

GGO00124 - INTRODUÇÃO À PROGRAMAÇÃO PARA GEOCIENTISTAS - A1

Flora Solon
florasolon@id.uff.br



Objetivos da Disciplina

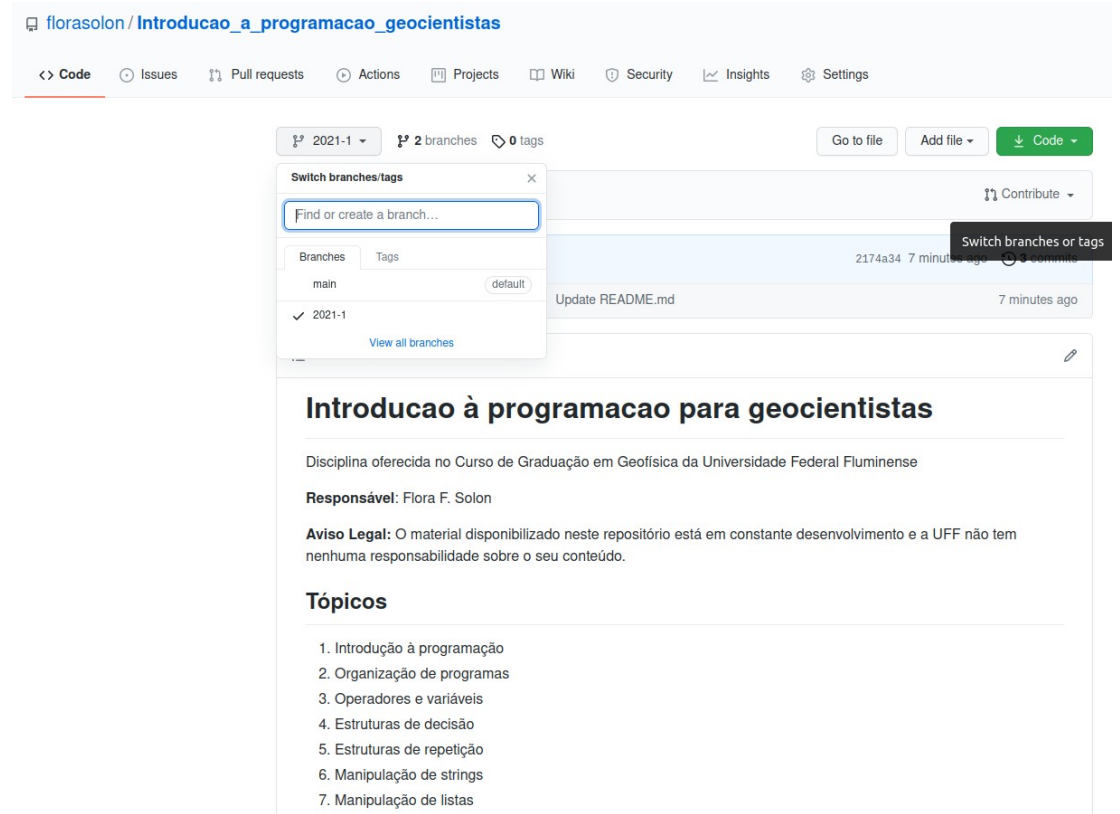
A disciplina vai apresentar os conceitos básicos de programação de computadores. O aluno aprenderá a desenvolver programas utilizando técnicas básicas de programação e o conceito de tipos de dados aplicados principalmente às geociências.

- Solucionar de problemas com foco em geociencias
- Desenvolver pensamento lógico e computacional
- Escrever e ler na linguagem do computador
- Objetivo secundário - Programar em Python

