Técnicas de Programação II

Prof° Ramon Trigo

Conteúdo Programático

- Varáveis e Entrada de Dados
- Condições
- Repetições
- Lista
- Funções
- Arquivos
- Gui
- Integração com Banco de Dados

Formas de Avaliação

Avaliação Integradora

+

Atividades

P1

Atividades

P2

+

Projeto Interdisciplinar

Introdução

Python é uma linguagem de altíssimo nível (em inglês, Very High Level Language) orientada a objeto, de tipagem dinâmica e forte, interpretada e interativa.

Obs. **Tipagem dinâmica** é uma característica de determinadas linguagens de programação, que não exigem declarações de tipos de dados, pois são capazes de escolher que tipo utilizar dinamicamente para cada variável, podendo alterá-lo durante a compilação ou a execução do programa.

Tipagem forte costuma ser a característica que não permite um mesmo dado ser tratado como se fosse de outro tipo.

Características

O Python possui uma sintaxe clara e concisa, que favorece a legibilidade do código fonte, tornando a linguagem mais produtiva.

A linguagem inclui diversas estruturas de alto nível (listas, dicionários, data / hora, complexos e outras) e uma vasta coleção de módulos prontos para uso, além de frameworks de terceiros que podem ser adicionados.

Características

Multiparadigma, a linguagem suporta programação modular e funcional, além da orientação a objetos.

Mesmo os tipos básicos no Python são objetos.

A linguagem é interpretada através de bytecode pela máquina virtual Python, tornando o código portável.

Com isso é possível compilar aplicações em uma plataforma e rodar em outros sistemas ou executar direto do código fonte.

Características

Python é um software de código aberto (com licença compatível com a General Public License (GPL), porém menos restritiva, permitindo que o Python seja inclusive incorporado em produtos proprietários).

A especificação da linguagem é mantida pela Python Software Foundation2 (PSF).

Características

Além de ser utilizado como linguagem principal no desenvolvimento de sistemas, o Python também é muito utilizado como linguagem script em vários softwares, permitindo automatizar tarefas e adicionar novas funcionalidades, entre eles: BrOffice.org, PostgreSQL, Blender, GIMP e Inkscape.

Características

É possível integrar o Python a outras linguagens, como a Linguagem C e Fortran. Em termos gerais, a linguagem apresenta muitas similaridades com outras linguagens dinâmicas, como Perl e Ruby

Histórico

A linguagem foi criada em 1990 por Guido van Rossum, no Instituto Nacional de Pesquisa para Matemática e Ciência da Computação da Holanda (CWI) e tinha originalmente foco em usuários como físicos e engenheiros. O Python foi concebido a partir de outra linguagem existente na época, chamada ABC. Hoje, a linguagem é bem aceita na indústria por empresas de alta tecnologia,

Versões

A implementação oficial do Python é mantida pela PSF e escrita em C, e por isso, é também conhecida como CPython. A versão estável mais recente está disponível para download no endereço: http://www.python.org/download/ Para a plataforma Windows, basta executar o instalador. Para outras plataformas, como em sistemas Linux, geralmente o Python já faz parte do sistema, porém em alguns casos pode ser necessário compilar e instalar o interpretador a partir dos arquivos fonte. Existem também implementações de Python para .NET (IronPython), JVM (Jython) e em Python (PyPy).

