Processamento da Informação

Apresentação

Prof. Jesús P. Mena-Chalco
CMCC/UFABC

Q1/2020

Apresentação

- **Professor:**
  Jesús P. Mena-Chalco (CMCC)
  jesus.mena@ufabc.edu.br

- **Formação:**
  - Engenheiro da Computação.
  - Mestre e Doutor em Ciência da Computação.
    Instituto de Matemática e Estatística da USP.

- Sala 517-A, torre 2, 5º Andar.

- **Áreas de pesquisa:**
  - Pattern recognition
  - Graph mining
  - Scientometrics/Bibliometrics

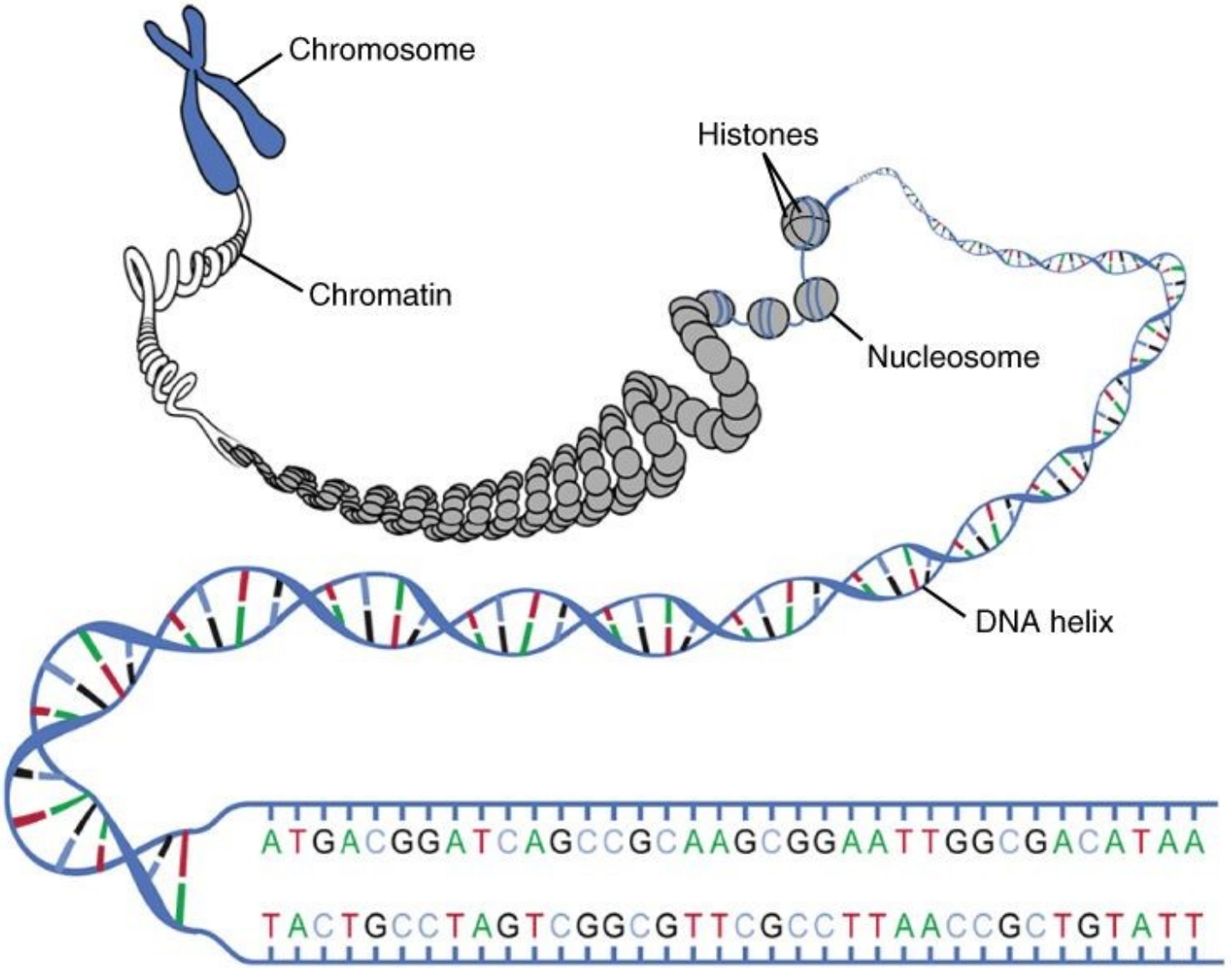
leonardodicaprio

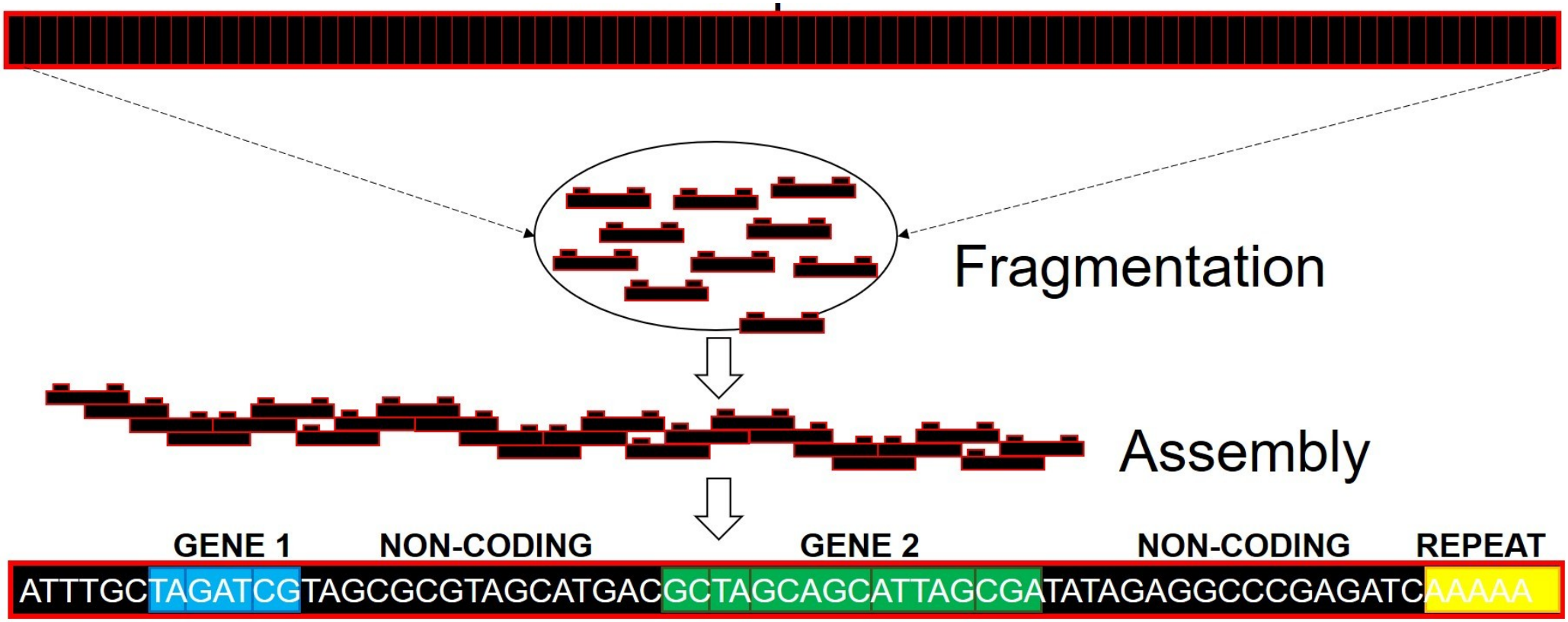
344.639 curtidas

