Introdução a linguagem python

M.Sc. Luiza P. Stein
lu.p.stein@gmail.com

Quem eu sou?

- Oceanógrafa e Mestre em Ciências [USP]
- Doutorando no laboratório de dinâmica costeira (LDC - USP)
- Uso a programação como ferramenta para solução de problemas e análise de dados em todos os meus projetos desde 2014
- Trabalho com python desde 2017
- Trabalhei como analista de dado freelancer (web scraping)
- Ministrei um minicurso de python para a XVI STO - “Explorando a Interface Terra-Mar”

Mulheres no python

- Pyladies
  http://brasil.pyladies.com/about/
  https://github.com/PyLadiesSP
- Pizza de dados
  https://pizzadedados.com/
- Leticia Portella
  https://leportella.com/pt-br/
- Peixe babel

  https://canalpeixebabel.com.br/

Estrutura do minicurso

- Lógica de programação
- Fundamentos do python
- Principais pacotes para:
  - Análise de séries temporais
  - Visualização de dados
  - Elaboração de mapa

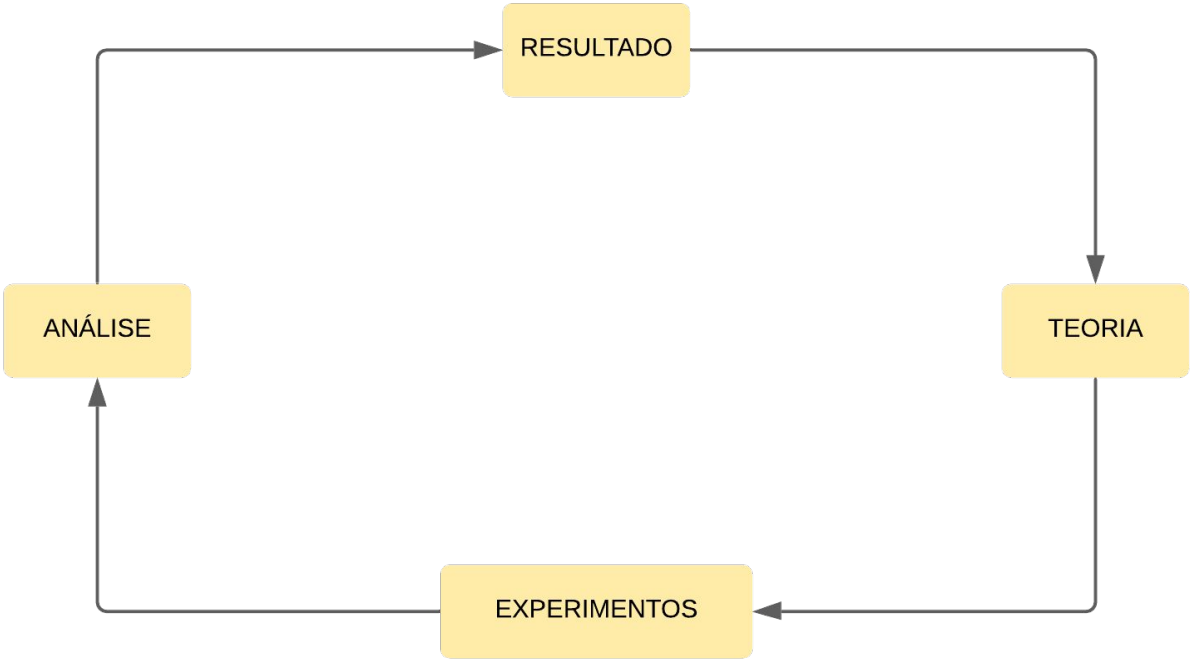
GitHub

- repositório de códigos
- permite fazer controle de versionamento desses códigos
- funciona como portfólio
- tem muito conteúdo/pacotes para resolver diversos problemas:
    - códigos, textos, mas não dados!

