# Apresentação do Curso

## BCM0505-15 – Processamento da Informação

**Prof. Maycon Sambinelli**

m.sambinelli@ufabc.edu.br

Centro de Matemática, Computação e Cognição – Universidade Federal do ABC

Sejam bem-vindos!

# Sobre Mim

- ✉ **E-mail:** m.sambinelli@ufabc.edu.br
- 🏢 **Sala:** 518-2, Bloco A, Torre 2
- 💻 **Page:** `http://professor.ufabc.edu.br/~m.sambinelli`

# Introdução

# Objetivos do Curso

Introdução a

- Algoritmos
- Programação de Computadores

# Algoritmos

Um **algoritmo** é uma sequência finita de passos bem definidos que descrevem como executar uma determinada tarefa

# Algoritmos

Um **algoritmo** é uma sequência finita de passos bem definidos que descrevem como executar uma determinada tarefa

- Receita de bolo

# Algoritmos

Um **algoritmo** é uma sequência finita de passos bem definidos que descrevem como executar uma determinada tarefa

- Receita de bolo
- Fórmula da equação quadrática (Bhaskara)

# Algoritmos

Um **algoritmo** é uma sequência finita de passos bem definidos que descrevem como executar uma determinada tarefa

- Receita de bolo
- Fórmula da equação quadrática (Bhaskara)
- Multiplicação de dois números

# Algoritmos

Um **algoritmo** é uma sequência finita de passos bem definidos que descrevem como executar uma determinada tarefa

- Receita de bolo
- Fórmula da equação quadrática (Bhaskara)
- Multiplicação de dois números
- Partitura

# Algoritmos

Um **algoritmo** é uma sequência finita de passos bem definidos que descrevem como executar uma determinada tarefa

- Receita de bolo
- Fórmula da equação quadrática (Bhaskara)
- Multiplicação de dois números
- Partitura

# Algoritmos

Um **algoritmo** é uma sequência finita de passos bem definidos que descrevem como executar uma determinada tarefa

- Receita de bolo
- Fórmula da equação quadrática (Bhaskara)
- Multiplicação de dois números
- Partitura

Algoritmo é a "ideia/método" de como fazer a tarefa

# Programação de Computadores

Descrever, para um computador, como realizar uma tarefa

# Programação de Computadores

Descrever, para um computador, como realizar uma tarefa

Português (Inglês e etc) não é adequado para programar computadores

# Programação de Computadores

Descrever, para um computador, como realizar uma tarefa

Português (Inglês e etc) não é adequado para programar computadores

- Ambígua

# Programação de Computadores

Descrever, para um computador, como realizar uma tarefa

Português (Inglês e etc) não é adequado para programar computadores

- Ambígua
- Gramaticalmente complexa

# Programação de Computadores

Descrever, para um computador, como realizar uma tarefa

Português (Inglês e etc) não é adequado para programar computadores

- Ambígua
- Gramaticalmente complexa
- Muitas exceções

# Programação de Computadores

Descrever, para um computador, como realizar uma tarefa

Português (Inglês e etc) não é adequado para programar computadores

- Ambígua
- Gramaticalmente complexa
- Muitas exceções

# Programação de Computadores

Descrever, para um computador, como realizar uma tarefa

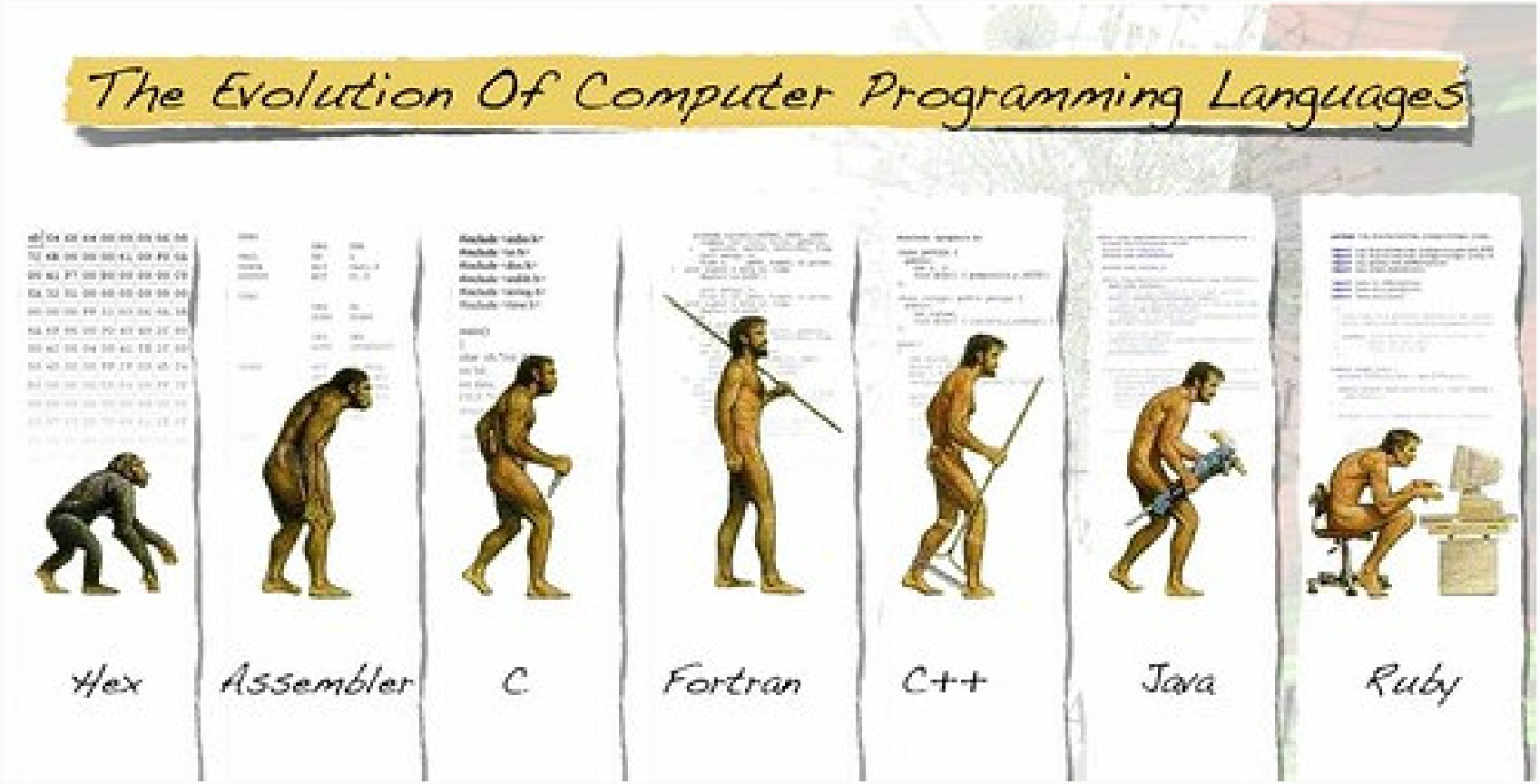
Português (Inglês e etc) não é adequado para programar computadores

