

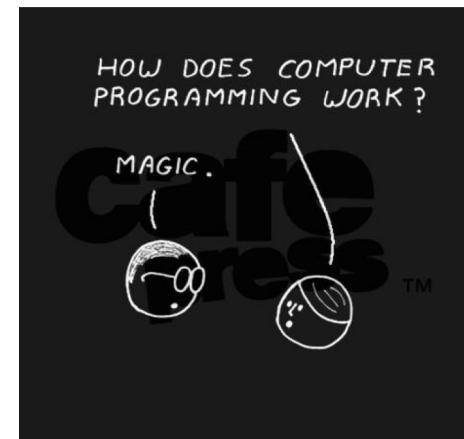
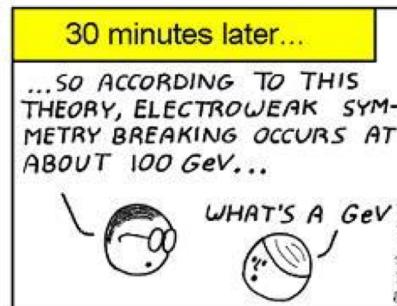
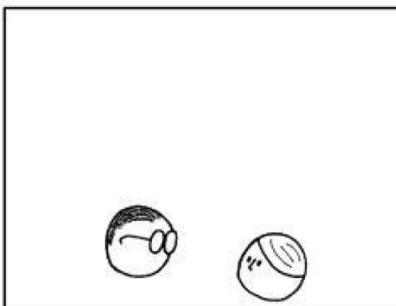
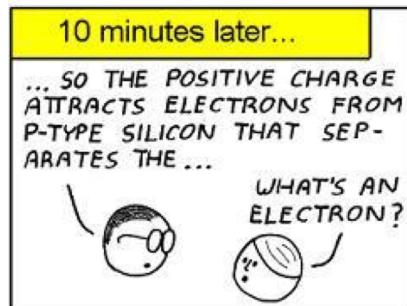
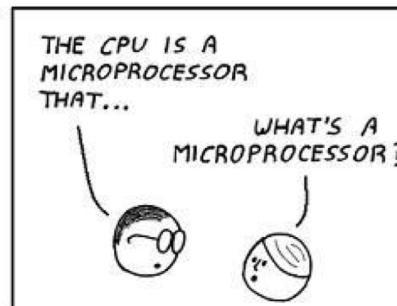
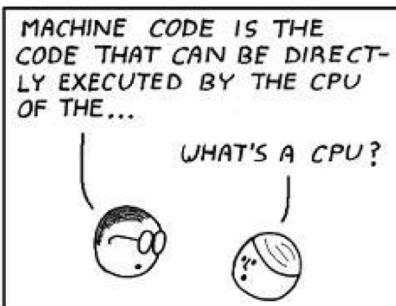
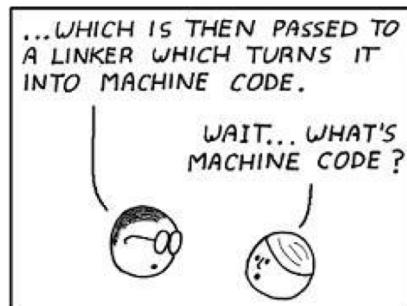
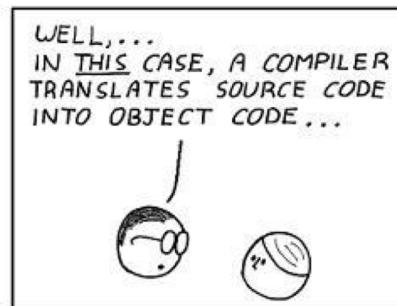
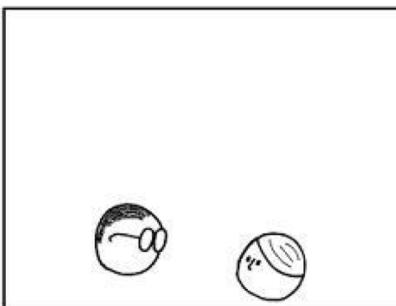
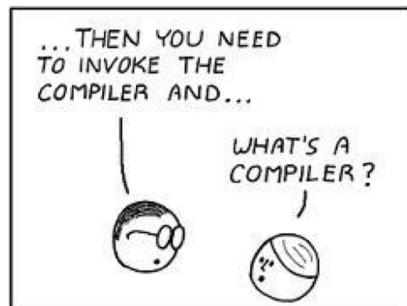
Apresentação da Disciplina de Programação de Computadores I