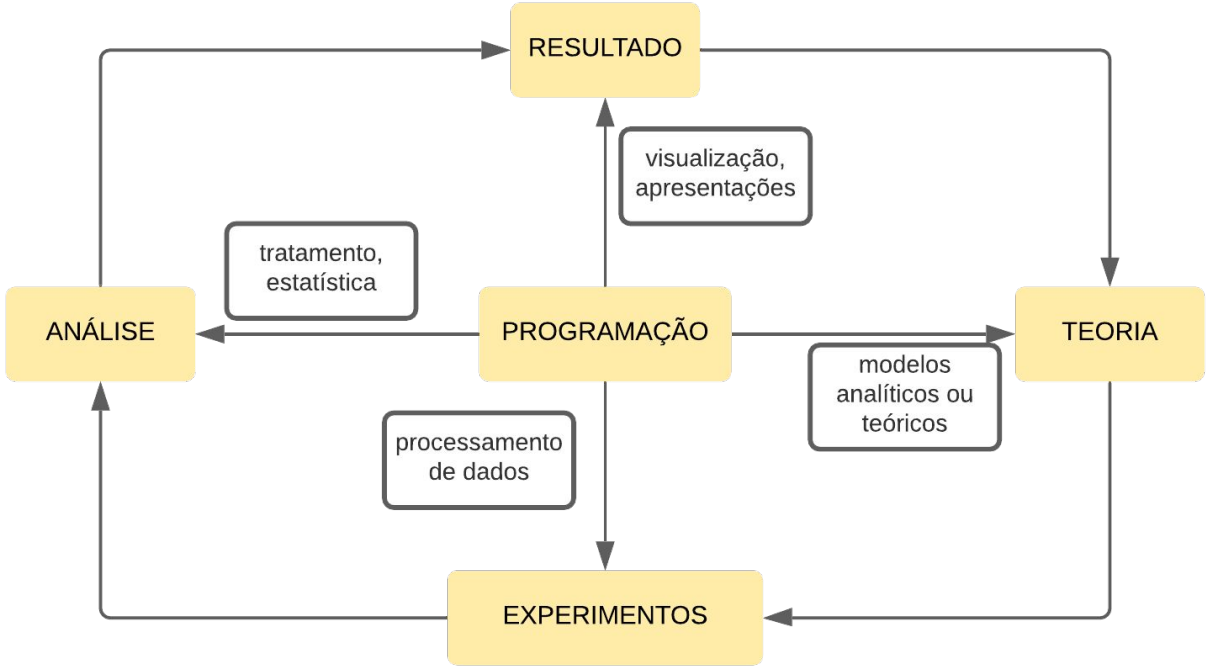
O que é programação?

Uma ferramenta para resolver problemas

Onde se encaixa na ciência e no mercado?



Onde se encaixa na ciência e no mercado?



O que é lógica de programação?

Lógica de programação é a organização coesa de uma sequência de instruções necessárias para que uma função seja executada.

O que é lógica de programação?

Lógica de programação é a organização coesa de uma sequência de instruções necessárias para que uma função seja executada.

A sequência narrativa desses eventos, damos o nome de **algoritmo.**

O que é lógica de programação?

Lógica de programação é a organização coesa de uma sequência de instruções necessárias para que uma função seja executada.

A sequência narrativa desses eventos, damos o nome de **algoritmo.**

- **Ensinar à alguém o caminho para o mercado**
- **Receita de bolo**

O que é lógica de programação

Podemos comparar e exemplificar o algoritmo como uma receita gastronômica. É preciso seguir cada passo para que o resultado final da receita dê certo e ela fique saborosa.

Na programação é a mesma coisa, os algoritmos são escritos seguindo uma lógica, para que o sistema leia e entenda o que deve ser executado.

Por que a lógica de programação é importante?

- É a base para o desenvolvimento de um algoritmo
- Ao aprendermos a pensar logicamente, tendemos a uma maior organização de alguns processos
- Nos ajuda a dividir um problema complexo em pequenas partes para, então, resolvê-las gradualmente a partir de trechos de código
- Ajuda a melhorar a concentração, pois quanto mais claras as ações que precisamos desempenhar para atingir determinados objetivos, podemos ordená-las e executá-las, uma a uma, de maneira mais categórica

O que é lógica de programação?

- Lógica de programação → esboço de uma solução
- Algoritmo → receita/tutorial desta solução

O que é lógica de programação?

Vamos tomar como exemplo o café que tomamos de manhã.

Como vocês fazem para tomar o cafe?

O que é lógica de programação?

1. Vou até a cozinha;
2. Pego o canecão;
3. Coloco água no canecão para fever;
4. Pego a garrafa térmica;
5. Pego o filtro de café;
6. Apoio o filtro de café na garrafa;
7. Pego o pó de café;
8. Coloco o pó de café dentro do filtro;
9. Jogo a água quente no pó de café que está no filtro;
10. Quando o café está pronto, pego a garrafa;
11. Despejo o café pronto dentro de uma caneca;
12. Bebo o café.

Lógica de programação

Exercício:

- escolha um problema que você tenha familiaridade
- esboce um rascunho do que seria a lógica para resolver este problema
- esboce um algoritmo desta lógica

Dicas:

- simples
- qualquer um possa entender

5-10 minutos

Cuidados na elaboração de um algoritmo

- Expressões matemáticas
- Lógica booleana (True/False)

- Existem diversas formas de resolver um problema
- Simplificado
- Dando voltas

- Seja explícito
- Algoritmo bem elaborado
- Código bem escrito.

