Calcular o fatorial de um número
- ▶ Desenhar cada lado de um polígono
- ▶ Imprimir cada letra de um texto

Exemplo: Calcular $n!$

Problema: Calcular

$$f = n! = 1 \times 2 \times 3 \times 4 \times \cdots \times n$$

Algoritmo

Dado n

Faça $F = 1$

Para todo k entre 1 e n

Faça $F = F * k$

Repetições no Python

Para todo k entre 1 e n

```
range( 1, n+1 )
```

Começa em 1 e para antes de $n + 1$

Repetições no Python

```
def fatorial( n ):  
    f = 1  
  
    for k in range( 1, n+1 ):  
        f = f * k  
  
    return f
```

Conteúdo

Apresentação

Introdução a Programação

Introdução ao Python

Gerando Gráficos

Bibliotecas

Escolhemos as bibliotecas

```
import numpy as np
```

```
from matplotlib import pyplot as plt
```


Vetores e Matrizes

A biblioteca `numpy` implementa

- ▶ Tipo (objeto) `array` que armazena vetores e matrizes
- ▶ Métodos de Álgebra Linear

Criando um vetor

Listando elemento a elemento

```
x = [ 1, 2, 5, 6, 7 ]
```

Utilizando uma função

```
x = np.linspace( -5, 5, 100 )
```

Desenhar Linha Poligonal

A biblioteca `matplotlib` implementa a função `plot`

- ▶ vetor com coordenadas x
- ▶ vetor com coordenadas y

```
plt.plot( [0 1], [0 1] )
```

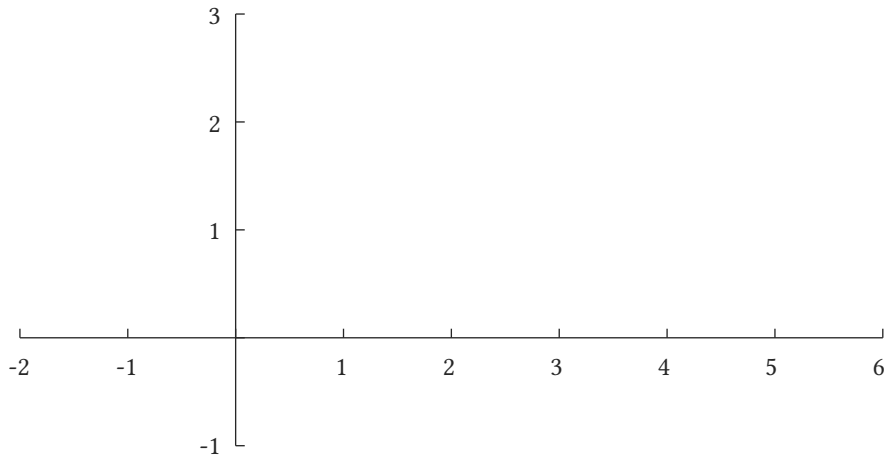
Desenhar o Gráfico de Uma Função

- ▶ Avaliar os valores x desejados e criar o vetor x
- ▶ Avaliar a função nesses pontos e criar o vetor y
- ▶ Desenhar a linha poligonal ligando os pontos

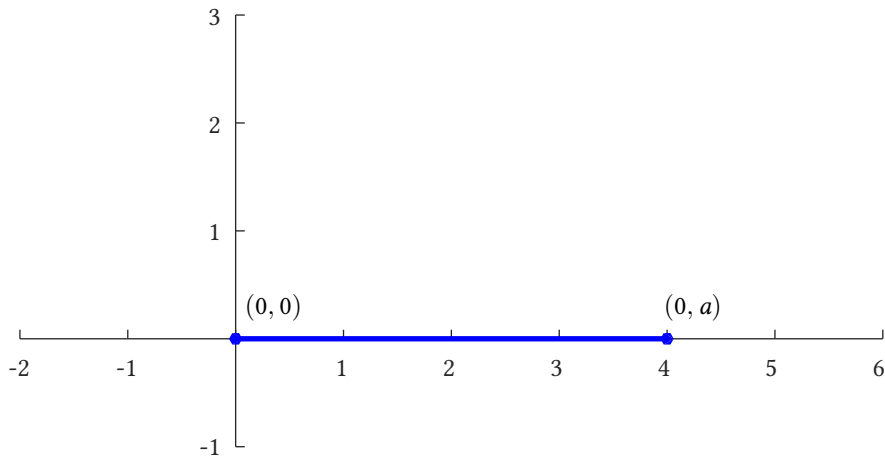
Desenhar Um Triângulo Retângulo

- ▶ Dadas as medidas dos catetos a e b
- ▶ Desenhar um triângulo retângulo no plano cartesiano