Vanessa Braganholo
vanessa@ic.uff.br

Apresentações

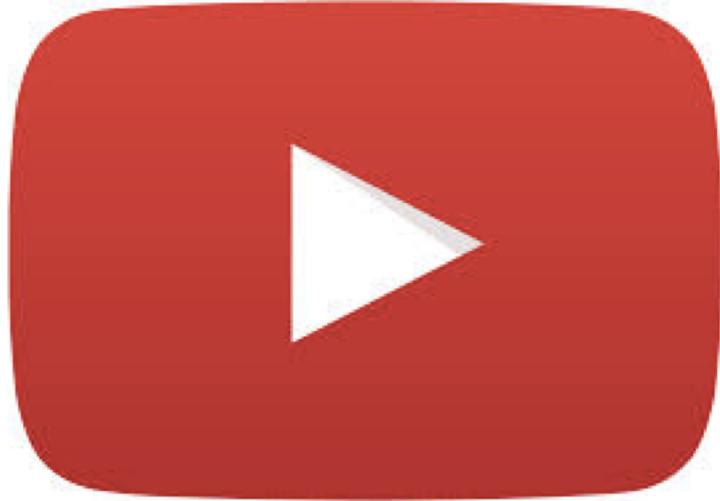
- ▶ Quem sou eu?
 - ▶ Vanessa Braganholo
 - ▶ <http://www.ic.uff.br/~vanessa>
- ▶ Quem são vocês?
 - ▶ Nome?
 - ▶ Onde estudou? O que sabe de computação?
 - ▶ Algum hobby “sério”?

O quê vocês esperam do curso?



My code is
guaranteed
100% mistake
free.

O que você quer ser quando crescer: PROGRAMADOR



Aulas

- ▶ Quartas e Sextas
 - ▶ Sala 302
- ▶ Laboratório
 - ▶ 307
- ▶ Monitoria
 - ▶ Jackson e João Victor



Home

Publications

Courses

2018.2

ED

PROG I

2018.1

2017.2

2017.1

2016.2

2016.1

2015.2

2015.1

2014.2

2014.1

2013.1

2012.2

Programação de Computadores I

Horário: quartas e sextas de 11h às 13h

Local: a definir

Sala de aula virtual da disciplina: usaremos o **Google Classroom** para entrega de exercícios da disciplina. É necessário ter uma conta no ID UFF. A sala de aula no Google Classroom será usada também para divulgar avisos gerais e para dúvidas.

Importante: todos os alunos **devem** se inscrever no Google Classroom (os alunos para os quais eu possuía o endereço de email do ID UFF receberam convite – nesse caso basta aceitar o convite). Para se inscrever, clique no símbolo de "+" no canto superior direito da página, e selecione a opção "Participar da Turma". O código de inscrição na turma está na lista de presença.

Monitoria

Jackson - TER 07:00-09:00; QUI 07:00-09:00; S

João Victor - SEG 11:00-13:00; TER 14:00-18:00

Ementa

Importante:
cadastrem-se na turma
do Google Classroom

Leiam as **regras** do curso no site,
anotem as **datas** e tragam as
dúvidas na próxima aula

Objetivos da Disciplina

- ▶ Solucionar problemas (x 1000)
- ▶ Desenvolver pensamento computacional
- ▶ Escrever e ler na linguagem do computador
- ▶ Objetivo secundário: Programar em Python
 - ▶ Atualmente a mais popular linguagem introdutória de cursos de programação nas universidades top dos EUA
 - ▶ Criada por Guido van Rossum, por volta de 1991
 - ▶ Fácil partir para outras linguagens, se necessário



Quem usa Python?