leonardodicaprio #Regram #RG @leonardodicapriofdn: The Rondonia region of #Brazil originally had over 200,000 km2 of rainforest but has become one of the most deforested places in the Amazon. Side by side images shows C. 2006 to 2018. #10YearChallenge

Como determinar de forma objetiva a proporção de desmatamento?

Usando ajuda de métodos computacionais


Chromosome
Histones
Chromatin
Nucleosome
DNA helix
ATGACGGATCAGCCGCAAGCGGAATTGGCGACATAA
TACTGCCTAGTCGGCGTTCGCCTTAACCGCTGTATT

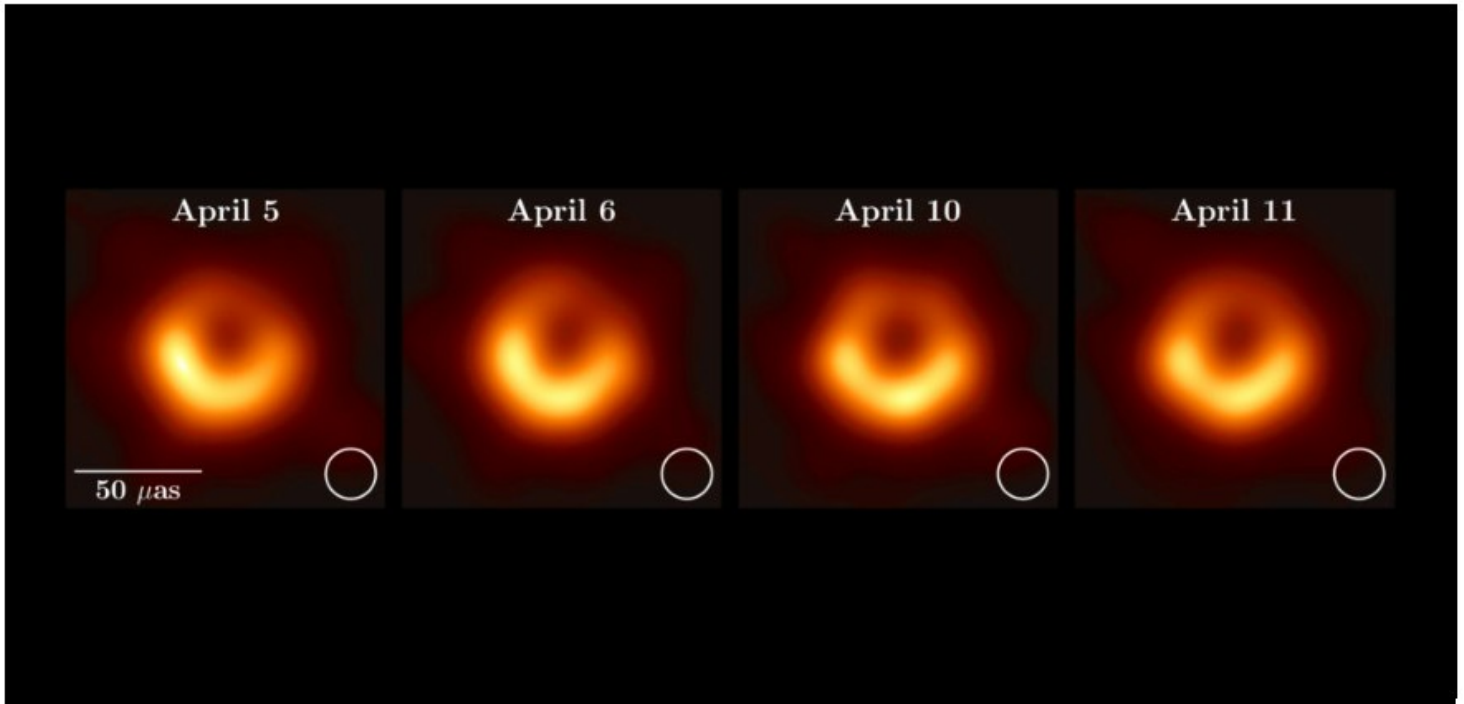


Como obter a sequência de DNA de uma espécie?

Usando ajuda de métodos computacionais!!