Tipagem Dinâmica

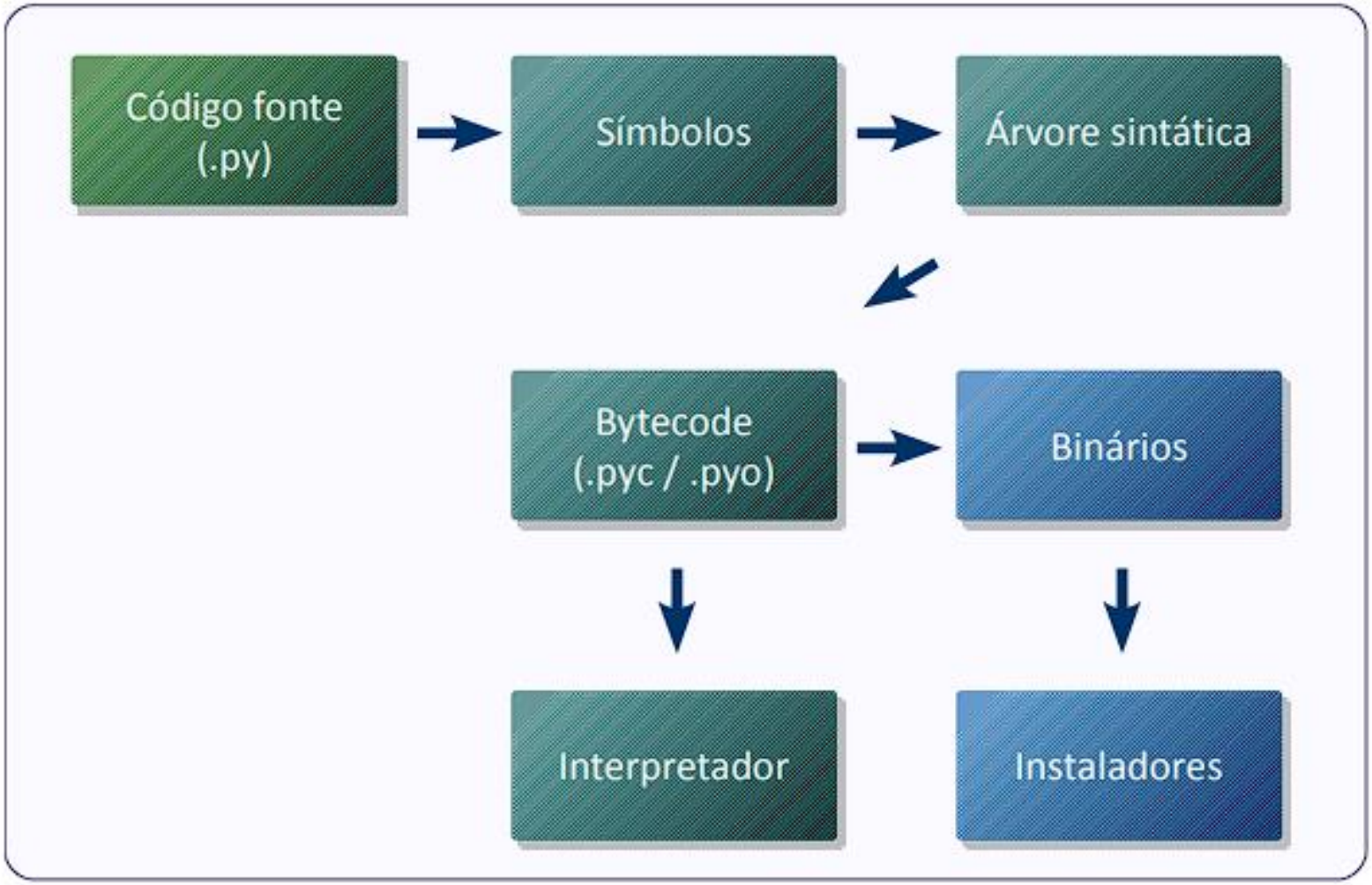
Python utiliza tipagem dinâmica, o que significa que o tipo de uma variável é inferido pelo interpretador em tempo de execução (isto é conhecido como Duck Typing). No momento em que uma variável é criada através de atribuição, o interpretador define um tipo para a variável, com as operações que podem ser aplicadas.