Avaliação

- ▶ $P_1 = \text{Prova sem consulta}$
- ▶ $P_2 = \text{Prova sem consulta}$
- ▶ $\text{Média} = (P_1 + P_2)/2$

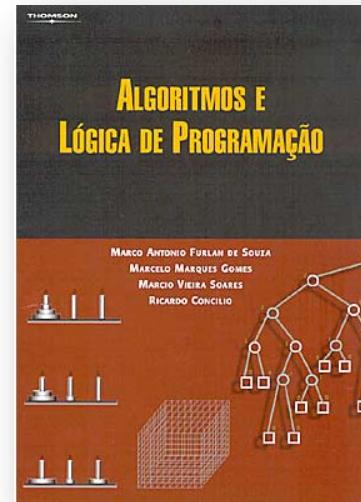
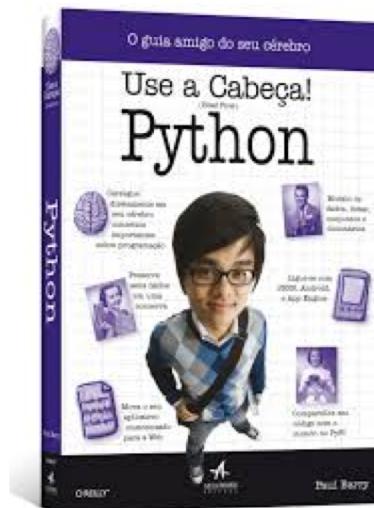
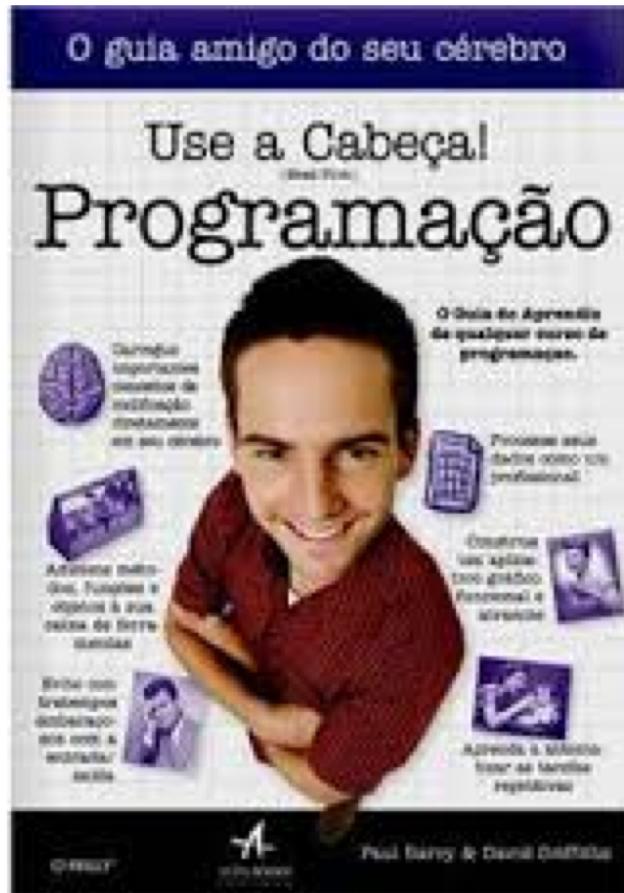
Avaliação

- ▶ **APROVADO:** (Presença $\geq 75\%$) E (Média ≥ 6)
- ▶ **VS:** (Presença $\geq 75\%$) E ($4 \leq$ Média < 6)
 - ▶ Será aprovado na VS se tirar nota maior ou igual a 6
- ▶ **REPROVADO:** (Presença $< 75\%$) OU (Média < 4)

Exercícios

- ▶ Serão apresentados exercícios em todas as aulas
- ▶ Alguns serão feitos em sala, os demais devem ser feitos em casa
- ▶ Todos terão data para serem entregues via Google Classroom
- ▶ Dúvidas devem ser tiradas com os monitores ou postadas no Google Classroom