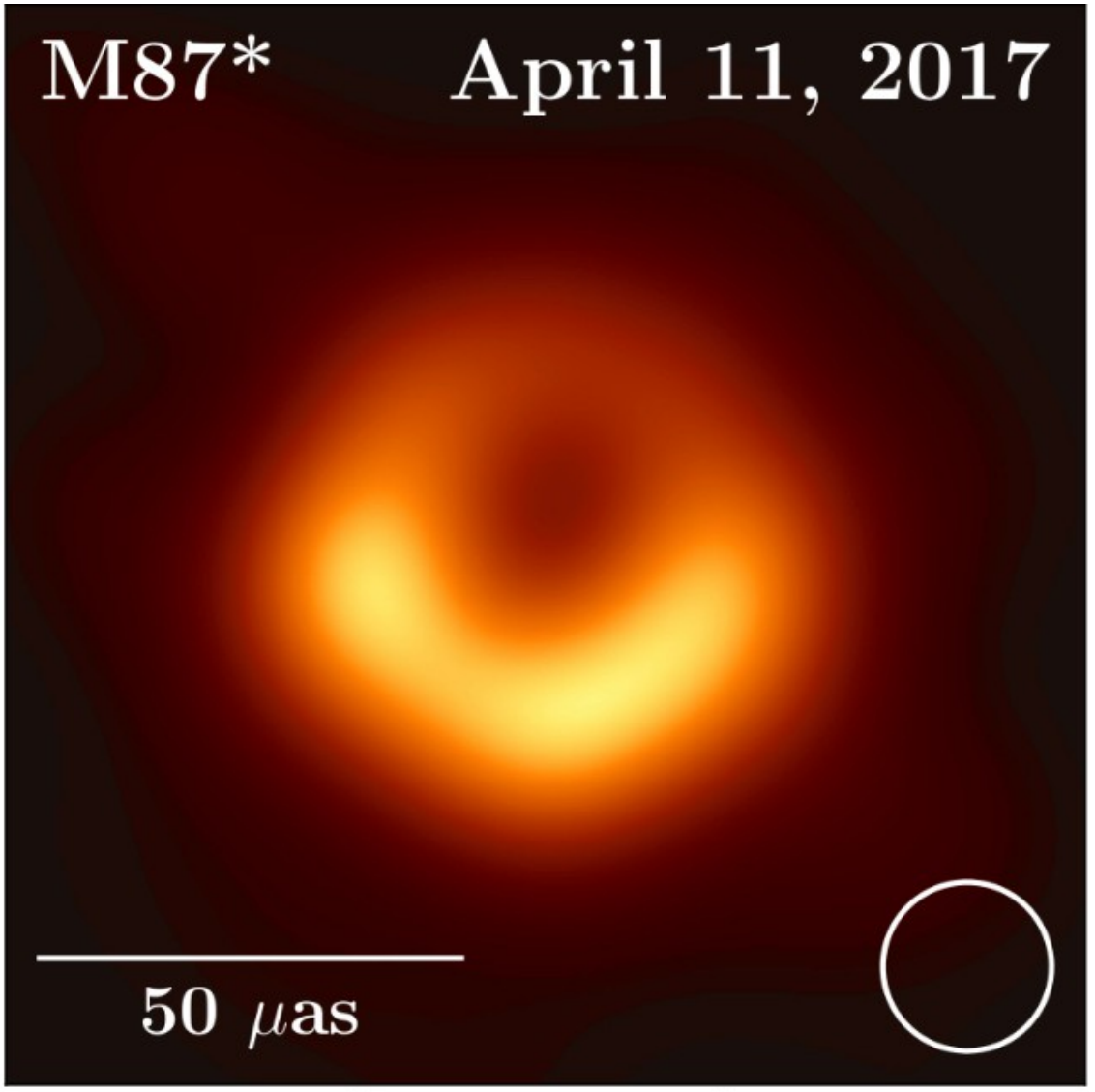
First Images of a Black Hole from the Event Horizon Telescope

By Susanna Kohler on 10 April 2019 FEATURES



Observations from the Event Horizon Telescope of the supermassive black hole at the center of the elliptical galaxy M87, for four different days. [EHT Collaboration et al 2019]

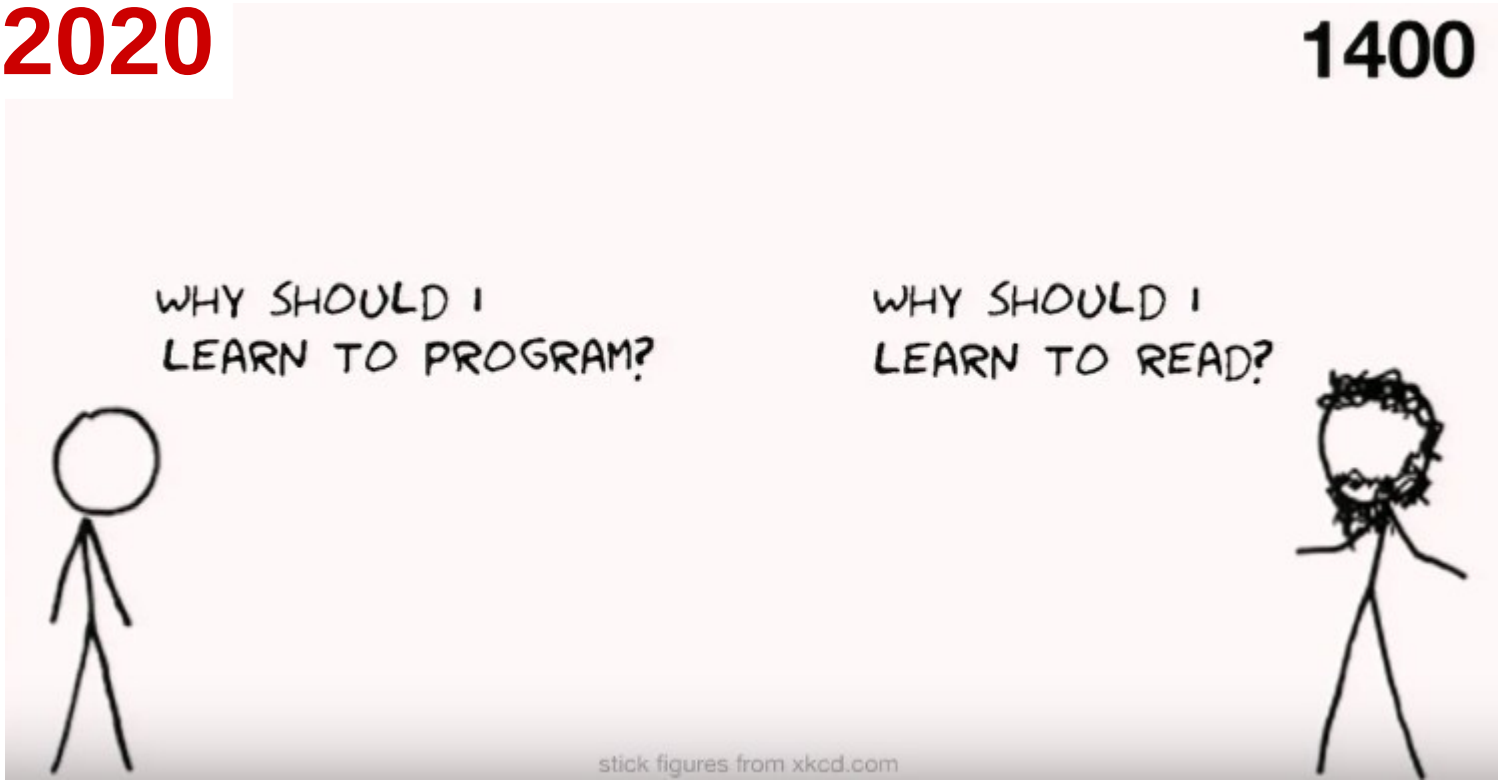
Astronomers have used a telescope that spans the globe to capture the first detailed images of a black hole: the nearby supermassive black hole in the Messier 87 galaxy. The first results from the Event Horizon Telescope (EHT) are detailed in six articles that make up a new Focus Issue in *The Astrophysical Journal Letters*.