Cuidados na elaboração de um algoritmo

1. Vou até a cozinha;
2. Pego o canecão;
3. Coloco água no canecão para fever;
4. Pego a garrafa térmica;
5. Pego o filtro de café;
6. Apoio o filtro de café na garrafa;
7. *Coloco o café dentro do filtro;*
8. Jogo a água quente no pó de café que está no filtro;
9. *Despejo o café pronto dentro de uma caneca;*
10. Bebo o café.

Cuidados na elaboração de um algoritmo

1. Vou até a cozinha;
2. Pego o canecão;
3. Coloco água no canecão para fever;
4. Pego a garrafa térmica;
5. Pego o filtro de café;
6. Apoio o filtro de café na garrafa;
7. **Pego o pó de café;**
8. *Coloco o café dentro do filtro;*
9. Jogo a água quente no pó de café que está no filtro;
10. **Quando o café está pronto, pego a garrafa;**
11. **Pego uma caneca;**
12. *Despejo o café pronto dentro de uma caneca;*
13. Bebo o café.

https://www.youtube.com/watch?v=Ct-lOOUqmyY

[intervalo de 5 minutos]

Como melhorar a lógica?

Não tem segredo:

- prática
- paciência
- experiência

Diagrama de blocos pode ajudar!

Diagrama de blocos

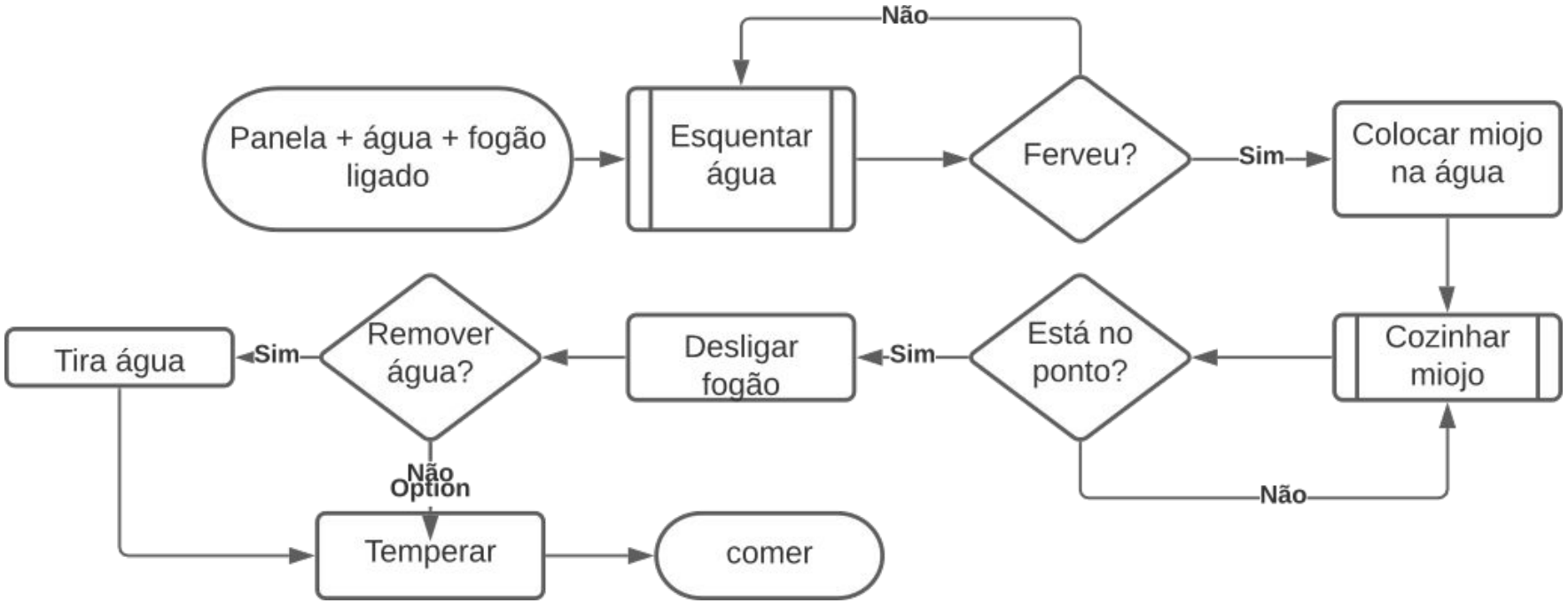
“é uma forma padronizada de representar os diversos fluxos que um algoritmo pode ter através de um conjunto de símbolos com significado específico, podemos sinalizar a intenção do algoritmo em cada etapa, criando uma espécie de mapa”

Exemplos ...