Repositório

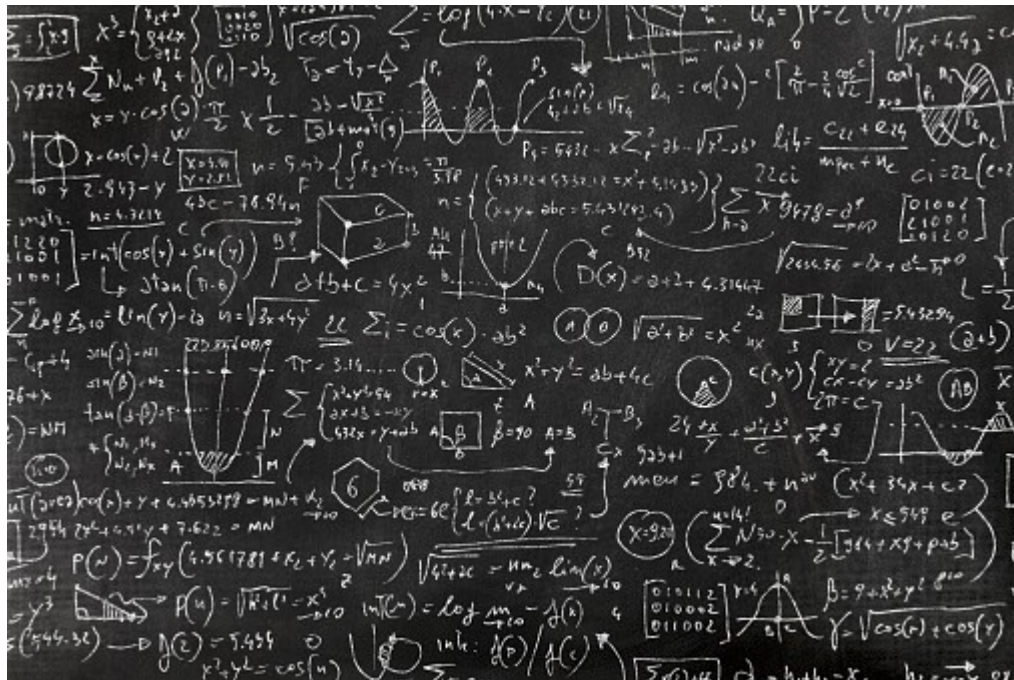
- Github

https://github.com/florasolon/Introducao_a_programacao_geocientistas/tree/2021-1



Por que aprender a programar?

- Resolver sistemas complexos de equações que não necessariamente podem ser resolvidos por softwares padrões



Por que aprender a programar?

- Resolver sistemas complexos de equações que não necessariamente podem ser resolvidos por softwares padrões
- Implementar seus próprios códigos.

Disciplina: Introdução a programação para geocientistas

Aula 1 - Oi mundo

```
In [3]: print('Oi mundo pandêmico')
```

Oi mundo pandêmico

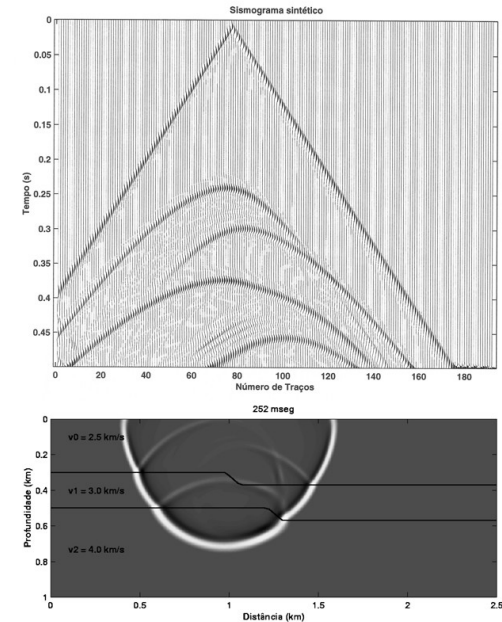
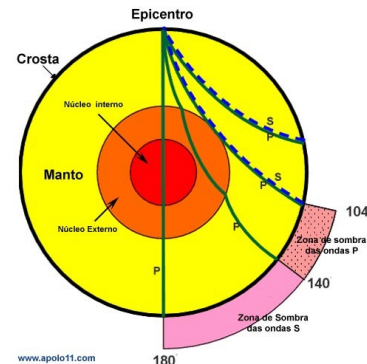
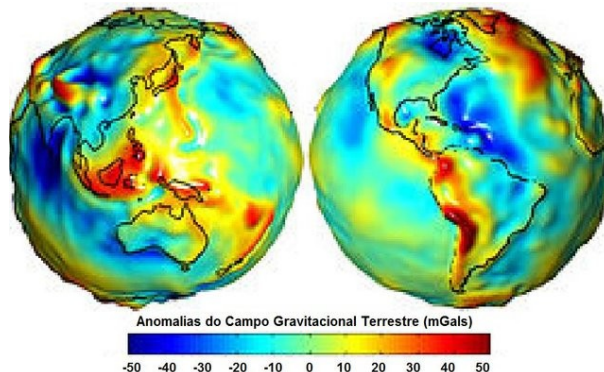
```
In [ ]: |
```

Por que aprender a programar?