Como obter uma fotografia de um buraco negro?

Usando ajuda de métodos computacionais!!!

2020



https://xkcd.com/

Coding Is the Must-Have Job Skill of the Future

7.1k
SHARES

2985 Share | 2372 Tweet | 403 Share | 1079 | 218 | 41

WHAT'S THIS?

Why Programming Is The Core Skill Of The 21st Century

It's never been easier, more accessible, or more essential to learn coding skills.

How Coding Can Boost Everyone's Career

"Everybody in this country should learn how to program a computer... because it teaches you how to think."

-Steve Jobs

https://www.youtube.com/watch?v=Dv7gLpW91DM

GETTY IMAGES

Programming languages give you much more freedom than you have with commercial applications.

SCIENTIFIC COMPUTING

Code alert

Programming tools can speed up and strengthen analyses, but mastering the skills takes time and can be daunting.

BY MONYA BAKER

Auriel Fournier had no choice but to learn programming. The ecology PhD student wanted to use a complex set of tutorials, mastered the package and now helps other researchers to make sense of R and similar tools. It can be "really intimidating" to learn a programming language, but the long-term benefits are well worth the effort, she says.

methodologist and psychometrician at the University of Amsterdam. "You can only do what the buttons say you can do." Programming languages are much broader in scope, he says. "In R or other programming languages you can do anything you want." But to achieve this broad latitude, researchers need to learn how to code — to formulate commands in a programming language that tell a computer what to do. Researchers who know just a bit of code gain access to a wealth of software packages that automate repetitive tasks and increase options for compiling, analysing, sharing and presenting data.

Coders often use R for doing statistics, whereas Python, which is often considered more intuitive, is good for repetitive tasks and for extracting data from web-based applications. Ruby provides more ways of doing the same tasks and is popular with web developers.

Learning to code is empowering and can hugely improve a researcher's career prospects. "But it does require an investment," says Fournier. And that includes not just time, but also an ongoing effort to assemble a community of helpful experts and to be willing to make, find and fix mistakes.

SMOOTH OPERATORS

Coding is becoming a crucial part of research, says Ethan White, an ecologist at the University of Florida, Gainesville, who has designed courses in quantitative methods. "When you do this well, your life is easier and your results can be regenerated," he points out. Scientists commonly use languages such as Python and R to conduct and automate analyses, because in this way they can speed data crunching, increase reproducibility, protect data from accidental deletion or alteration and handle data sets that would overwhelm commercial applications. Researchers who use these languages can tackle questions that would be impractical to address if data were manipulated manually, says White. Reconfiguring an analysis or revising graphs becomes quick and straightforward, and researchers can more easily build on their own or others' work.

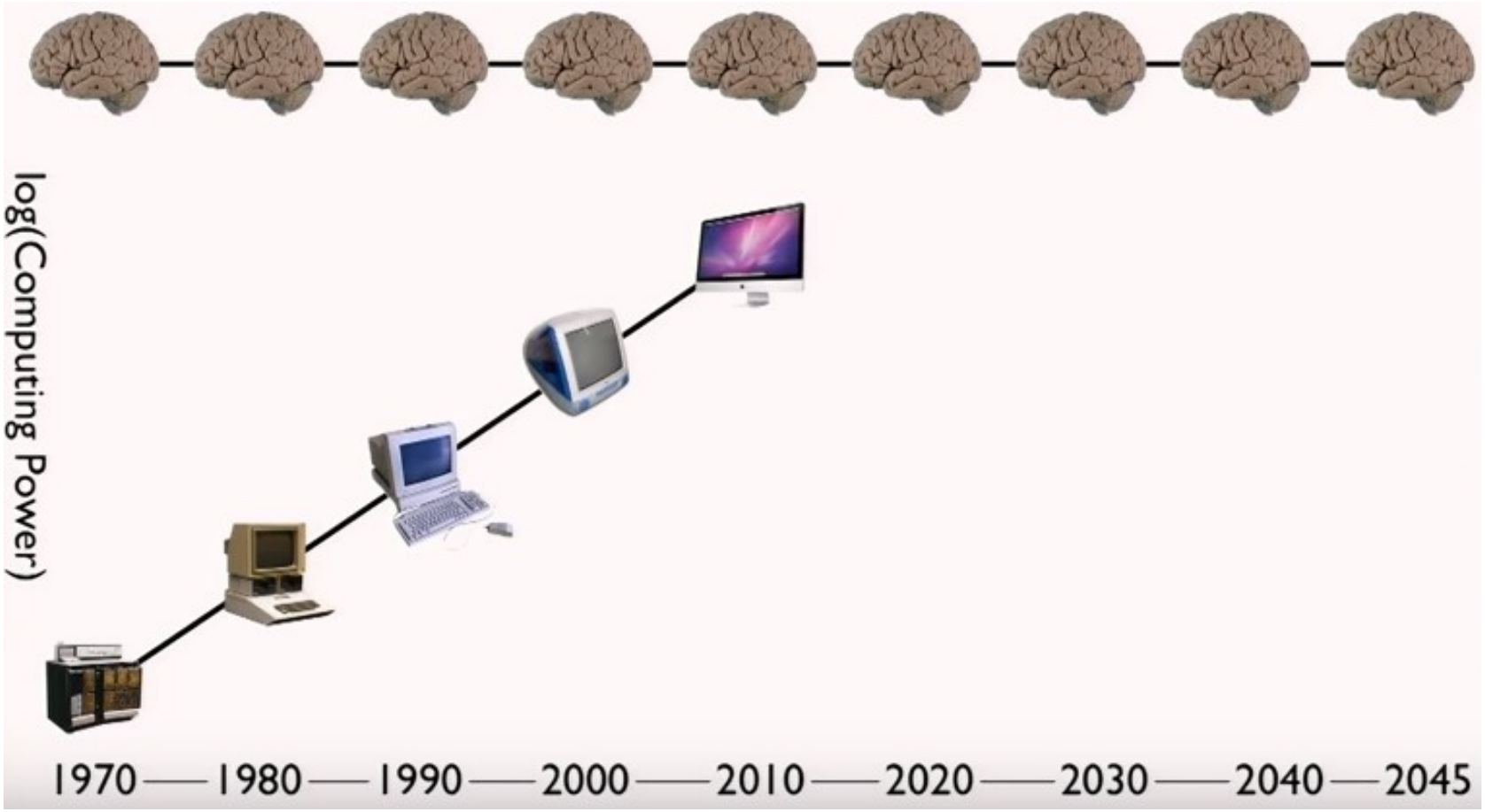
Andrew Durso can vouch for those upsides.