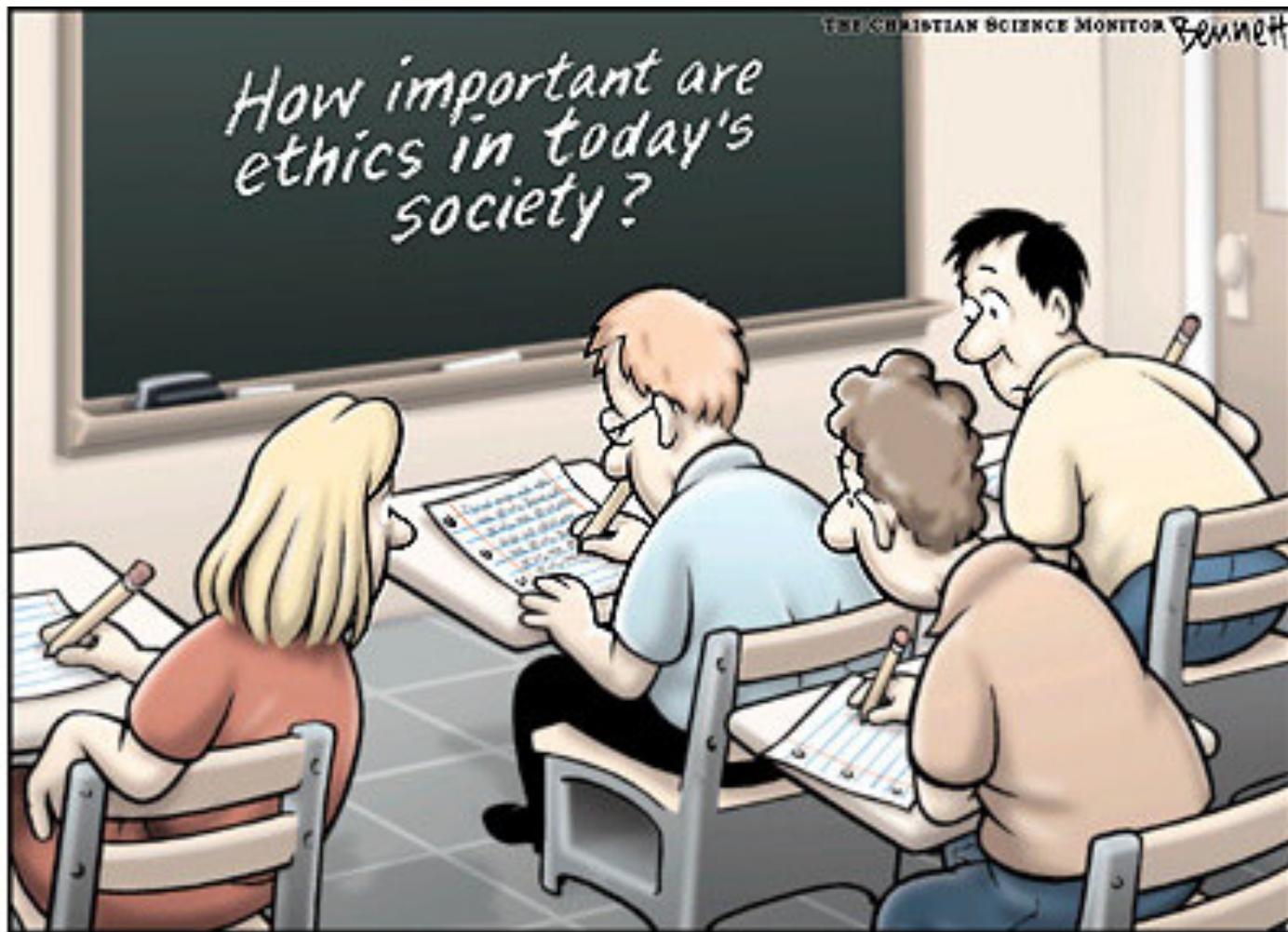
Bibliografia do curso



Dois conselhos

1. Aprender a programar é como aprender a tocar um instrumento musical: não basta ler, **tem que praticar**
2. Escreva seus programas de forma que seja fácil para outras pessoas os entenderem

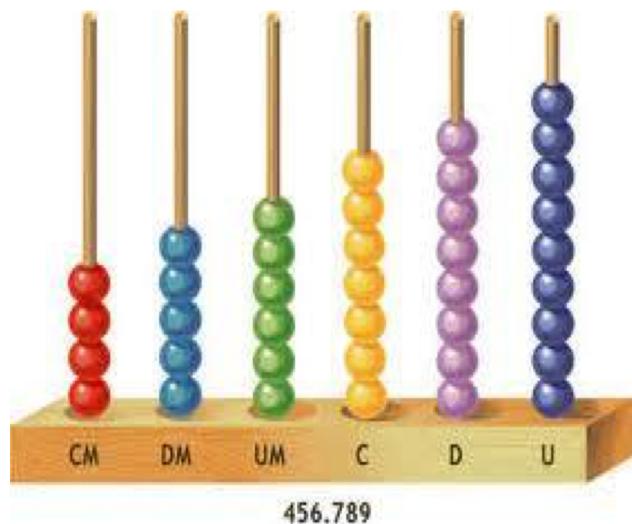
Fair Play!