- Ambígua
- Gramaticalmente complexa
- Muitas exceções

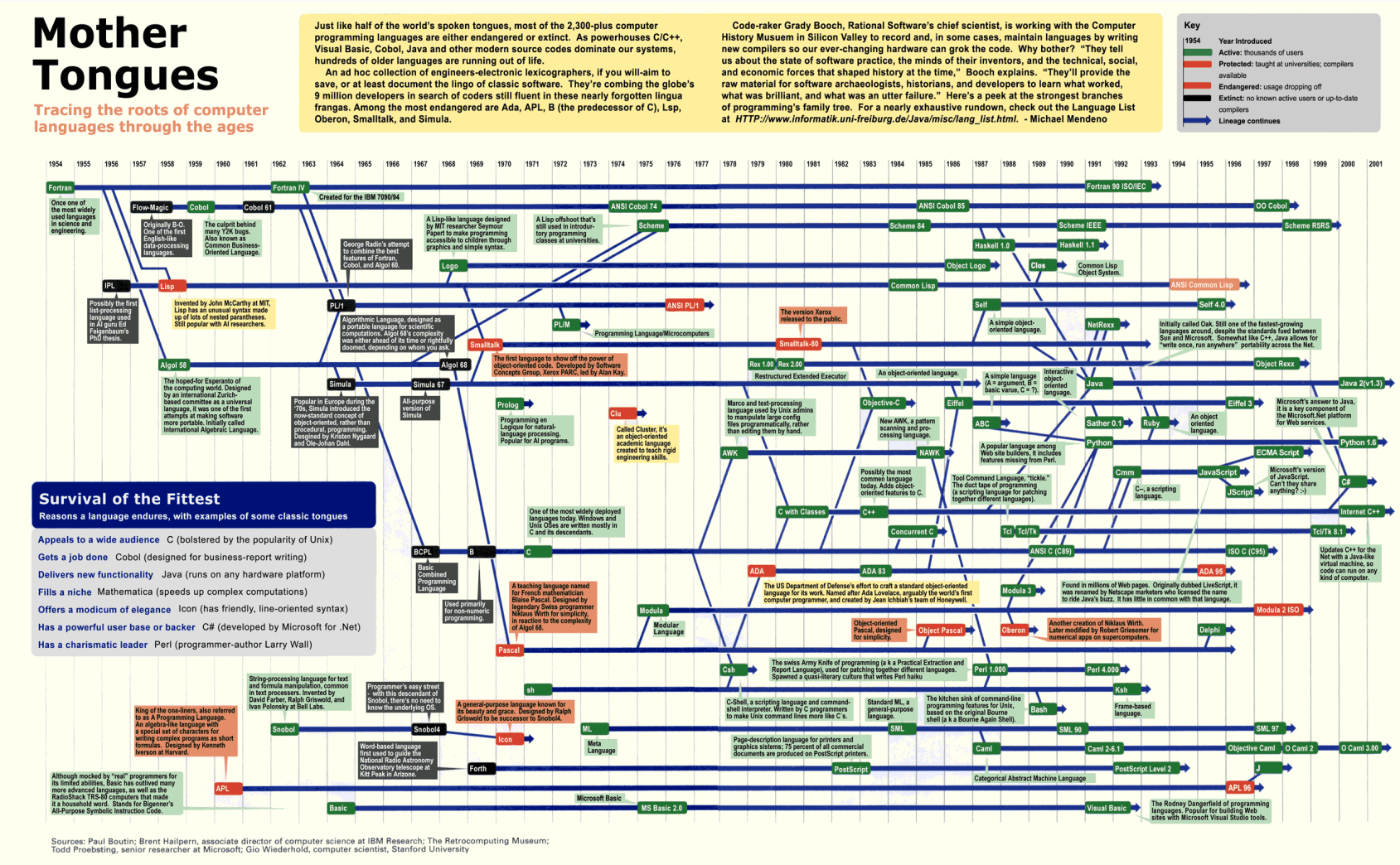
Solução: criar uma linguagem com esse fim

# Linguagens de Programação

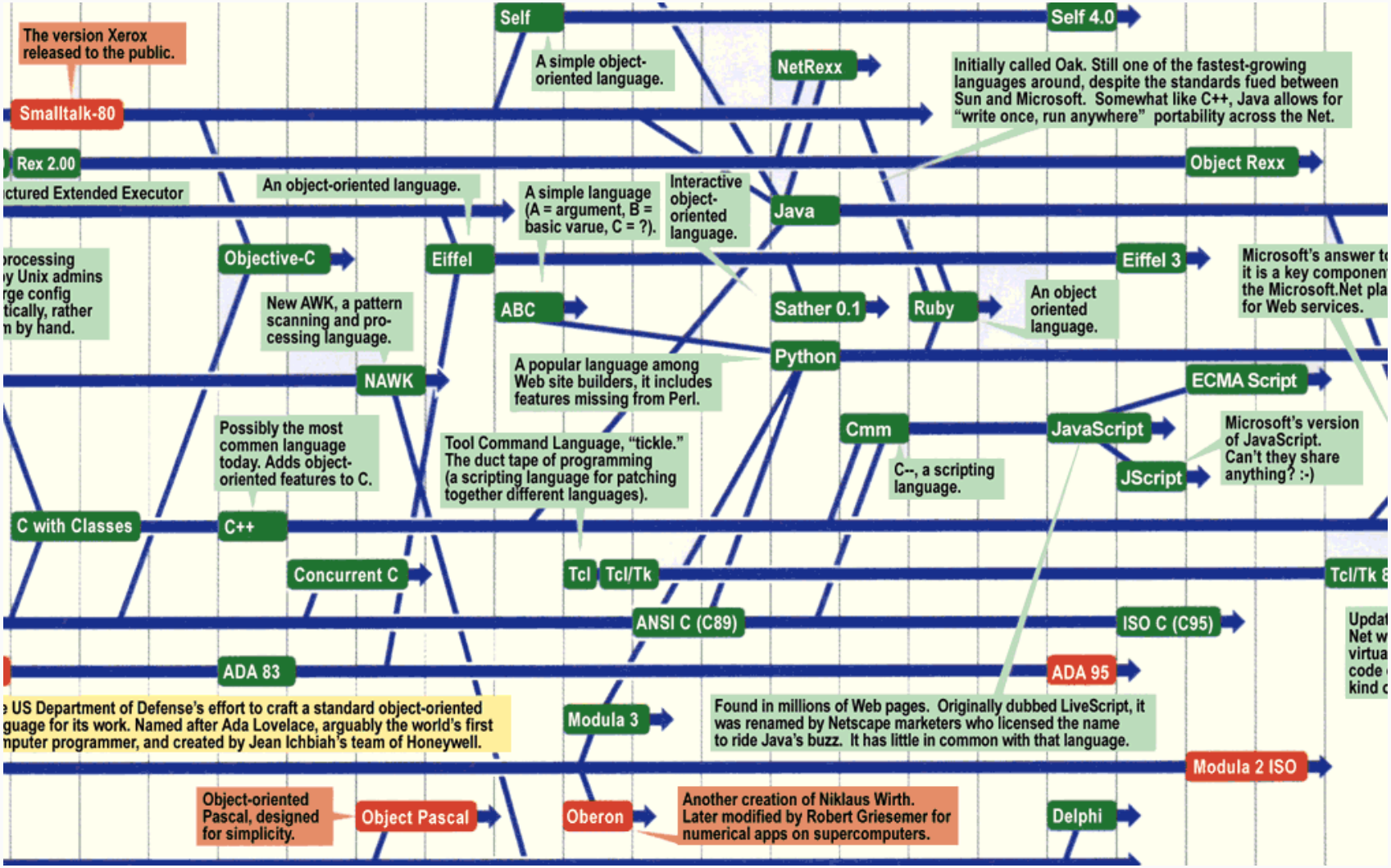
# Evolução das Linguagens de Programação



# Evolução das Linguagens de Programação



# Evolução das Linguagens de Programação



# Linguagens Mais Utilizadas

| Rank | Language | Type | Score |
|---|---|---|---|
| 1 | Python | | 100.0 |
| 2 | Java | | 96.3 |
| 3 | C | | 94.4 |
| 4 | C++ | | 87.5 |
| 5 | R | | 81.5 |
| 6 | JavaScript | | 79.4 |
| 7 | C# | | 74.5 |
| 8 | Matlab | | 70.6 |
| 9 | Swift | | 69.1 |
| 10 | Go | | 68.0 |