Preconceito sobre programador(a)?

Alunos e professores da UFABC

Programar?

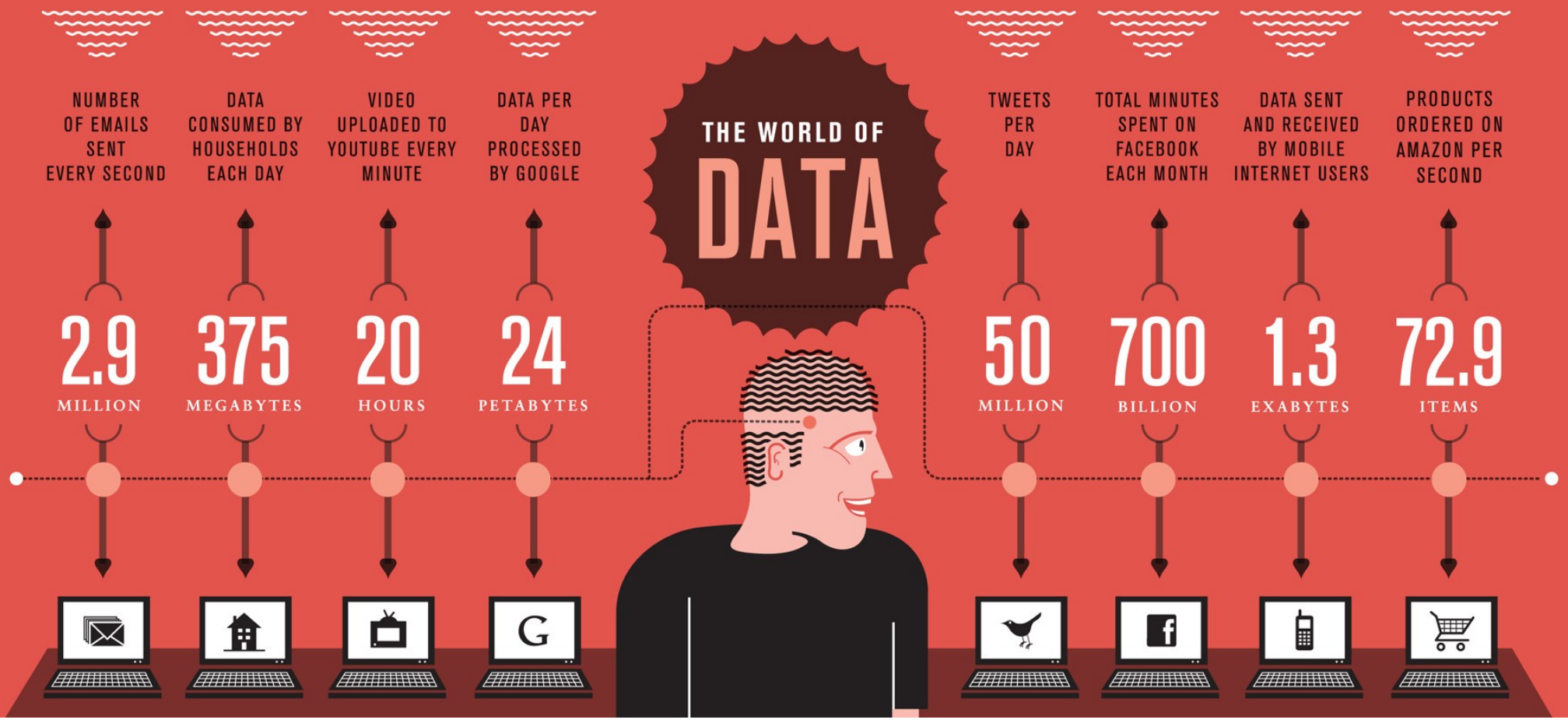
- Programar auxilia a **desenvolver** o **pensamento lógico**.
- Programar **lhe torna mais independente**.
- ...
- Programar te faz forte!