Sequência de instruções

1. Pegar uma panela
2. Colocar água na panela
3. Colocar a panela com água na boca do fogão
4. Ascender o fogo da boca que está a panela
5. Esquentar a água até ferver
6. Colocar o miojo na água
7. Cozinhar o miojo até estar no ponto
8. Desligar o fogão
9. Remover água do miojo se quiser
10. Temperar
11. Comer

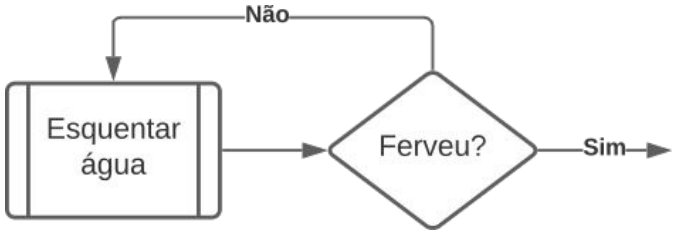
Diagrama de blocos

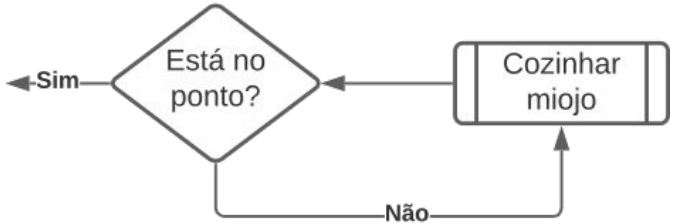


Estruturas de repetição (loops)

Nos permite executar mais de uma vez um mesmo código.

- repetir mesma ação até que algo de diferente aconte

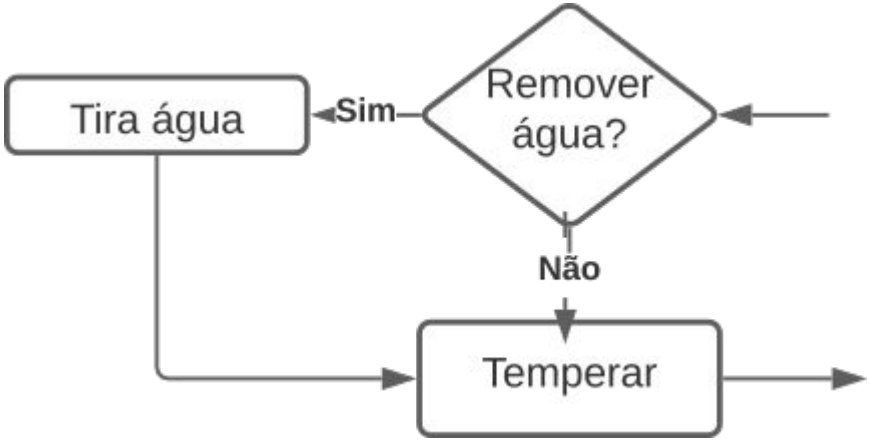




Desvios condicionais

Tomada de decisão no código.

- realizar um código diferente caso uma condição seja satisfeita:



Lógica de programação

Exercício:

Com o algoritmo feito anteriormente, tente adequá-lo a um esquema de diagrama de blocos.

Usem os elementos para um desvio condicional e estrutura de repetição sempre que possível.

5-10 minutos.

Linguagem de programação: o que são?

Para realizar esse processo, é necessário o uso de uma **linguagem de programação**

- é como se fosse uma língua/idioma normal, onde as palavras possuem significados
- ao utilizá-la o computador assimila cada comando e executa cada função do algoritmo, como escrito pelo programador
- traduzidos (compilados) para uma mesma linguagem:
    - binário (é disso que o computador gosta)

Linguagem de programação: o que são?

Podemos entender que essa linguagem é a maneira de se escrever o algoritmo e compreender a lógica de programação é saber o que dizer para o computador executar o desejado.

Linguagem de programação: o que são?

Exemplos:

- C#, php: desenvolvimento web
- Java: mainframe de bancos
- Fortran, C: modelos numéricos
- R, Julia: análise e visualização de dados

Cada linguagem tem suas próprias particularidades, como sua sintaxe, seus tipos de dados e sua orientação, mas a lógica por trás de todas é a mesma.

Linguagem de programação: o que são?

O python é uma dessas linguagens de programação!

E se encaixa em (quase) todas essas aplicações!

Python!

Python é uma linguagem de alto nível!

Ou seja, muito próximo à linguagem humana!

Python!

A linguagem Python é tipicamente usada para aplicações web ou linguagem de scripts para administração de sistemas.