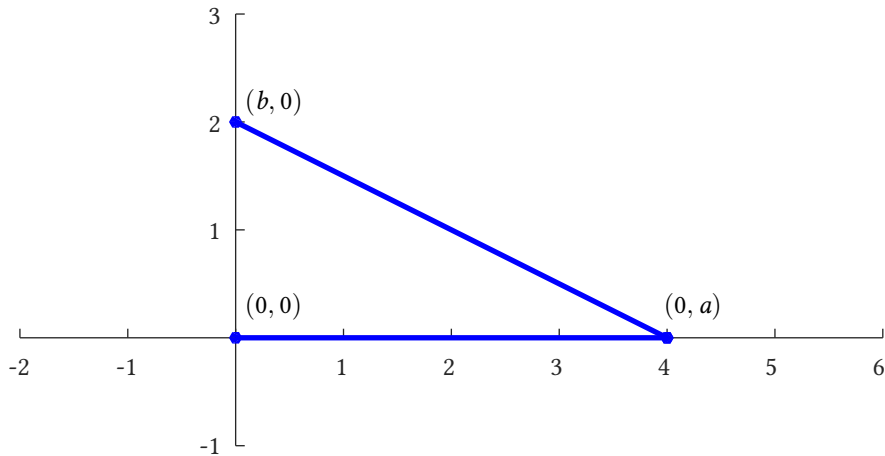
Triangulo 01



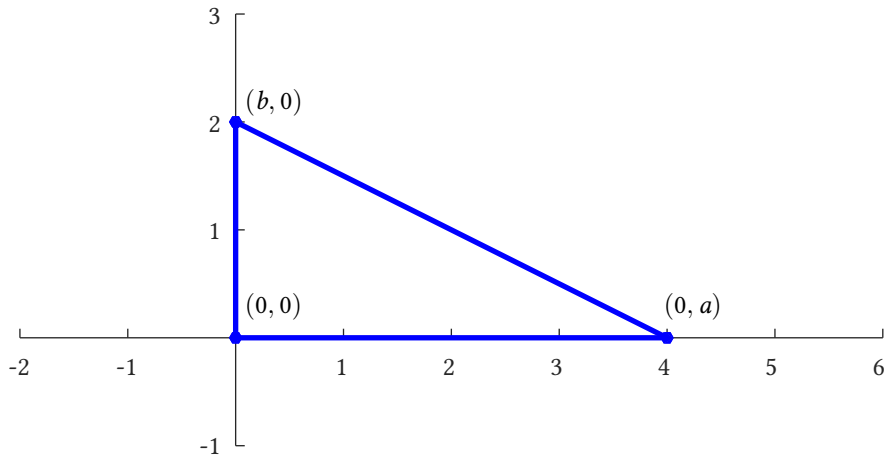
Triangulo 02



Triângulo 03



Triângulo 04



Desenhar Um Triângulo Retângulo

Dados a , b

Desenhar linha poligonal

$(0, 0)$

$(a, 0)$

$(0, b)$

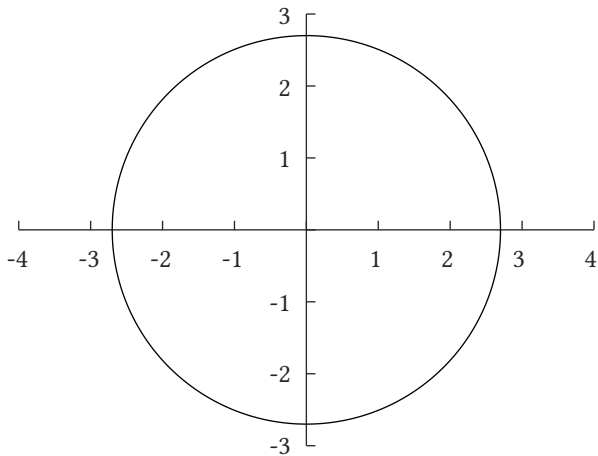
$(0, 0)$

Desenhar Polígono Regular

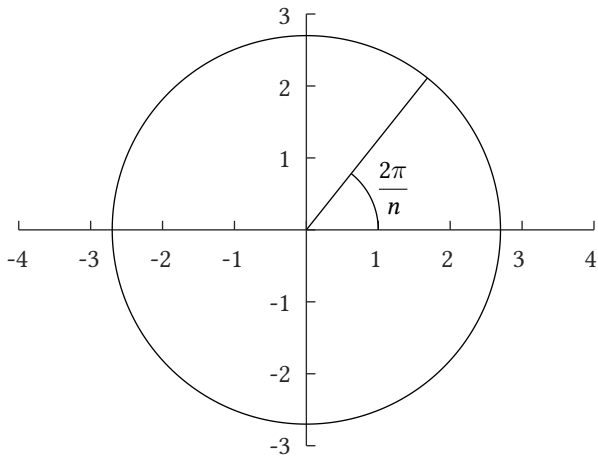
Desenhar o polígono regular de n lados circunscrito na circunferência de raio r

- ▶ Dividir o intervalo $[0, 2\pi]$
- ▶ Calcular as coordenadas (x, y) de cada vértice
- ▶ Desenhar a linha poligonal ligando os vértices

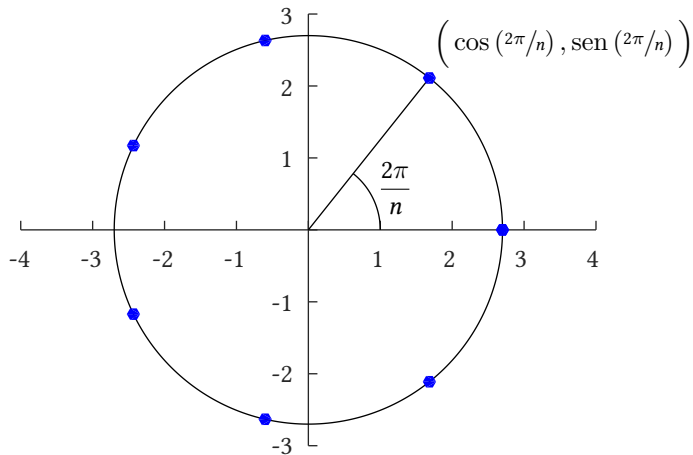
Poligono 01



Poligono 02



Poligono 03



Poligono 04