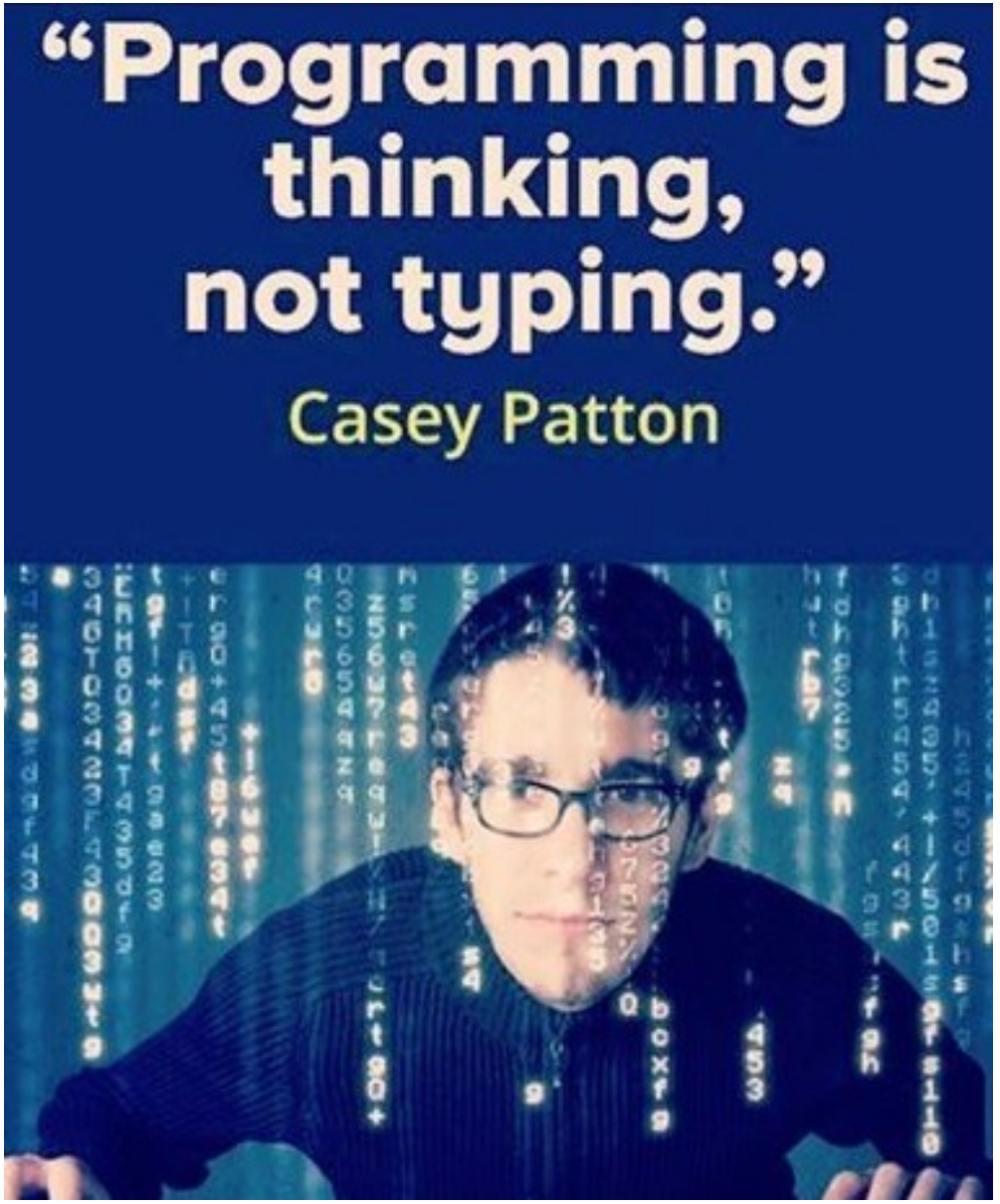
You Should Learn to Program: Christian Genco at TEDxSMU
https://www.youtube.com/watch?v=xfBWk4nw440

Grande escala?

http://blog.bimeanalytics.com/english/world-of-data-infographic



Em vez de a ciência não avançar devido à escassez de dados, hoje em dia ela frequentemente encontra **dificuldades em avançar por seu excesso**.

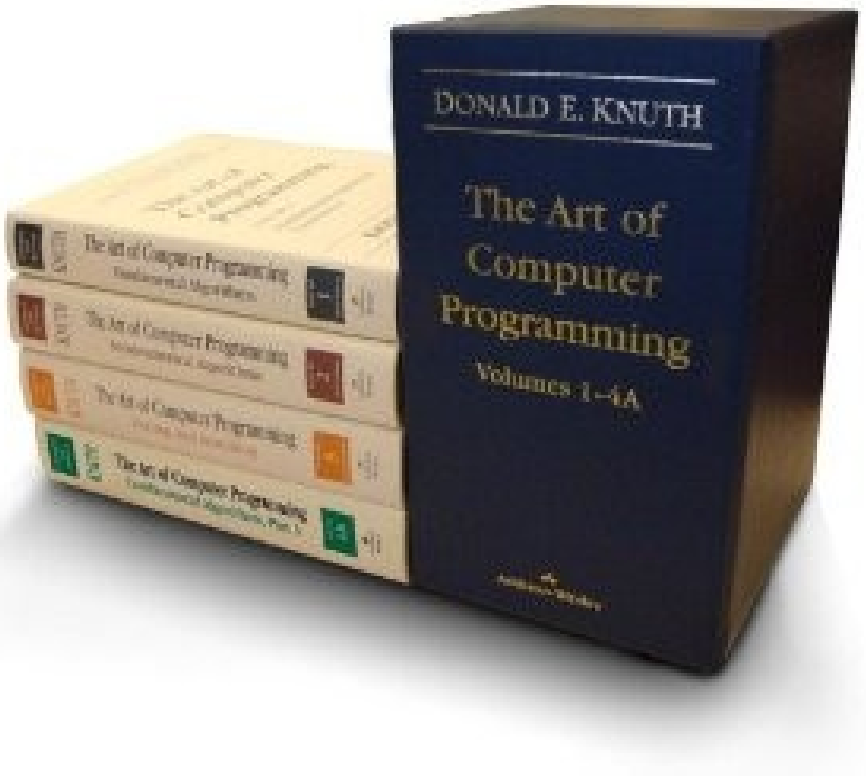

“Programming is thinking, not typing.”
Casey Patton

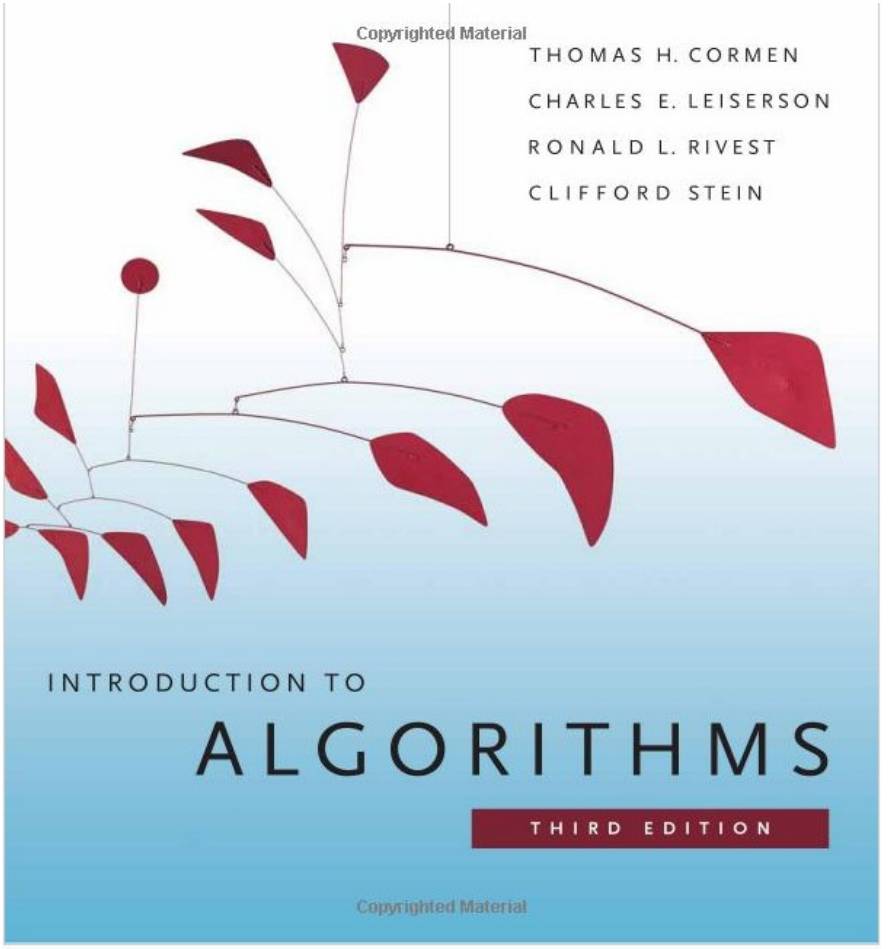
Programar?