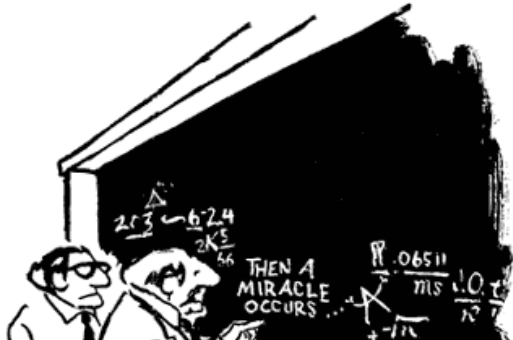
- Resolver sistemas complexos de equações que não necessariamente podem ser resolvidos por softwares padrões
- Implementar seus próprios códigos.
- Fazer simulações. Gerar modelos para explicar algum fenômeno

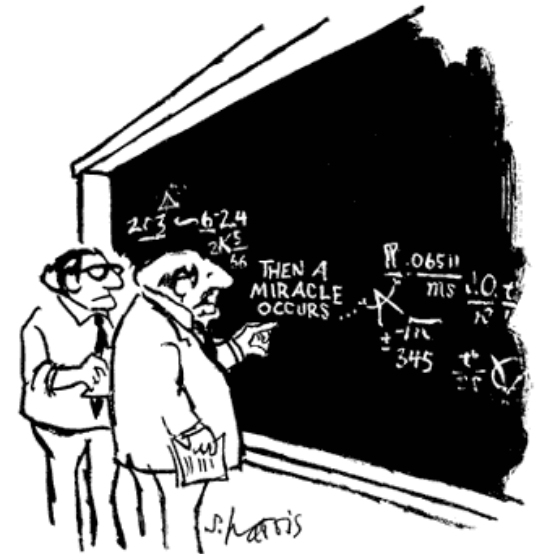
Por que aprender a programar?

- Resolver sistemas complexos de equações que não necessariamente podem ser resolvidos por softwares padrões
- Implementar seus próprios códigos.
- Fazer simulações. Gerar modelos para explicar algum fenômeno
 - Propagação de uma onda sísmica – conhecer a estrutura interna da Terra
 - Simular o campo gravitacional de um corpo
 - Simular o campo magnético de um corpo



Por que aprender a programar?

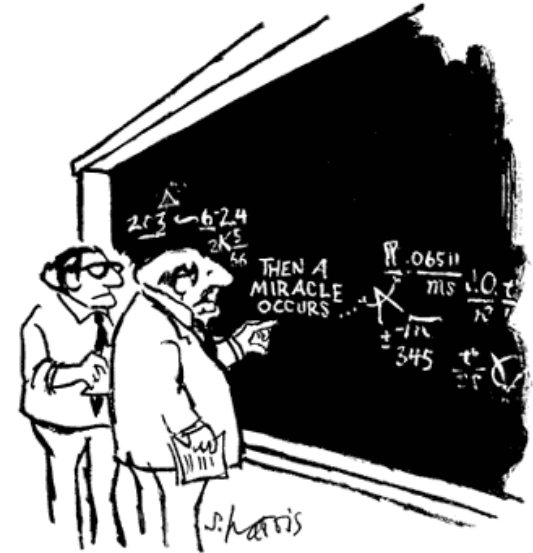
- Resolver sistemas complexos de equações que não necessariamente podem ser resolvidos por softwares padrões
 - Implementar seus próprios códigos.
 - Fazer simulações. Gerar modelos para explicar algum fenômeno
 - Propagação de uma onda sísmica – conhecer a estrutura interna da Terra
 - Simular o campo gravitacional de um corpo
 - Simular o campo magnético de um corpo
 - Propor uma hipótese e testá-la.
- 
- A cartoon illustration of two men in suits standing in front of a chalkboard. The chalkboard is covered with various mathematical formulas and symbols, including
- Δ
- ,
- 203
- ,
- 6.24
- ,
- $2K^{\frac{5}{66}}$
- ,
- $\frac{R \cdot 0.6511}{ms}$
- ,
- $\frac{1.0}{K}$
- , and
- $\frac{1}{\pi}$
- . The text "THEN A MIRACLE OCCURS" is written on the board. The man on the left is pointing at the board, and the man on the right is looking at it with a concerned expression.



"I think you should be more explicit here in step two."

Por que aprender a programar?

- Resolver sistemas complexos de equações que não necessariamente podem ser resolvidos por softwares padrões
- Implementar seus próprios códigos.
- Fazer simulações. Gerar modelos para explicar algum fenômeno
 - Propagação de uma onda sísmica – conhecer a estrutura interna da Terra
 - Simular o campo gravitacional de um corpo
 - Simular o campo magnético de um corpo
- Propor uma hipótese e testá-la.
- Quando os sistemas podem ser “modelados matematicamente”, são criados programas que fazem a simulação do sistema para checagem de uma hipótese.