Fonte: https://spectrum.ieee.org/computing/software/the-top-programming-languages-2019

# Problema: resolver equação $ax^2 + bx + c = 0$

## Algoritmo

- calcule o discriminante: $\Delta = b^2 - 4ac$
- determine a quantidade de raízes: se $\Delta = 0$, existe apenas uma; se $\Delta < 0$, existem duas raízes imaginárias; caso contrário, existem duas raízes reais
- calcule a(s) raiz(es): $\frac{-b \pm \sqrt{\Delta}}{2a}$

# Problema: resolver equação $ax^2 + bx + c = 0$

## Algoritmo

- calcule o discriminante: $\Delta = b^2 - 4ac$
- determine a quantidade de raízes: se $\Delta = 0$, existe apenas uma; se $\Delta < 0$, existem duas raízes imaginárias; caso contrário, existem duas raízes reais
- calcule a(s) raiz(es): $\frac{-b \pm \sqrt{\Delta}}{2a}$

## Algoritmo em Pseudocódigo

**leia** $a$
**leia** $b$
**leia** $c$
$\Delta \leftarrow b^2 - 4ac$
**se** $\Delta > 0$ **então**
    $r_1 \leftarrow \frac{-b+\sqrt{\Delta}}{2a}$
    $r_2 \leftarrow \frac{-b-\sqrt{\Delta}}{2a}$
    **escreva** "Raizes: $r_1$, $r_2$"
**senão se** $\Delta = 0$ **então**
    $r \leftarrow \frac{-b}{2a}$
    **escreva** "Raiz: r"
**senão**
    **escreva** "Não há raizes reais."

# Problema: resolver equação $ax^2 + bx + c = 0$

## Algoritmo em Pseudocódigo

**leia** $a$

**leia** $b$

**leia** $c$

$\Delta \leftarrow b^2 - 4ac$

**se** $\Delta > 0$ **então**

    $r_1 \leftarrow \frac{-b+\sqrt{\Delta}}{2a}$

    $r_2 \leftarrow \frac{-b-\sqrt{\Delta}}{2a}$

    **escreva** "Raizes: $r_1$, $r_2$"

**senão se** $\Delta = 0$ **então**

    $r \leftarrow \frac{-b}{2a}$

    **escreva** "Raiz: r"

**senão**

    **escreva** "Não há raizes reais."

## Python

```python
from math import sqrt

a = float(input())
b = float(input())
c = float(input())
D = b**2 - 4*a*c

if D > 0:
  r1 = (((-b) + sqrt(D))/(2*a))
  r2 = (((-b) - sqrt(D))/(2*a))
  print("Raizes: %f, %f" % (r1, r2))
elif D == 0:
  r = (-b) / 2*a
  print("Raiz: ", x)
else:
  print("Não há raizes reais.")
```

# Problema: resolver equação $ax^2 + bx + c = 0$

## Algoritmo em Pseudocódigo

**leia** $a$
**leia** $b$
**leia** $c$
$\Delta \leftarrow b^2 - 4ac$
**se** $\Delta > 0$ **então**
  $r_1 \leftarrow \frac{-b+\sqrt{\Delta}}{2a}$
  $r_2 \leftarrow \frac{-b-\sqrt{\Delta}}{2a}$
  **escreva** "Raizes: $r_1$, $r_2$"
**senão se** $\Delta = 0$ **então**
  $r \leftarrow \frac{-b}{2a}$
  **escreva** "Raiz: r"
**senão**
  **escreva** "Não há raizes reais."

## Java

```java
import java.util.Scanner;
public class EquacaoSegundoGrau {
  public static void main(String[] Strings) {
    Scanner input = new Scanner(System.in);
    double a = input.nextDouble();
    double b = input.nextDouble();
    double c = input.nextDouble();
    double D = b * b - 4 * a * c;
    double r1, r2;

    if (D > 0.0) {
      r1 = (-b + Math.pow(D, 0.5)) / (2.0 * a);
      r2 = (-b - Math.pow(D, 0.5)) / (2.0 * a);
      System.out.println("Raizes: " + r1 + ", " +
      ↪  r2);
    } else if (D == 0.0) {
      r1 = -b / (2.0 * a);
      System.out.println("Raiz: " + r1);
    } else {
      System.out.println("Sem raizes reais.");
    }
  }
}
```

# Problema: resolver equação $ax^2 + bx + c = 0$

## Algoritmo em Pseudocódigo

**leia** $a$
**leia** $b$
**leia** $c$
$\Delta \leftarrow b^2 - 4ac$
**se** $\Delta > 0$ **então**
  $r_1 \leftarrow \frac{-b+\sqrt{\Delta}}{2a}$
  $r_2 \leftarrow \frac{-b-\sqrt{\Delta}}{2a}$
  **escreva** "Raizes: $r_1$, $r_2$"
**senão se** $\Delta = 0$ **então**
  $r \leftarrow \frac{-b}{2a}$
  **escreva** "Raiz: r"
**senão**
  **escreva** "Não há raizes reais."

```c
#include <stdio.h>
#include <math.h>

int main() {
    double a, b, c;
    scanf("%lf %lf %lf", &a, &b, &c);
    double D = b * b - 4 * a * c;
    double r1, r2;

    if (D > 0.0) {
        r1 = (-b + sqrt(D)) / (2.0 * a);
        r2 = (-b - sqrt(D)) / (2.0 * a);
        printf("Raizes: %lf, %lf\n", r1,
        ↪  r2);
    } else if (D == 0.0) {
        r1 = -b / (2.0 * a);
        printf("Raiz: %lf\n", r1);
    } else {
        printf("Não há raizes reais.\n");
    }
}
```

# Por que aprender a programar?

Automatizar tarefas monótonas no seu computador:

- planilhas
- extrair informações da web
- tratar dados de forma automatizada

# Por que aprender a programar?

Automatizar tarefas monótonas no seu computador:

- planilhas
- extrair informações da web
- tratar dados de forma automatizada

*"Todo bom programador é preguiçoso"* – Larry Wall

# Por que aprender a programar?

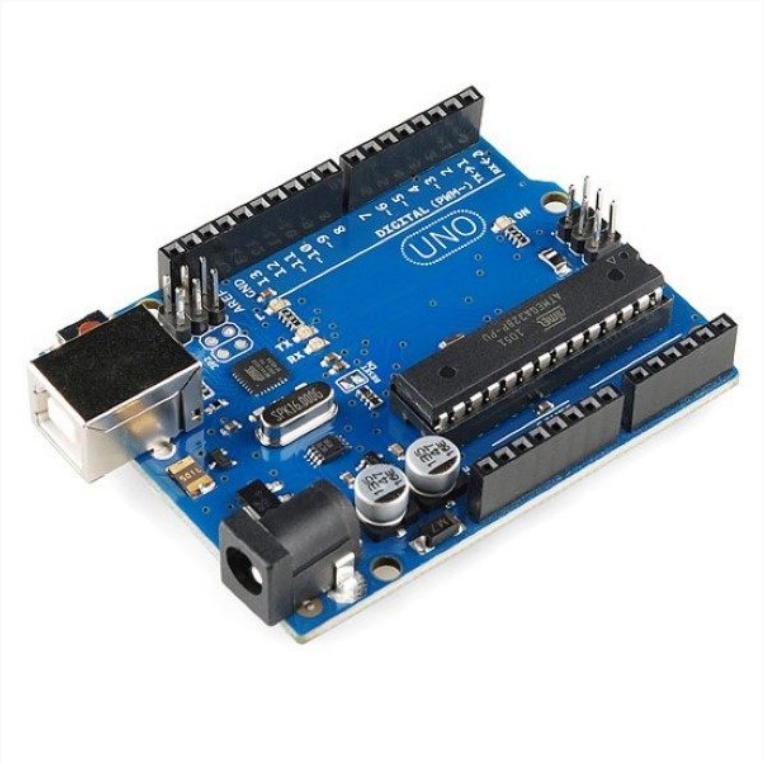
Automatizar tarefas monótonas no seu computador:

- planilhas
- extrair informações da web
- tratar dados de forma automatizada

*"Todo bom programador é preguiçoso"* – Larry Wall

Automatizar tarefas no mundo físico com Arduino

# Arduino

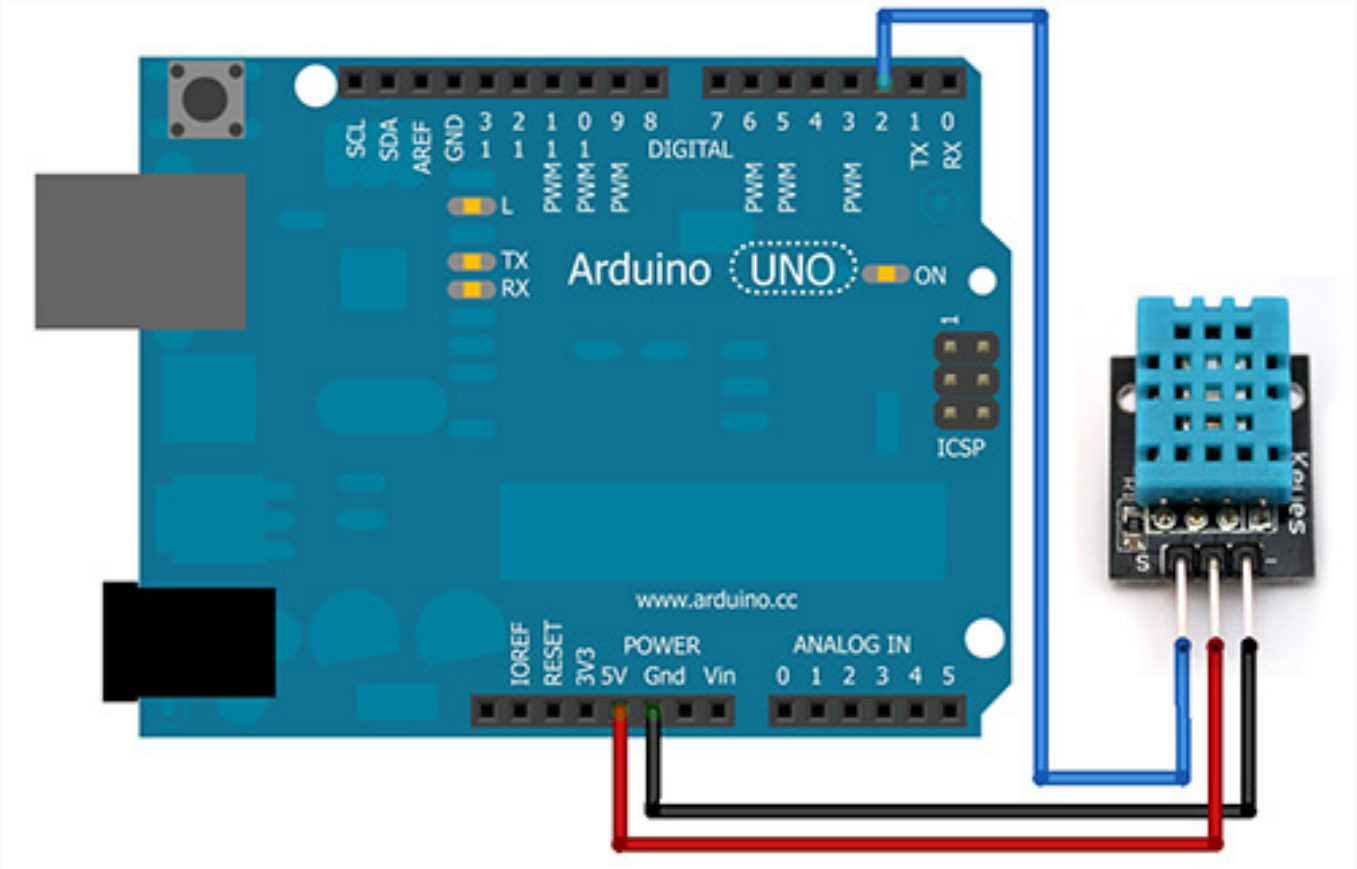


R$ 40,00

```
#include <LiquidCrystal.h>
#include "DHT.h"

LiquidCrystal lcd(12, 11, 5, 4,
↪  3, 2);

#define DHTTYPE DHT11
#define DHTPIN 8
DHT dht(DHTPIN, DHTTYPE);

void loop() {
  delay(500);
  lcd.setCursor(0, 1);
  float h = dht.readHumidity();
  float f =
  ↪  dht.readTemperature(true);

  if (isnan(h) || isnan(f)) {
    lcd.print("ERROR");
    return;
  }

  lcd.print(f);
  lcd.setCursor(7,1);
  lcd.print(h);
}
```

# Arduino: Sensores e Módulos

# Arduino: Sensores e Módulos



# Sobre as aulas

# Sobre as Aulas

## Aulas Teóricas

Jesús P. Mena-Chalco

- Quartas 8h-10h (Quinzenal I) -- Sala A-105-0
- Sextas 10h-12h -- Sala A-105-0

## Aulas Práticas

Maycon Sambinelli

- Terças 8h-10h -- Sala L603

# Critério de Avaliação

$$MF = .35 \times T + .35 \times P + .3 \times L$$

- $MF$ é a média final antes da REC

# Critério de Avaliação

$$MF = .35 \times T + .35 \times P + .3 \times L$$

- $MF$ é a média final antes da REC
- $T$ é a média aritmética da nota das duas provas teóricas

# Critério de Avaliação

$$MF = .35 \times T + .35 \times P + .3 \times L$$

- $MF$ é a média final antes da REC
- $T$ é a média aritmética da nota das duas provas teóricas
- $P$ é a média aritmética da nota das duas provas práticas

# Critério de Avaliação

$$MF = .35 \times T + .35 \times P + .3 \times L$$

- $MF$ é a média final antes da REC
- $T$ é a média aritmética da nota das duas provas teóricas
- $P$ é a média aritmética da nota das duas provas práticas
- $L$ é a médias aritmética da nota das listas de exercícios

# Critério de Avaliação

$$MF = .35 \times T + .35 \times P + .3 \times L$$

- $MF$ é a média final antes da REC
- $T$ é a média aritmética da nota das duas provas teóricas
- $P$ é a média aritmética da nota das duas provas práticas
- $L$ é a médias aritmética da nota das listas de exercícios
- $(0 \leq T, P, L \leq 10)$

# Critério de Avaliação

$$MF = .35 \times T + .35 \times P + .3 \times L$$

- $MF$ é a média final antes da REC
- $T$ é a média aritmética da nota das duas provas teóricas
- $P$ é a média aritmética da nota das duas provas práticas
- $L$ é a médias aritmética da nota das listas de exercícios
- $(0 \leq T, P, L \leq 10)$