Mas tem um custo:

- → **Tempo**
- → **Dedicação**

If you think you're a really good programmer… read [Knuth's] Art of Computer Programming… *You should definitely send me a résumé if you can read the whole thing.*
—Bill Gates





Graphs, Networks and Algorithms. Second Edition. Dieter Jungnickel.
An Introduction to the Theory of Numbers. Zuckerman y Montgomery.
Game Theory. Drew Fudenberg.
Theory of Games and Economic Behavior. John von Neumann, Oskar Morgenstern.

Sobre a disciplina

Objetivos

Objetivos:

- Apresentar os fundamentos sobre manipulação e tratamento da Informação, principalmente por meio da explicação e experimentação dos conceitos e do uso prático da lógica de programação.

Competências:

- Compreender os conceitos fundamentais a respeito da manipulação e tratamento da Informação.
- Entender a lógica de programação de computadores.
- Desenvolver algoritmos básicos para modelar e solucionar problemas de natureza técnico-científica.

Estudando programação

Combinação de teoria e prática de maneira inseparável.

- Análise de um problema
- Planejar a abordagem
- Implementar uma solução

Créditos (T-P-I)=(3-2-5)

Cada disciplina na UFABC é representada por três algarismos: T-P-I

- **T:** Número de horas semanais de aulas expositivas presenciais da disciplina (teóricas)
- **P:** Número médio de horas semanais de trabalho de laboratório, aulas práticas ou aulas de exercícios, realizadas em sala de aula (práticas)
- **I:** Estimativa de horas semanais adicionais de trabalhos necessárias para o bom aproveitamento da disciplina (estudos e trabalhos)

Páginas importantes

- http://professor.ufabc.edu.br/~jesus.mena/courses

Procurar: “2020/Q1”

- https://ava.ufabc.edu.br/

Procurar pelo nome de sua turma:

  - DA4-SA, DA5-SA, DA6-SA ou
  - DB4-SA, DB5-SA, DB6-SA

Calendário

Fevereiro

	Dom	Seg	Ter	Qua	Qui	Sex	Sáb
							1
	2	3	4	5	6	7	8
1	9	10	11	12	13	14	15
2	16	17	18	19	20	21	22
3	23	24	25	26	27	28	29

Março

	Dom	Seg	Ter	Qua	Qui	Sex	Sáb
4	1	2	3	4	5	6	7
5	8	9	10	11	12	13	14
6	15	16	17	18	19	20	21
7	22	23	24	25	26	27	28
8	29	30	31				

Abril

	Dom	Seg	Ter	Qua	Qui	Sex	Sáb
8				1	2	3	4
9	5	6	7	8	9	10	11
10	12	13	14	15	16	17	18
11	19	20	21	22	23	24	25
12	26	27	28	29	30		

Maio

	Dom	Seg	Ter	Qua	Qui	Sex	Sáb
12						1	2
13	3	4	5	6	7	8	9
14	10	11	12	13	14	15	16
	17	18	19	20	21	22	23
	24	25	26	27	28	29	30
	31						

Teoria
Prática

Agenda

Aula	Semana	Dia		Modalidade	Tópicos - Alinhamento
1	1	11/fev	Terça	Prática	Ambiente de trabalho e Programas sequenciais
2	1	12/fev	Quarta	Teoria - sem. I	Apresentação da discipciplina e Algoritmos sequenciais
3	1	14/fev	Sexta	Teoria	Modularização **e** Estruturas de seleção - Parte 1
4	2	18/fev	Terça	Prática	Estruturas de seleção - Parte 1
5	2	21/fev	Sexta	Teoria	Estruturas de seleção - Parte 2
6	3	28/fev	Sexta	Teoria	Estruturas de repetição - Parte 1
7	4	03/mar	Terça	Prática	Estruturas de seleção - Parte 2 / repetição Parte 1
8	4	06/mar	Sexta	Teoria	Estruturas de repetição - Parte 2
9	5	10/mar	Terça	Prática	Estruturas de repetição - Parte 2
10	5	11/mar	Quarta	Teoria - sem. I	Revisão
11	5	13/mar	Sexta	Teoria	**P1 - Teoria**
12	6	17/mar	Terça	Prática	**P1 - Prática**
13	6	20/mar	Sexta	Teoria	Vetores - Parte 1
14	7	24/mar	Terça	Prática	Vetores - Parte 1
15	7	25/mar	Quarta	Teoria - sem. I	Vetores - Parte 2
16	7	27/mar	Sexta	Teoria	Vetores - Parte 3
17	8	31/mar	Terça	Prática	Vetores - Partes 2 e 3
18	8	03/abr	Sexta	Teoria	Matrizes - Parte 1
19	9	07/abr	Terça	Prática	Matrizes - Parte 1a
20	10	14/abr	Terça	Prática	Matrizes - Parte 1b
21	10	17/abr	Sexta	Teoria	Matrizes - Parte 2
22	11	22/abr	Quarta	Teoria - sem. I	Recursão - Parte 1
23	11	24/abr	Sexta	Teoria	Recursão - Parte 2
24	12	28/abr	Terça	Prática	**Recursão**
25	Reposição	06/mai	Quarta	Prática	**P2 - Prática**
26	Reposição	07/mai	Quinta	Teoria	Exercícios
27	Reposição	08/mai	Sexta	Teoria	Revisão
28	Reposição	11/mai	Segunda	Teoria	**P2 - Teoria**
29	Reposição	13/mai	Quarta	Prática	**SUB - Prática**
30	Reposição	14/mai	Quinta	Teoria	**SUB – Teoria**
		20/jun	Sábado	Teoria+Prática	**REC – Sábado às 10h (sala a ser definida)**

Avaliação

Parte de Teoria: Duas provas.