Python!

Python é uma linguagem de alto nível:

- open source (não precisa pagar)
- relativamente mais fácil de aprender
- flexibilidade de aplicações
- comunidade científica ativa
- manutenção constante
- revisão de pacotes científicos

Python!

Além disso:

- demanda de empresas
- serviços de download de conjuntos globais (mercator, ecmwf, ncep/ncar)
- NOAA migrando do NCL para python
- modelos numéricos possuem pacotes específicos em python

Python!

Java

```java
public class Hello
{
    public static void main(String args[]) {
        java.util.Scanner s = new java.util.Scanner(System.in);
        System.out.print("Digite seu nome:");
        String nome = s.nextLine();
        System.out.println("Olá, " + nome);
    }
}
```

Pascal

```pascal
program HelloWorld(output);
var
    nome: string;
begin
    writeln('Digite seu nome:');
    read(nome);
    writeln('Olá, ', nome);
end.
```

C

```c
#include &lt;stdio.h&gt;
int main()
{
    char nome[200];
    printf("Digite seu nome:");
    scanf("%s", nome);
    printf("Olá, %s\n", nome);
    return 0;
}
```

Python

```python
nome = input('Digite seu nome:')
print ('Olá,', nome)
```

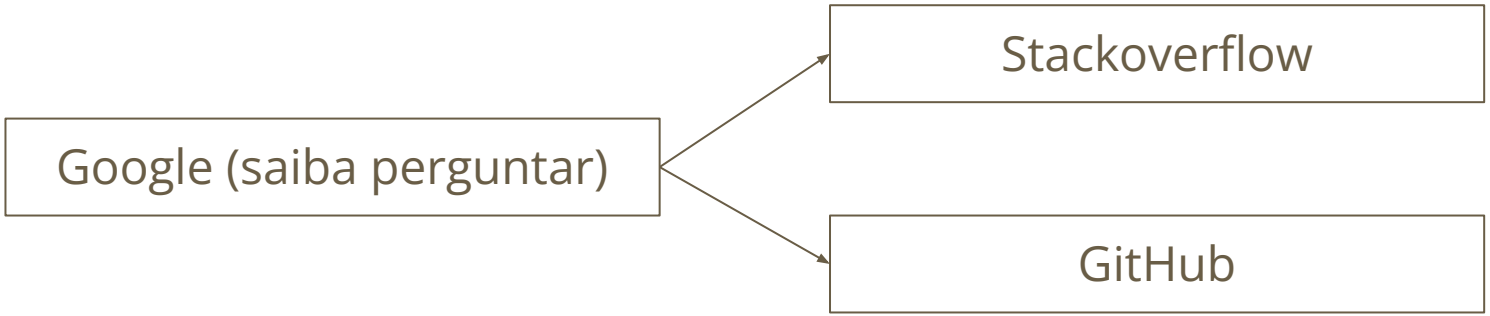
Fonte: https://pythonhelp.wordpress.com/por-que-python/

Existe uma linguagem melhor?

A que você se adequar melhor:

- souber usar
- resolver seus problemas

Sites para dúvidas no python



Com certeza alguém já teve o seu problema antes!

Ferramentas para programar em python

Integrated Development Environment (IDEs)

- Pycharm
- Spyder
- VS Code
- Jupyter (roda no navegador)

C:/Users/TestUser/Documents/Spyder - Spyder (Python 3.6)

File Edit Search Source Run Debug Consoles Projects Tools View Help

C:\Users\TestUser\Downloads\16740156420fb678bb5ba67c3ee3aae4-551407feca17b0f67bb9f85687f4db8d1b953678\16740156420fb678bb5ba67c3ee3aae4-551407feca17b0f67bb9f85687f4db8d1b953678

Project explorer

Spyder
Data
spyder
.github
conda.recipe
continuous_integration
doc
img_src
requirements
rope_profiling
scripts
spyder
app
tests
__init__.py
cli_options.py
mac_stylesheet.qss
mainwindow.py
restart.py
start.py
tour.py
config
defaults
fonts
images
locale
plugins
tests
utils
widgets
windows
workers
__init__.py
dependencies.py
interpreter.py
otherplugins.py
pil_patch.py
py3compat.py
pyplot.py
requirements.py
spyder_breakpoints
spyder_io_dcm
spyder_io_hdf5
spyder_profiler
spyder_pylint
.checkignore
.ciocheck
.ciocopyright
.codecov.yml
.coveragerc
.gitignore
.pep8speaks.yml
.project
.travis.yml
Announcements.md
appveyor.yml

Editor - C:\Users\TestUser\Downloads\16740156420fb678bb5ba67c3ee3aae4-551407feca17b0f67bb9f85687f4db8d1b953678\16740156420fb6...