# Critério de Avaliação

$$MF = .35 \times T + .35 \times P + .3 \times L$$

- $MF$ é a média final antes da REC
- $T$ é a média aritmética da nota das duas provas teóricas
- $P$ é a média aritmética da nota das duas provas práticas
- $L$ é a médias aritmética da nota das listas de exercícios
- $(0 \leq T, P, L \leq 10)$

Seu conceito final será:

- $A$, se $MF \geq 9.0$
- $B$, se $7.5 \leq MF < 9.0$
- $C$, se $6.0 \leq MF < 7.5$
- $D$, se $5.0 \leq MF < 6.0$
- $F$, se $MF < 5.0$
- $O$, se ausência total exceder 25%

# Mecanismo de Recuperação

$$MFR = \max\{MF, (MF + NR)/2\}$$

- $MFR$ é a sua média final com a REC
- $NR$ é a nota da REC

# Mecanismo de Recuperação

$$MFR = \max\{MF, (MF + NR)/2\}$$

- $MFR$ é a sua média final com a REC
- $NR$ é a nota da REC

- Alunos com $D$ ou $F$ têm direito a REC

# Listas de Exercícios

- Listas de exercícios semanais
- Página: `https://ava.ufabc.edu.br/`
- Correção Automática

# Plágio

# Plágio

Regra 1 Você não pode enviar para avaliação um trabalho que não seja de sua própria autoria ou que seja derivado/baseado em soluções elaboradas por outros.

# Plágio

Regra 1 Você não pode enviar para avaliação um trabalho que não seja de sua própria autoria ou que seja derivado/baseado em soluções elaboradas por outros.

Regra 2 Você não pode compartilhar a sua solução com outros alunos nem pedir aos seus colegas que compartilhem as soluções deles com você.

# Plágio

Regra 1 Você não pode enviar para avaliação um trabalho que não seja de sua própria autoria ou que seja derivado/baseado em soluções elaboradas por outros.

Regra 2 Você não pode compartilhar a sua solução com outros alunos nem pedir aos seus colegas que compartilhem as soluções deles com você.

Regra 3 Nos trabalhos enviados para avaliação você deve indicar eventuais assistências que você tenha recebido.

# Plágio

Regra 1 Você não pode enviar para avaliação um trabalho que não seja de sua própria autoria ou que seja derivado/baseado em soluções elaboradas por outros.

Regra 2 Você não pode compartilhar a sua solução com outros alunos nem pedir aos seus colegas que compartilhem as soluções deles com você.

Regra 3 Nos trabalhos enviados para avaliação você deve indicar eventuais assistências que você tenha recebido.

- Nós encorajamos fortemente que você procure outras pessoas quando houver a necessidade. Discuta o problema e possíveis ideias para soluções, mas elabore sua própria solução, por conta própria.

# Plágio

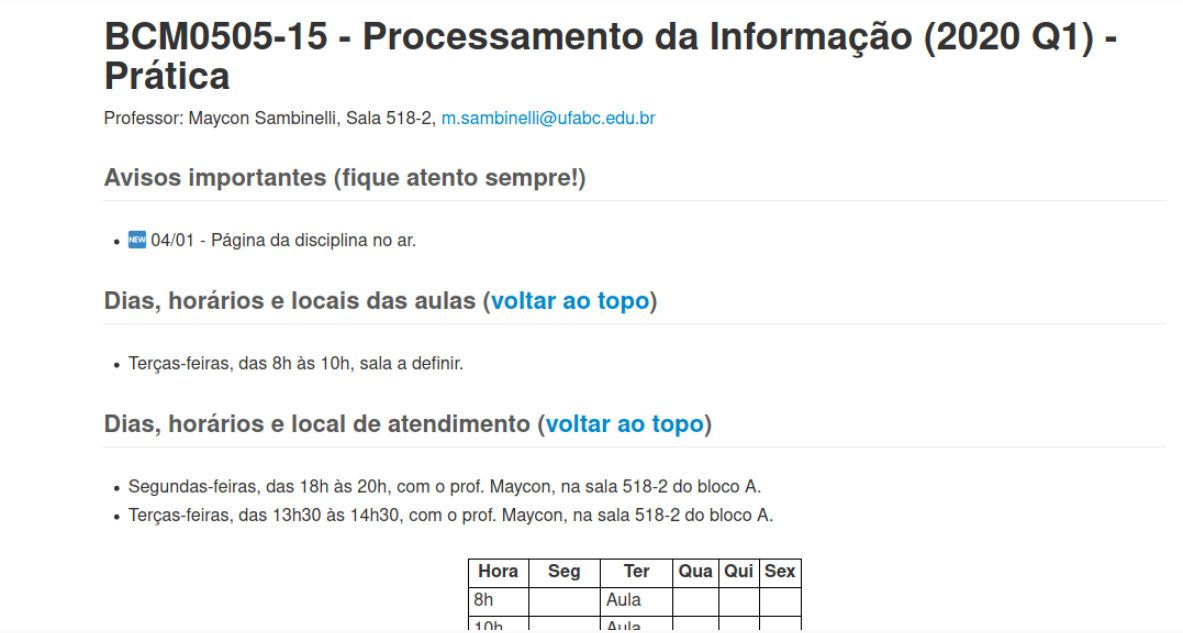
Regra 1 Você não pode enviar para avaliação um trabalho que não seja de sua própria autoria ou que seja derivado/baseado em soluções elaboradas por outros.

Regra 2 Você não pode compartilhar a sua solução com outros alunos nem pedir aos seus colegas que compartilhem as soluções deles com você.

Regra 3 Nos trabalhos enviados para avaliação você deve indicar eventuais assistências que você tenha recebido.

- Nós encorajamos fortemente que você procure outras pessoas quando houver a necessidade. Discuta o problema e possíveis ideias para soluções, mas elabore sua própria solução, por conta própria.
- **Qualquer violação às regras descritas acima implicará em descarte dos conceitos atribuídos a TODAS as tarefas avaliativas regulares de TODOS os envolvidos, causando assim suas reprovações automáticas com conceito F.**

# Página da Disciplina



## BCM0505-15 - Processamento da Informação (2020 Q1) - Prática

Professor: Maycon Sambinelli, Sala 518-2, m.sambinelli@ufabc.edu.br

### Avisos importantes (fique atento sempre!)

- NEW 04/01 - Página da disciplina no ar.

### Dias, horários e locais das aulas (voltar ao topo)

- Terças-feiras, das 8h às 10h, sala a definir.

### Dias, horários e local de atendimento (voltar ao topo)

- Segundas-feiras, das 18h às 20h, com o prof. Maycon, na sala 518-2 do bloco A.
- Terças-feiras, das 13h30 às 14h30, com o prof. Maycon, na sala 518-2 do bloco A.

| Hora | Seg | Ter | Qua | Qui | Sex |
|---|---|---|---|---|---|
| 8h | | Aula | | | |
| 10h | | Aula | | | |

http://professor.ufabc.edu.br/~m.sambinelli/courses/2020Q1-PI/

# Conselhos

# Tirando um A

- Compareça as aulas (teóricas e práticas)

# Tirando um A

- Compareça as aulas (teóricas e práticas)
- Faça as listas de exercícios

# Tirando um A

- Compareça as aulas (teóricas e práticas)
- Faça as listas de exercícios
- Use os horários de atendimento (procure ajuda)

# Tirando um A

- Compareça as aulas (teóricas e práticas)
- Faça as listas de exercícios
- Use os horários de atendimento (procure ajuda)
- Respeite os horário

# Tirando um A

- Compareça as aulas (teóricas e práticas)
- Faça as listas de exercícios
- Use os horários de atendimento (procure ajuda)
- Respeite os horário
- Planeje seus estudos

# Tirando um A

- Compareça as aulas (teóricas e práticas)
- Faça as listas de exercícios
- Use os horários de atendimento (procure ajuda)
- Respeite os horário
- Planeje seus estudos
    - Revise o material da aula teórica no dia seguinte

# Tirando um A

- Compareça as aulas (teóricas e práticas)
- Faça as listas de exercícios
- Use os horários de atendimento (procure ajuda)
- Respeite os horário
- Planeje seus estudos
  - Revise o material da aula teórica no dia seguinte
  - Fazer a lista + tempo para tirar dúvida

# Tirando um A

- Compareça as aulas (teóricas e práticas)
- Faça as listas de exercícios
- Use os horários de atendimento (procure ajuda)
- Respeite os horário
- Planeje seus estudos
  - Revise o material da aula teórica no dia seguinte
  - Fazer a lista + tempo para tirar dúvida
  - Tenha uma agenda!