- Prova 1 (50%): 13/03/2020
- Prova 2 (50%): 11/05/2020
- Bônus: 2 desafios de programação (10%)

- Prova Substitutiva: 14/05/2020
- Prova de Recuperação: Q3/2020 → 20/06/2020 às 10h

Avaliação

Ver o plano de ensino disponível na página da disciplina.

MF = 0,35 * Teoria + 0,35 * Prática + 0,3 Listas de exerc.

- **A:** nota ≥ 9
- **B:** 7,5 ≤ nota < 9
- **C:** 6 ≤ nota < 7,5
- **D:** 5,0 ≤ nota < 6
- **F:** nota < 5,0

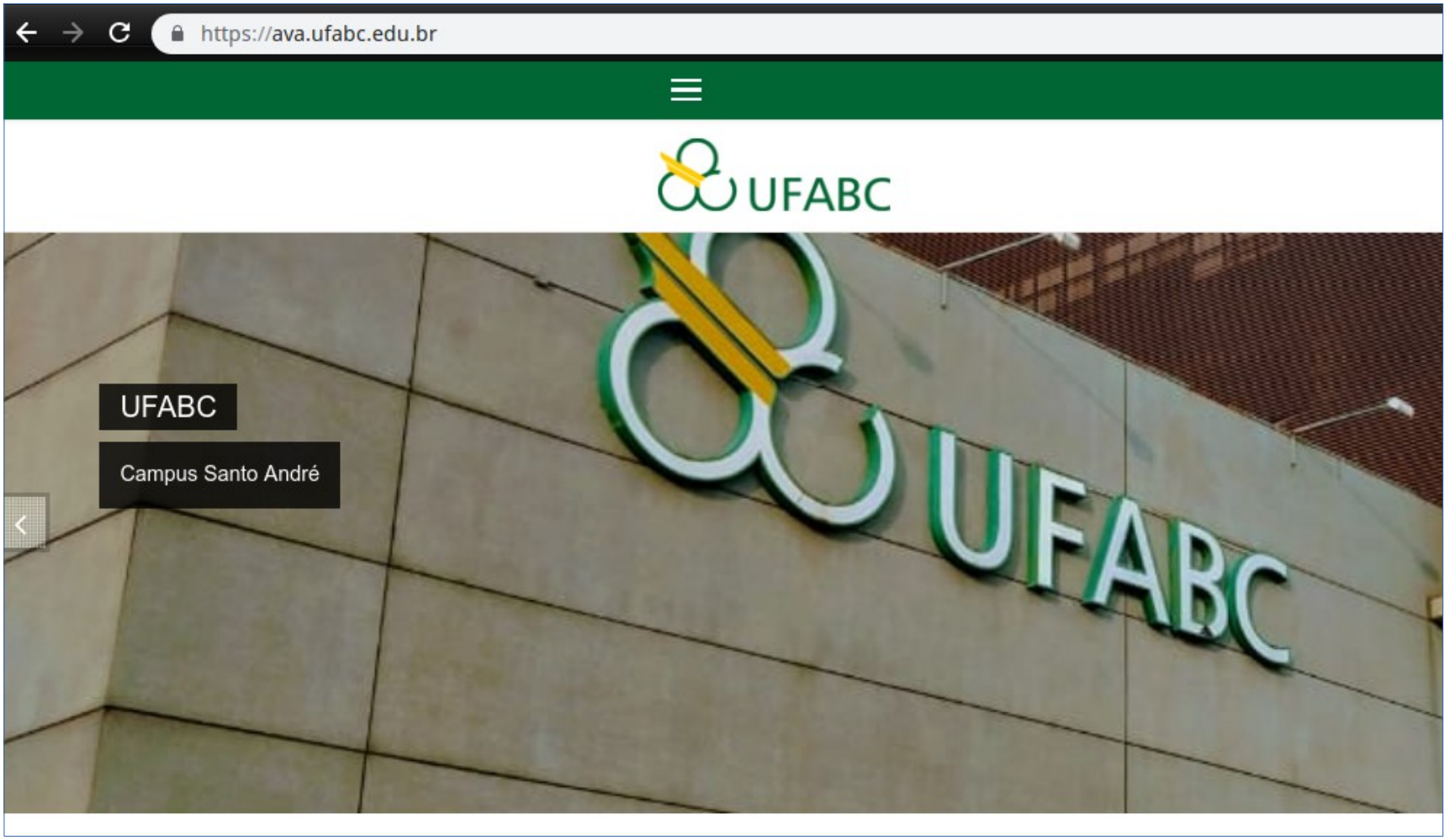
Listas de exercícios

- `Lista 1:  18/fev  Entrega: 03/mar`
- `Lista 2:  03/mar  Entrega: 09/mar`
- `Lista 3:  10/mar  Entrega: 16/mar`
- `Lista 4:  24/mar  Entrega: 30/mar`
- `Lista 5:  31/mar  Entrega: 06/abr`
- `Lista 6:  07/abr  Entrega: 13/abr`
- `Lista 7:  14/abr  Entrega: 27/mar`
- `Lista 8:  28/abr  Entrega: 12/mai`

A nota das listas é média aritmética dos exercícios.

Listas para treino (não valen nota no lab)

Temos adicionalmente Exercícios-Programa (EPs) que serão considerados para treino: https://ava.ufabc.edu.br



Bibliografia

Bibliografia Básica

- Forbellone, A. L. V.; Eberspächer, H. F.; Lógica de Programação - A Construção de Algoritmos e Estruturas de Dados; 3ª edição, Editora Pearson Prentice-Hall, 2005.
- Sebesta, R. W.; Conceitos de Linguagens de Programação; 5ª edição, Editora Bookman, 2003.

Bibliografia Complementar

- Ascensio, A.F.; Campos, E.A., Fundamentos da Programação de Computadores, Pearson, 3a edição, 2012.
- Puga, S., Lógica de programação e estruturas de dados com aplicações em Java, Pearson Prentice-Hall, 2a edição, 2009.