"I think you should be more explicit here in step two."

Linguagem de Programação

- Python

Linguagem de Programação

- Python

Diferentemente de C ou Fortran, a linguagem Python é interpretada. Isso significa que o código não precisa ser previamente compilado e os comandos são executados imediatamente. De acordo com a *Software Carpentry*, quando estamos programando, o tempo total necessário para obtermos a solução desejada é determinado por duas coisas: o tempo gasto por você para desenvolver o código e o tempo gasto pelo computador para rodar o código. Estes fatores devem ser levados em consideração no momento da escolha de uma linguagem de programação. Para fins acadêmicos de pesquisa e ensino, a linguagem Python oferece algumas vantagens, dentre as quais eu destaco o fato de ser gratuita e distribuída livremente na internet, relativamente fácil de aprender e extremamente bem documentada.

Linguagem de Programação

- Python

Diferentemente de C ou Fortran, a linguagem Python é interpretada. Isso significa que o código não precisa ser previamente compilado e os comandos são executados imediatamente. De acordo com a *Software Carpentry*, quando estamos programando, o tempo total necessário para obtermos a solução desejada é determinado por duas coisas: o tempo gasto por você para desenvolver o código e o tempo gasto pelo computador para rodar o código. Estes fatores devem ser levados em consideração no momento da escolha de uma linguagem de programação. Para fins acadêmicos de pesquisa e ensino, a linguagem Python oferece algumas vantagens, dentre as quais eu destaco o fato de ser gratuita e distribuída livremente na internet, relativamente fácil de aprender e extremamente bem documentada.

Neste curso, usaremos a distribuição **Python Anaconda 3.x** e os códigos serão feitos com o **Jupyter Notebook**.

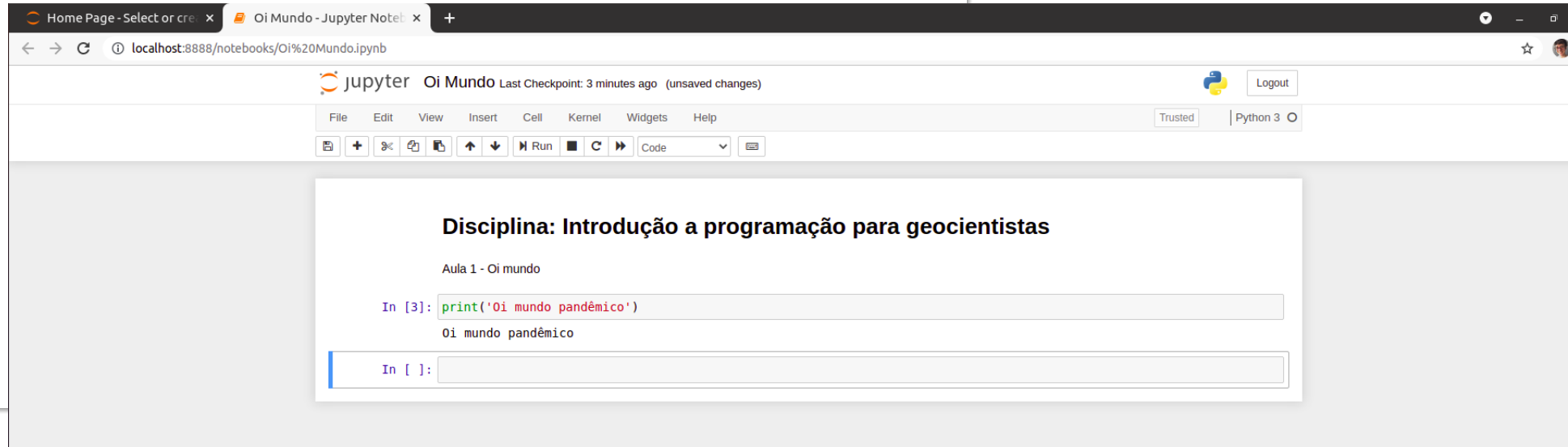
Linguagem de Programação

- Jupyter Notebook

O **Jupyter Notebook** é um arquivo com extensão **.ipynb** e permite combinar código, texto, equações feitas em LaTeX, figuras e animações. Além disso, é gratuito e extremamente bem documentado. Esta poderosa ferramenta computacional possibilita reunir (quase) todas as etapas envolvidas no desenvolvimento de um código com fins acadêmicos, desde a leitura e processamento dos dados até a visualização dos resultados.

Linguagem de Programação

- Jupyter Notebook



Instalação

Para instalar o Python Anaconda, sugiro acessar diretamente o site do Python Anaconda.