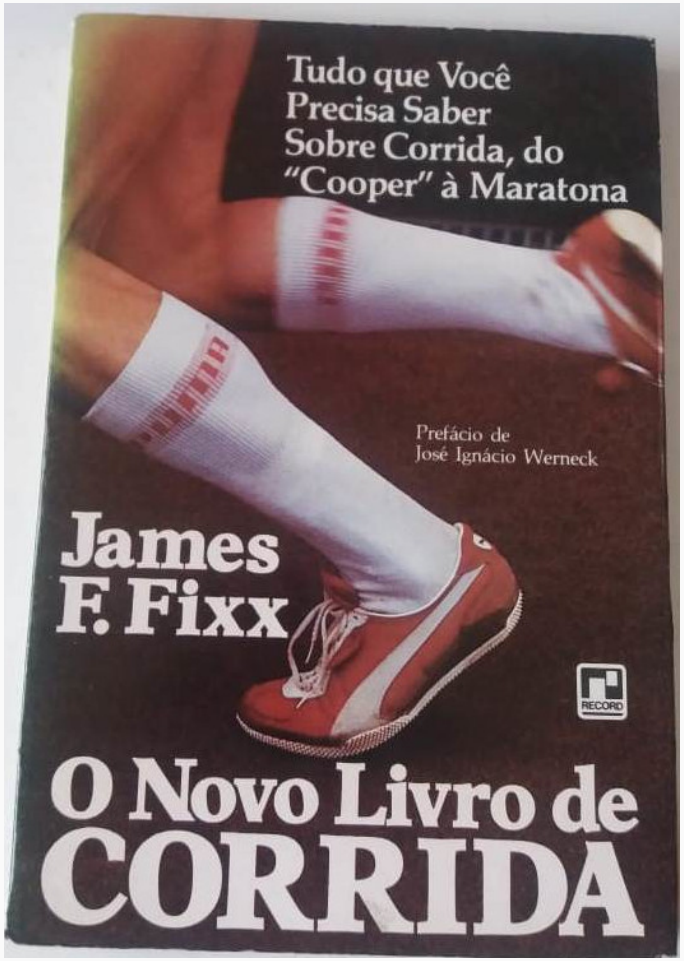
# Aprendendo a programar

- Programar é uma arte

# Aprendendo a programar

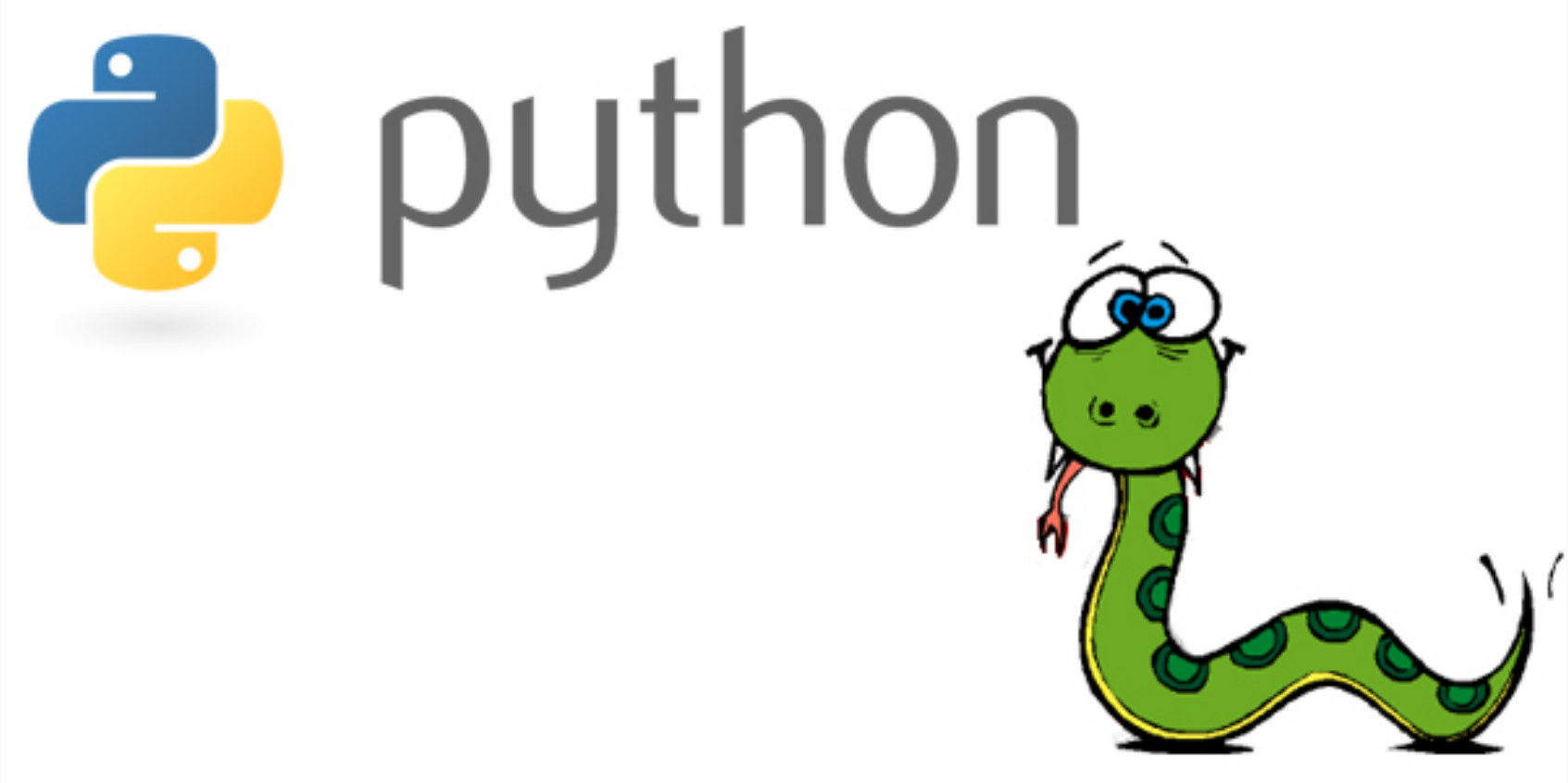
- Programar é uma arte
- Única forma de aprender: programando

# Maratona e Programação




# Linguagem do curso: Python

# Sobre Python




Python é uma linguagem de programação de alto nível e multipropósito (web, GUI, CLI, mobile, etc..)