temp.py | interpolation.py | __init__.py | umd_helper.py | umd_main.py | README.md

```
6
7  import pylab
8  from numpy import cos, linspace, pi, sin, random
9  from scipy.interpolate import splprep, splev
10
11 # %% Generate data for analysis
12
13 # Make ascending spiral in 3-space
14 t = linspace(0, 1.75 * 2 * pi, 100)
15
16 x = sin(t)
17 y = cos(t)
18 z = t
19
20 # Add noise
21 x += random.normal(scale=0.1, size=x.shape)
22 y += random.normal(scale=0.1, size=y.shape)
23 z += random.normal(scale=0.1, size=z.shape)
24
25
26 # %% Perform calculations
27
28 # Spline parameters
29 smoothness = 3.0  # Smoothness parameter
30 k_param = 2  # Spline order
31 nests = -1  # Estimate of number of knots needed (-1 = maximal)
32
33 # Find the knot points
34 knot_points, u = splprep([x, y, z], s=smoothness, k=k_param, nests=-1)
35
36 # Evaluate spline, including interpolated points
37 xnew, ynew, znew = splev(linspace(0, 1, 400), knot_points)
38
39
40 # %% Plot results
41
42 # TODO: Rewrite to avoid code smell
43 pylab.subplot(2, 2, 1)
44 data, = pylab.plot(x, y, 'bo-', label='Data with X-Y Cross Section')
45 fit, = pylab.plot(xnew, ynew, 'r-', label='Fit with X-Y Cross Section')
46 pylab.legend()
47 pylab.xlabel('x')
48 pylab.ylabel('y')
49
50 pylab.subplot(2, 2, 2)
51 data, = pylab.plot(x, z, 'bo-', label='Data with X-Z Cross Section')
52 fit, = pylab.plot(xnew, znew, 'r-', label='Fit with X-Z Cross Section')
53 pylab.legend()
54 pylab.xlabel('x')
```

Outline

interpolation.py
Generate data for analysis
Perform calculations
Plot results
imputerNan
Queue
__init__
appendleft
pop
with open(data_path + output_file_n...
with open(data_path + output_file_n...
with open(data_path + output_file_n...
print_file
Example Bitzer class
DataSet
__init__
prepare_dataset
Serie
something
Dataframe
something
foo
__init__
spam
with open(file) as f:
Base
Derived
for r, bar in zip(radii, bars):
with np.load(filename) as dem:

Variable explorer

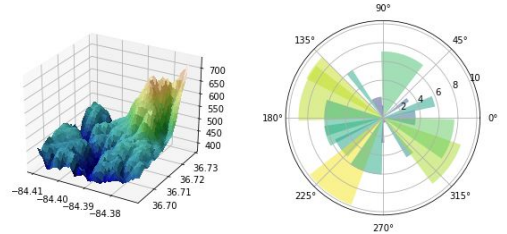
Name	Type	Size	Value
array_int8	int8	(2, 3)	Min: -7 Max: 6
array_uint32	uint32	(2, 2, 3)	Min: 1 Max: 7
bars	container.BarContainer	20	BarContainer object of matplotlib.conta…
df	DataFrame	(3, 2)	Column names: bools, ints
filename	str	1	C:\ProgramData\Anaconda3\lib\site-packa…
list_test	list	2	[Dataframe, Numpy array]
nrows	int	1	344
r	float64	1	7.611082589334796
radii	float64	(20,)	Min: 0.4983036638535687 Max: 9.856848974942551
region	tuple	2	(slice, slice)
rgb	float64	(45, 45, 4)	Min: 0.0 Max: 1.0
series	Series	(1,)	Series object of pandas.core.series mod…
test_none	NoneType	1	NoneType object of builtins module

Variable explorer | Help | File explorer | Find in files | Breakpoints | Static code analysis | Profiler | Online help

IPython console

Console 1/A | Console 2/A | Custom Name/A | 00:34:13

```
...:
...: ls = LightSource(270, 45)
...: # To use a custom hillshading mode, override the built-in shading
...: # in the rgb colors of the shaded surface calculated from "shade".
...: rgb = ls.shade(z, cmap=cm.gist_earth, vert_exag=0.1, blend_mode='soft')
...: surf = ax.plot_surface(x, y, z, rstride=1, cstride=1, facecolors=rgb,
...:                        linewidth=0, antialiased=False, shade=False)
...:
...: plt.show()
```



In [12]:

IPython console | History log | Internal console

Permissions: RW | End-of-lines: LF | Encoding: UTF-8 | Line: 26 | Column: 4 | Memory: 49 % | CPU: 15 %