Um pouco de história

► 2400 AC: Ábaco

- ▶ Primeira tentativa de se criar um artefato capaz de contar



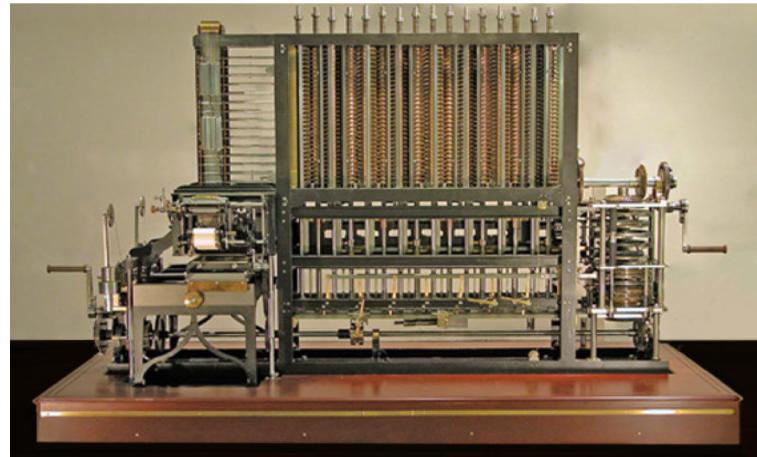
► 1642: Pascalina

- ▶ Criada por Blaise Pascal, aos 19 anos
- ▶ Uma das primeiras máquinas mecânicas de calcular



Um pouco de história

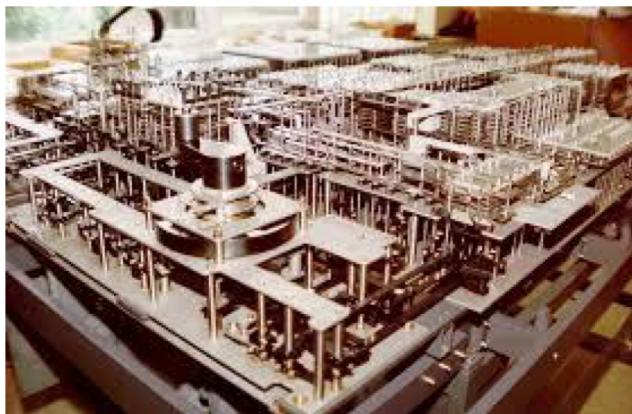
- ▶ 1822: Primeiro computador mecânico
 - ▶ Projetado por Charles Babbage mas não terminado devido à falta de recursos
 - ▶ Posteriormente, o seu projeto foi construído e exposto em um museu



Um pouco de história

▶ 1938: Z1

- ▶ O primeiro computador programável
- ▶ Muito foi perdido por causa da II Guerra Mundial



▶ 1943: Colossus

- ▶ Usado pelos britânicos para decodificar mensagens alemãs (assistam ao filme “O Jogo da Imitação”)



Um pouco de história

▶ 1946: ENIAC

- ▶ Considerado o primeiro computador eletrônico de propósito geral
- ▶ Construído na Universidade da Pensilvânia
- ▶ Ocupava uma sala inteira
- ▶ Pesava 30 toneladas
- ▶ Consumia 200 kw de potência
- ▶ Entrada: leitora de cartões perfurados
- ▶ Saída: perfuradora de cartões



ENIAC
<http://www.upenn.edu>

Um pouco de história

▶ 1973: Alto

- ▶ Primeiro computador pessoal
- ▶ Construído pela Xerox, mas nunca produzido em massa
- ▶ Já tinha mouse, interface gráfica e sistema operacional
- ▶ As características do Alto foram incorporadas no Macintosh



Um pouco de história

- ▶ Hoje: Computação móvel
- ▶ Laptop, Tablet, Celular
- ▶ Entrada: teclado, tela
- ▶ Saída: tela



Hardware x Software

Hardware

- ▶ Peças (Corpo)



Software

- ▶ Programas (Alma)



Computadores e programas

- ▶ Computador: máquina que pode executar programas
- ▶ Programa: sequência bem precisa de passos que um computador deve executar
- ▶ Linguagem de programação: linguagem projetada para produzir programas de computadores



O que os computadores entendem?

- ▶ Para que o computador faça o que você quer (e ele pode fazer (quase) tudo que você mandar), é necessário falar a linguagem dele
- ▶ Qual é a linguagem que o computador fala?