# Sobre Python

- Código aberto (open source)

# Sobre Python

- Código aberto (open source)
- Inventada nos Países Baixo no começo dos anos 90 por Guido van Rossum

# Sobre Python

- Código aberto (open source)
- Inventada nos Países Baixo no começo dos anos 90 por Guido van Rossum
- Benevolent Dictator for Life (BDFL)

# Sobre Python

- Código aberto (open source)
- Inventada nos Países Baixo no começo dos anos 90 por Guido van Rossum
- Benevolent Dictator for Life (BDFL)
- Nomeada em homenagem ao programa de TV Monty Python's Flying Circus

# Sobre Python

- Código aberto (open source)
- Inventada nos Países Baixo no começo dos anos 90 por Guido van Rossum
- Benevolent Dictator for Life (BDFL)
- Nomeada em homenagem ao programa de TV Monty Python's Flying Circus
- É interpretada

# Sobre Python

- Código aberto (open source)
- Inventada nos Países Baixo no começo dos anos 90 por Guido van Rossum
- Benevolent Dictator for Life (BDFL)
- Nomeada em homenagem ao programa de TV Monty Python's Flying Circus
- É interpretada
- Versão atual: 3.8.1

# Sobre Python (Cont.)

*"Python is an experiment in how much freedom programmers need. Too much freedom and nobody can read another's code; too little and expressiveness is endangered."*

— Guido van Rossum

# Zen of Python

1. Beautiful is better than ugly.

# Zen of Python

1. Beautiful is better than ugly.
2. Explicit is better than implicit.

# Zen of Python

1. Beautiful is better than ugly.
2. Explicit is better than implicit.
3. Simple is better than complex.

# Zen of Python

1. Beautiful is better than ugly.
2. Explicit is better than implicit.
3. Simple is better than complex.
4. Complex is better than complicated.

# Zen of Python

1. Beautiful is better than ugly.
2. Explicit is better than implicit.
3. Simple is better than complex.
4. Complex is better than complicated.
5. Flat is better than nested.

# Zen of Python

1. Beautiful is better than ugly.
2. Explicit is better than implicit.
3. Simple is better than complex.
4. Complex is better than complicated.
5. Flat is better than nested.
6. Sparse is better than dense.

# Zen of Python

1. Beautiful is better than ugly.
2. Explicit is better than implicit.
3. Simple is better than complex.
4. Complex is better than complicated.
5. Flat is better than nested.
6. Sparse is better than dense.
7. Readability counts.

# Zen of Python

1. Beautiful is better than ugly.
2. Explicit is better than implicit.
3. Simple is better than complex.
4. Complex is better than complicated.
5. Flat is better than nested.
6. Sparse is better than dense.
7. Readability counts.
8. Special cases aren't special enough to break the rules.

# Zen of Python

1. Beautiful is better than ugly.
2. Explicit is better than implicit.
3. Simple is better than complex.
4. Complex is better than complicated.
5. Flat is better than nested.
6. Sparse is better than dense.
7. Readability counts.
8. Special cases aren't special enough to break the rules.

...

# Por que Python?

- Fácil de aprender
- Fácil de usar
- Versátil (vem com baterias!)

# Linguagens Mais Utilizadas

| Rank | Language | Type | Score |
|---|---|---|---|
| 1 | Python | web, desktop, server | 100.0 |
| 2 | Java | web, mobile, desktop | 96.3 |
| 3 | C | mobile, desktop, server | 94.4 |
| 4 | C++ | mobile, desktop, server | 87.5 |
| 5 | R | desktop | 81.5 |
| 6 | JavaScript | web | 79.4 |
| 7 | C# | web, mobile, desktop, server | 74.5 |
| 8 | Matlab | desktop | 70.6 |
| 9 | Swift | mobile, desktop | 69.1 |
| 10 | Go | web, desktop | 68.0 |

Fonte: https://spectrum.ieee.org/computing/software/the-top-programming-languages-2019

# Quem usa Python?

- Wikipedia
- Google
- Yahoo!
- CERN
- NASA
- Facebook
- Amazon
- Instagram
- Spotify

# Meu Primeiro Programa

# Meu Primeiro Programa

**Código: ola.py**

```
print("Olá, Mundo")
```

**executando**

```
python ola.py
```

# Meu Primeiro Programa

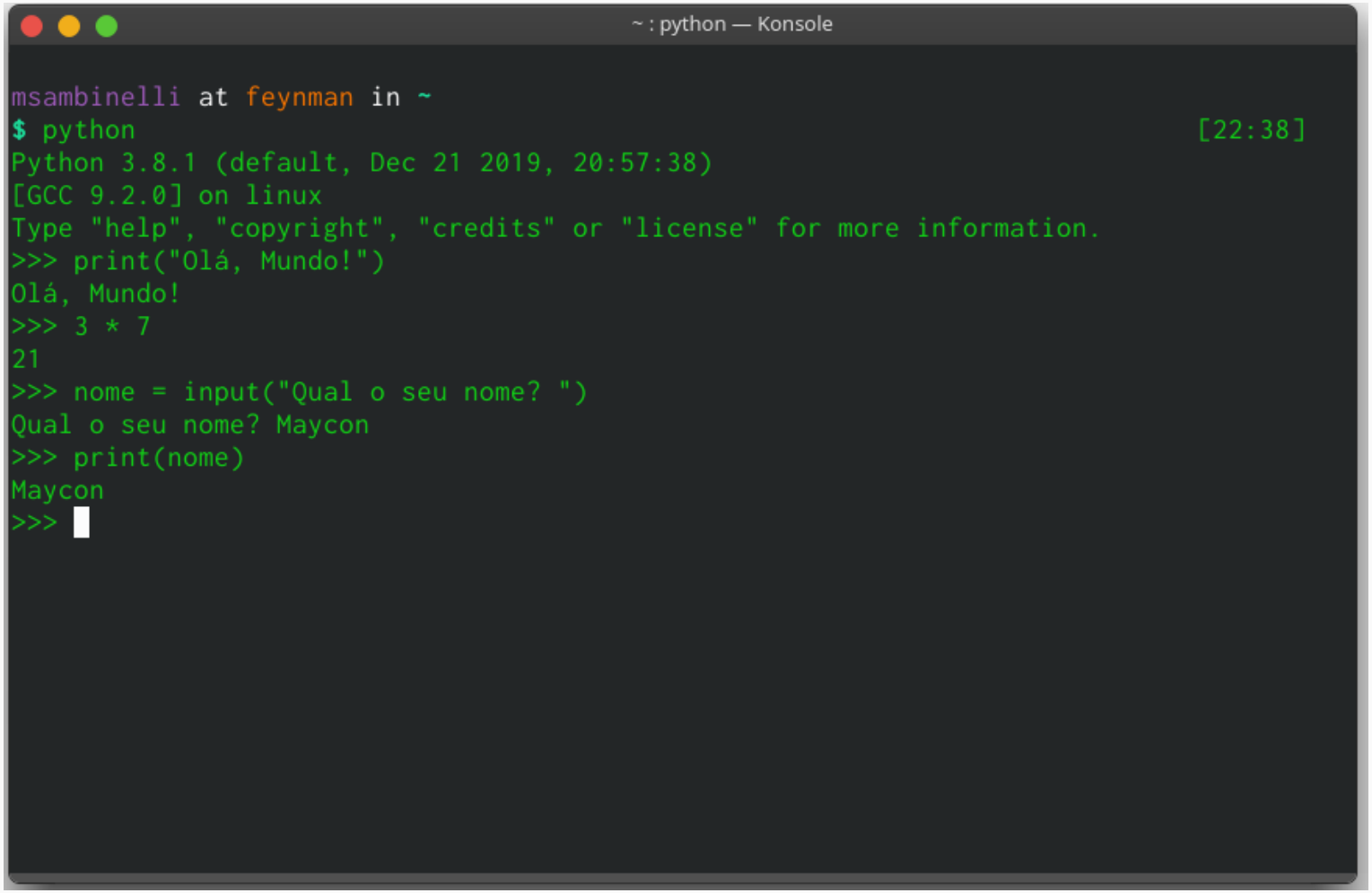
**Código: ola_pessoa.py**

```
nome = input(''Qual o seu nome? '')
print("Olá, ", nome)
```