Ferramentas para programar em python

Integrated Development Environment (IDEs)

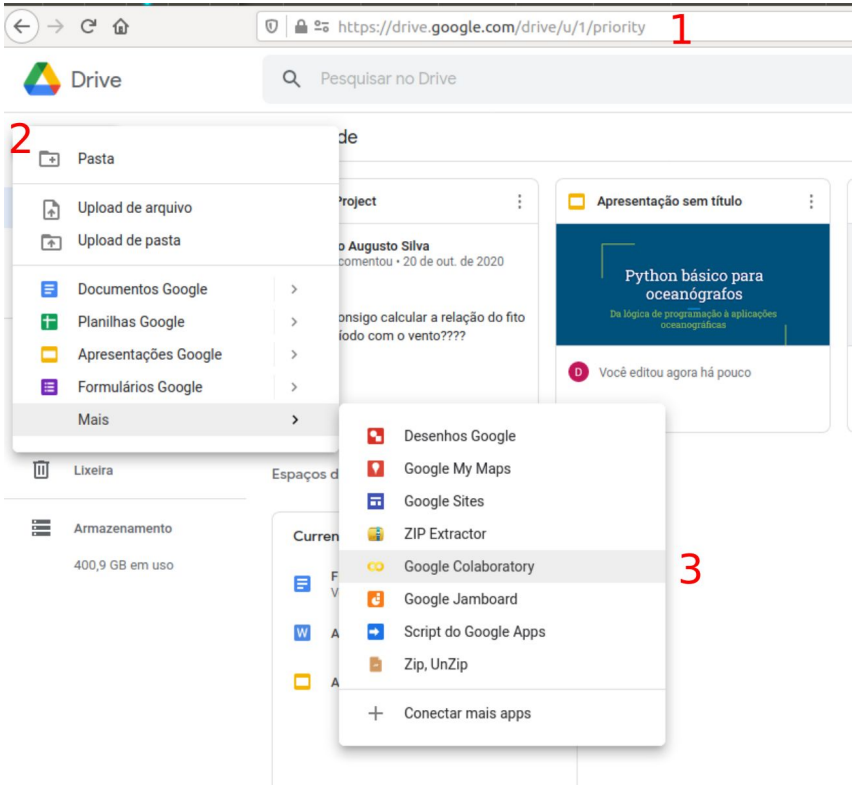
- pycharm
- spyder
- VS Code
- Jupyter (roda no navegador)

Computação em nuvem (cloud computing)

- Google Colab (sugestão de uso para trabalhar de fato)

Google colab

Acessar: drive.google.com



Ferramentas para programar em python

Integrated Development Environment (IDEs)

- pycharm
- spyder
- VS Code
- Jupyter (roda no navegador)

Computação em nuvem (cloud computing)

- Google Colab (sugestão de uso para trabalhar de fato)

Editores de texto + terminal/console

- notepad++ (windows)
- gedit
- atom
- sublime

Ferramentas para programar em python

Vamos olhar um editor de texto para ver como é...

Ambientes virtuais

- O que são?
  - Empacotamento de bibliotecas e softwares isolados do seu sistema
- Por que usá-los?
  - Proteção do seu sistema
  - Portabilidade
- Cada projeto = 1 ambiente [!!!]
  - Preservar as versões de cada biblioteca para reproduzir
  - Organização