Para checar se a instalação deu certo, abra uma janela do prompt de comando, caso você esteja no Windows, ou um terminal, se estiver no Linux. Em seguida, Digite o comando: **conda list**. Este comando mostrará uma lista de coisas que foram instaladas pelo Anaconda. Deve aparecer algo do tipo:

```
florislei@florislei: ~  
(base) florislei@florislei:~$ conda list  
# packages in environment at /home/florislei/anaconda3:  
#  
# Name                                Version                                Build      Channel  
_ipyw_jlab_nb_ext_conf                0.1.0                                py38_0  
_libgcc_mutex                         0.1                                  main  
alabaster                             0.7.12                              py_0  
anaconda                              2020.07                             py38_0  
anaconda-client                       1.7.2                              py38_0  
anaconda-navigator                   1.9.12                              py38_0  
anaconda-project                     0.8.4                               py_0  
argh                                  0.26.2                              py38_0  
asn1crypto                           1.3.0                               py38_0  
astroid                              2.4.2                               py38_0  
astropy                              4.0.1.post1                         py38h7b6447c_1  
atomicwrites                         1.4.0                               py_0  
attrs                                19.3.0                              py_0  
autopep8                             1.5.3                               py_0  
babel                                2.8.0                               py_0  
backcall                             0.2.0                               py_0  
backports                             1.0                                 py_2  
backports.functools_lru_cache         1.6.1                               py_0  
backports.shutil_get_terminal_size    1.0.0                              py38_2  
backports.tempfile                   1.0                                 py_1  
backports.weakref                    1.0.post1                           py_1  
basemap                              1.2.0                              py38h856778e_4  
beautifulsoup4                       4.9.1                              py38_0  
bitarray                             1.4.0                              py38h7b6447c_0  
bkcharts                             0.2                                 py38_0  
blas                                 1.0                                 mkl  
bleach                               3.1.5                              py_0  
blosc                                1.19.0                             hd408876_0  
bokeh                                 2.1.1                              py38_0  
boto                                  2.49.0                             py38_0  
bottleneck                           1.3.2                              py38heb32a55_1  
brotlipy                             0.7.0                              py38h7b6447c_1000  
bzip2                                1.0.8                              h7b6447c_0  
ca-certificates                      2020.6.24                           0  
cairo                                 1.14.12                             h8948797_3  
cartopy                              0.19.0.post1                         pypi_0    pypi
```

Instalação

- **Passo a Passo**

<http://swcarpentry.github.io/python-novice-gapminder/setup.html>

- Download Windows

<https://www.anaconda.com/products/individual#download-section>

- Download Linux

<https://www.anaconda.com/products/individual#download-section>

Aulas

- Terças e quintas das 11 às 13h
- Síncronas e assíncronas
 - Comunicação pelo Google Classroom
 - Aulas no Google Meet ou Zoom (mandarei o link pelo classroom)
- Assíncronas:
 - Conteúdo novo – video e slides e/ou Jupyter notebook exemplo
- Síncronas:
 - Dúvidas das aulas assíncronas
 - Conteúdo novo
 - Também serão gravadas

Avaliação

- Avaliação continuada com exercícios práticos a cada semana (aplicados um mês após o início do semestre letivo)
 - Comunicação pelo Google Classroom
 - Exercícios pelo run.codes
 - Programa avaliado por execução
- Nota final
 - Média aritmética simples das notas dos exercícios
 - Aprovado - nota final ≥ 6
 - Reprovado - nota final < 6

Ementa

- Introdução à programação
- Organização de programas
- Operadores e variáveis
- Estruturas de decisão
- Estruturas de repetição
- Manipulação de strings
- Manipulação de listas
- Vetores
- Matrizes
- Funções
- Manipulação de arquivos
- Manipulação e análise de dados
- Programação defensiva

Referências

- Eric Freeman, “Use a Cabeça! Aprenda a Programar”, Editora Alta Books, 2019
- Cormen, T. H. Leiserson, C. E; Rivest, R. – Algoritmos: Teoria e Prática, Editora Campus, 2002.
- Forbellone, A.L.V.; Eberspacher, H.F. “lógica de programação”. Makron Books, 2000.
- Furlan, M., Gomes, M., Soares, M., Concilio, R., 2005, Algoritmos e Lógica de Programação, Editora Thomson.
- William H. Press, Saul A. Teukolsky, William T. Vetterling, Numerical Recipes: The Art of Scientific Computing. Cambridge University Press; 3ª edição
- Tutorial Python

<https://docs.python.org/pt-br/3/tutorial/index.html>