Bits and pieces

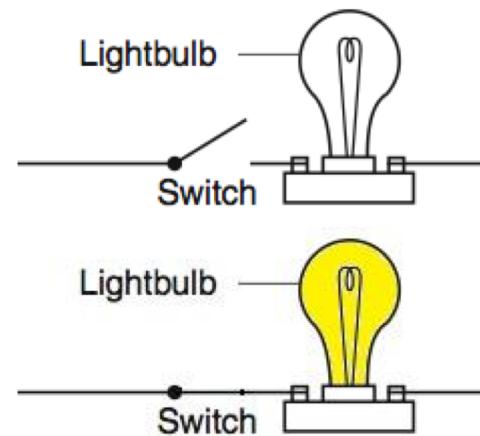
“Your computer successfully creates the illusion that it contains photographs, letters, songs, and movies. All it really contains is bits, lots of them, patterned in ways you can't see. **Your computer was designed to store just bits - all the files and folders and different kinds of data are illusions created by computer programmers.**”

(Hal Abelson, Ken Ledeen, Harry Lewis, in "Blown to Bits")



Instruções no computador

- ▶ Operações no computador são grupos de bits
 - ▶ 0 ou 1
 - ▶ ligado ou desligado
- ▶ Microprocessador
 - ▶ Move o conteúdo de grupos de bits
 - ▶ Soma pares de grupos de bits
 - ▶ Subtrai um grupo de bits de outro
 - ▶ Compara pares de grupos de bits
 - ▶ ...



Bits and pieces

“There are only 10 different kinds of people in the world:
those who know binary and those who don't.”

- Anônimo

- ▶ $1234 = 10011010010$
- ▶ “A” = 65 (decimal) = 01000001



Linguagens de montagem

- ▶ Usa símbolos mais amigáveis aos seres humanos para representar as instruções
- ▶ A memória do computador e os registradores também recebem nomes simbólicos
- ▶ Não são bits, mas ainda requerem um nível de detalhe muito próximo ao que a máquina de fato faz



Assembler – Soma de dois números (17 e 5)

```
main PROC
    mov eax, offset x
    push eax
    mov eax, 17
    push eax
    mov eax, 5
    push eax
    pop ebx
    pop eax
    add eax, ebx
    push eax
    pop eax
    pop ebx
    mov [ebx], eax
    call writeint
    call crlf
    exit
main ENDP
END main
```

Linguagens de programação de alto nível

- ▶ Bem próximas da linguagem humana
- ▶ O programador pode se concentrar **no que ele quer que o computador faça**, ao invés de ter que detalhar **como o computador executará cada instrução**
- ▶ Abstração
- ▶ Um outro programa (compilador/interpretador) se encarregará de traduzir seu programa para a linguagem de máquina
- ▶ Na nossa disciplina, usaremos um interpretador Python



Um programa em Java

```
public class HelloPrinter {  
    public static void main(String[] args) {  
        System.out.println("Hello,World!");  
    }  
}
```



Um programa em Java

```
public class HelloPrinter {  
    public static void main(String[] args) {  
        System.out.println("Hello,World!");  
    }  
}
```



O que eu quero que seja
feito



Um programa executando a mesma tarefa em Python

```
print("Hello, world!!")
```



Um programa executando a mesma tarefa em Python

```
print("Hello, world!!")
```

- ▶ Digo apenas o que eu quero que seja feito!



Um comando em Python

```
>>> print("3+4+5")
```



Um comando em Python

```
>>> print("3+4+5")
```

3+4+5



Um comando em Python

```
>>> print(3+4+5)
```



Um comando em Python

```
>>> print(3+4+5)
```

```
12
```



Erros

```
>>> printer(3+4+5)
```



Erros

```
>>> printer(3+4+5)
```

Traceback (most recent call last):

 File "<pyshell#3>", line 1, in <module>

 NameError: name 'printer' is not defined



Erro?

```
>>> print("Hello, world!!")
```



Erro?

```
>>> print(10/0)
```



Erro?

```
>>> print(10/0)
```

Traceback (most recent call last):

 File "<pyshell#5>", line 1, in <module>

ZeroDivisionError: division by zero



Créditos

- ▶ Material elaborado em conjunto com Leonardo Murta e Aline Paes

Apresentação da Disciplina de Programação de Computadores I

Vanessa Braganholo
vanessa@ic.uff.br