Ambientes virtuais: alguns nomes importantes

CONDA

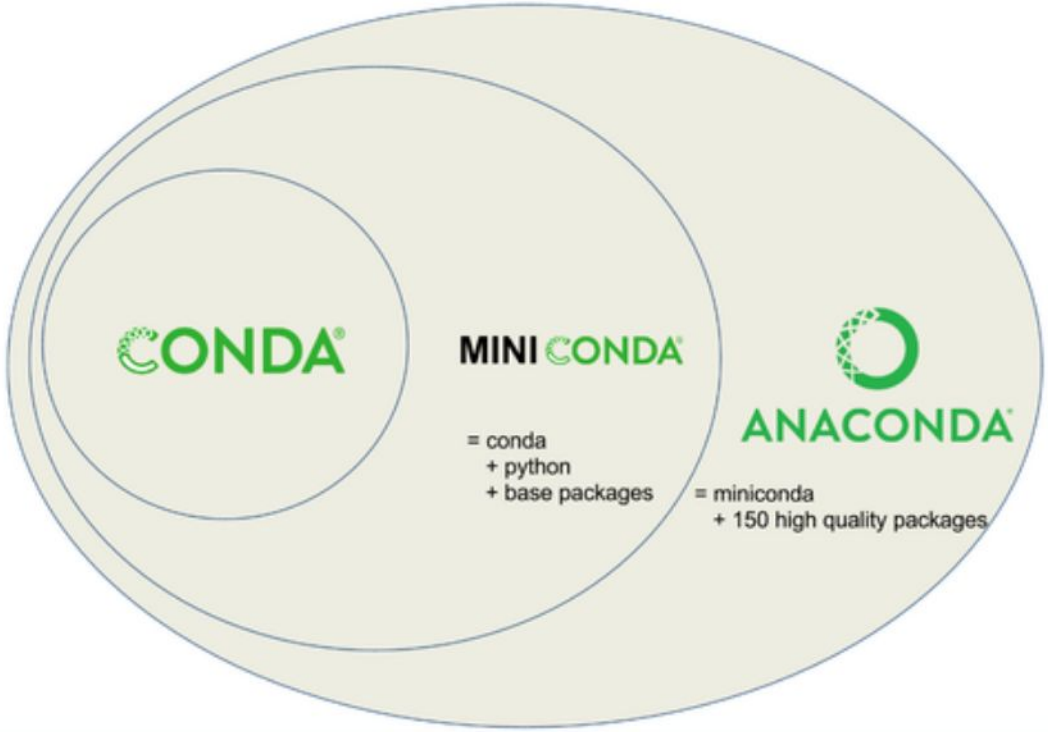
- gerenciador de ambientes virtuais

MINICONDA

- instalação limpa do conda com alguns pacotes úteis

ANACONDA

- instalação completa do conda com muitos pacotes



Ambientes virtuais: alguns nomes importantes

CONDA

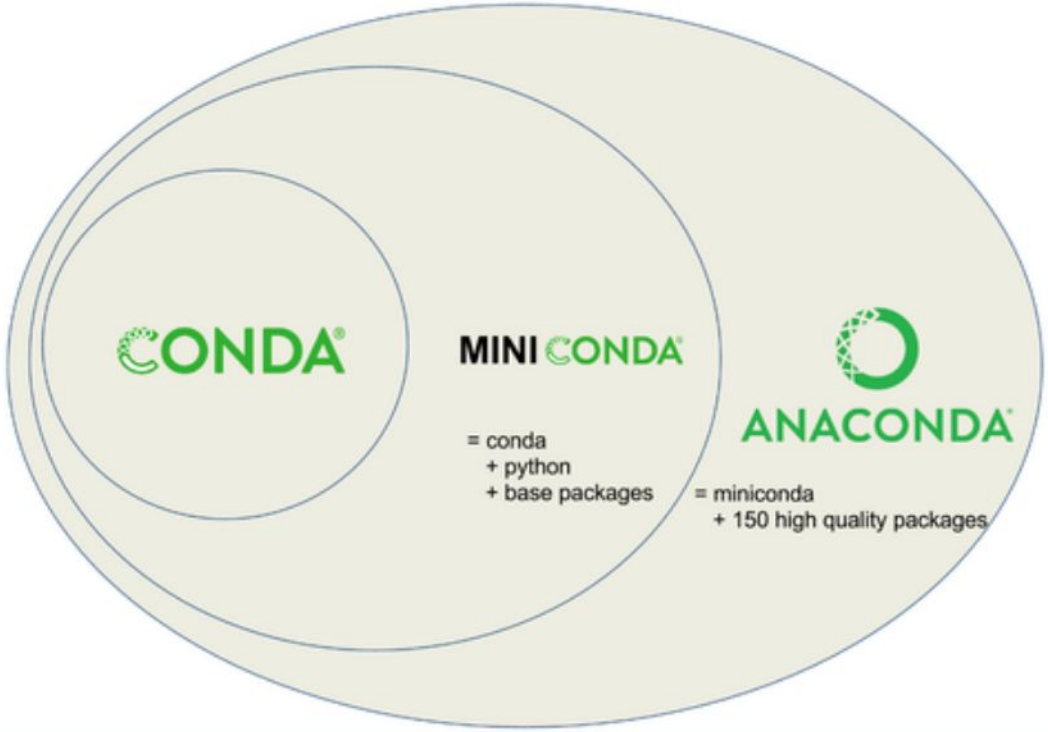
- gerenciador de ambientes virtuais

MINICONDA

- instalação limpa do conda com alguns pacotes úteis

ANACONDA

- instalação completa do conda com muitos pacotes



Ambientes virtuais

Vejamos um pouco de terminal ...

Dúvidas/Problemas na instalação do python

Obrigada!

Lembrem-se:

Qualquer dúvida, exercitem a pesquisa nos sites sugeridos e perguntem nos canais de comunicação (meu e-mail ou tragam na próxima aula).

Até amanhã!
Boa noite a todas!