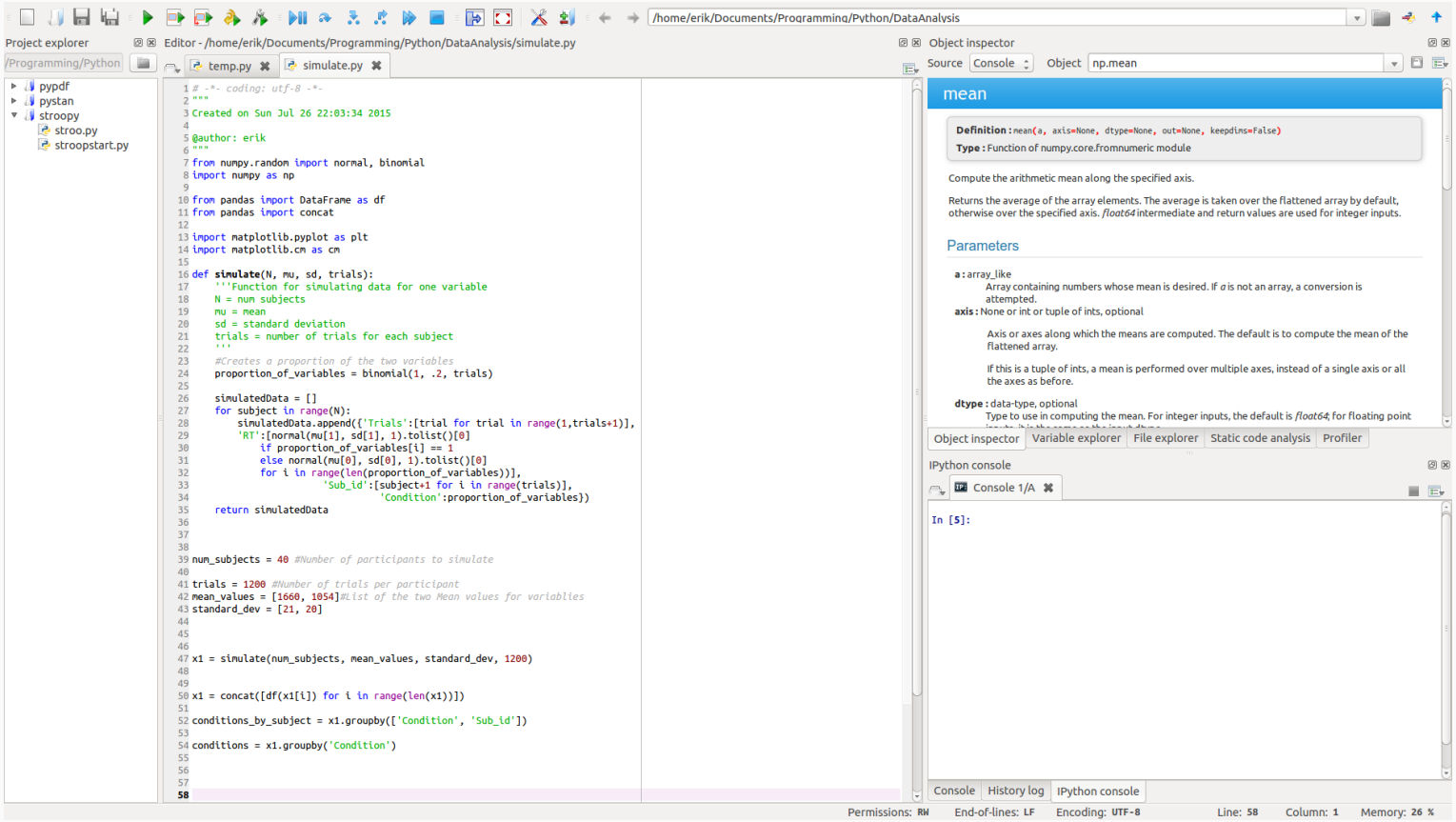
**executando**

```
python ola_pessoa.py
```

# Read–Eval–Print Loop (REPL)



# Spyder


/home/erik/Documents/Programming/Python/DataAnalysis
Project explorer
/Programming/Python
pypdf
pystan
stroopy
stroo.py
stroopstart.py
Editor - /home/erik/Documents/Programming/Python/DataAnalysis/simulate.py
temp.py
simulate.py
# -*- coding: utf-8 -*-
"""
Created on Sun Jul 26 22:03:34 2015
@author: erik
"""
from numpy.random import normal, binomial
import numpy as np
from pandas import DataFrame as df
from pandas import concat
import matplotlib.pyplot as plt
import matplotlib.cm as cm
def simulate(N, mu, sd, trials):
'''Function for simulating data for one variable
N = num subjects
mu = mean
sd = standard deviation
trials = number of trials for each subject
'''
#Creates a proportion of the two variables
proportion_of_variables = binomial(1, .2, trials)
simulatedData = []
for subject in range(N):
simulatedData.append({'Trials':[trial for trial in range(1,trials+1)],
'RT':[normal(mu[1], sd[1], 1).tolist()[0]
if proportion_of_variables[i] == 1
else normal(mu[0], sd[0], 1).tolist()[0]
for i in range(len(proportion_of_variables))],
'Sub_id':[subject+1 for i in range(trials)],
'Condition':proportion_of_variables})
return simulatedData
num_subjects = 40 #Number of participants to simulate
trials = 1200 #Number of trials per participant
mean_values = [1660, 1054]#List of the two Mean values for variablies
standard_dev = [21, 20]
x1 = simulate(num_subjects, mean_values, standard_dev, 1200)
x1 = concat([df(x1[i]) for i in range(len(x1))])
conditions_by_subject = x1.groupby(['Condition', 'Sub_id'])
conditions = x1.groupby('Condition')
Object inspector
Source
Console
Object
np.mean
mean
Definition : mean(a, axis=None, dtype=None, out=None, keepdims=False)
Type : Function of numpy.core.fromnumeric module
Compute the arithmetic mean along the specified axis.
Returns the average of the array elements. The average is taken over the flattened array by default, otherwise over the specified axis. float64 intermediate and return values are used for integer inputs.
Parameters
a : array_like
Array containing numbers whose mean is desired. If a is not an array, a conversion is attempted.
axis : None or int or tuple of ints, optional
Axis or axes along which the means are computed. The default is to compute the mean of the flattened array.
If this is a tuple of ints, a mean is performed over multiple axes, instead of a single axis or all the axes as before.
dtype : data-type, optional
Type to use in computing the mean. For integer inputs, the default is float64; for floating point
Object inspector
Variable explorer
File explorer
Static code analysis
Profiler
IPython console
Console 1/A
In [5]:
Console
History log
IPython console
Permissions: RW
End-of-lines: LF
Encoding: UTF-8
Line: 58
Column: 1
Memory: 26 %

# Operações aritméticas

**Código**

```
# Operações básicas
1 + 1   # => 2
8 - 1   # => 7
10 * 2  # => 20
35 / 5  # => 7.0

# Divisão inteira arredonda para baixo
5 // 3       # => 1
-5 // 3      # => -2
5.0 // 3.0   # => 1.0 # works on floats too

# Resto da divisão
7 % 3  # => 1

# Exponenciação  (x**y, x a y-ésima potência)
2**3  # => 8

# Parênteses forçam precedência
1 + 3 * 2    # => 7
(1 + 3) * 2  # => 8
```


That's all Folks!