A tipagem do Python é forte, ou seja, o interpretador verifica se as operações são válidas e não faz coerções automáticas entre tipos incompatíveis. Para realizar a operação entre tipos não compatíveis, é necessário converter explicitamente o tipo da variável ou variáveis antes da operação

Compilação e Interpretação

O código fonte é traduzido pelo Python para bytecode, que é um formato binário com instruções para o interpretador. O bytecode é multiplataforma e pode ser distribuído e executado sem fonte original.

Compilação e Interpretação



Compilação e Interpretação

Por padrão, o interpretador compila o código e armazena o bytecode em disco, para que a próxima vez que o executar, não precise compilar novamente o programa, reduzindo o tempo de carga na execução.

Se os arquivos fontes forem alterados, o interpretador se encarregará de regerar o bytecode automaticamente, mesmo utilizando o shell interativo.

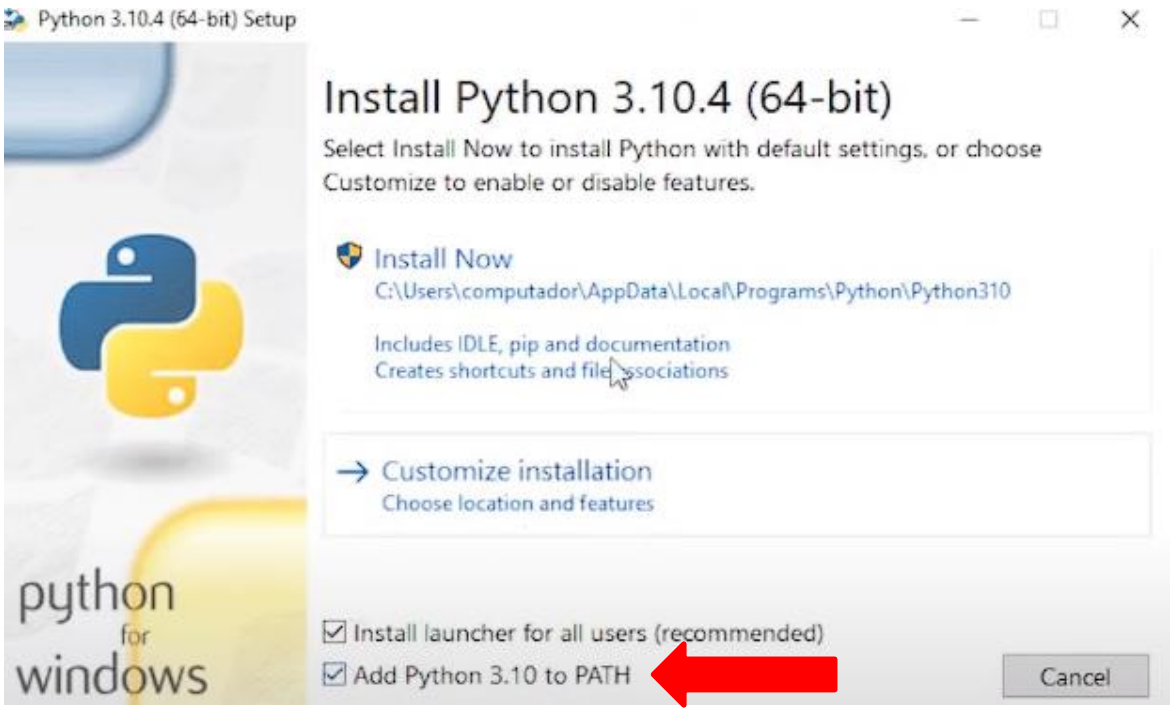
Quando um programa ou um módulo é evocado, o interpretador realiza a análise do código, converte para símbolos, compila (se não houver bytecode atualizado em disco)

Compilação e Interpretação

O bytecode é armazenado em arquivos com extensão “.pyc” (bytecode normal) ou “.pyo” (bytecode otimizado). O bytecode também pode ser empacotado junto com o interpretador em um executável, para facilitar a distribuição da aplicação, eliminando a necessidade de instalar Python em cada computador.

Modo Interativo

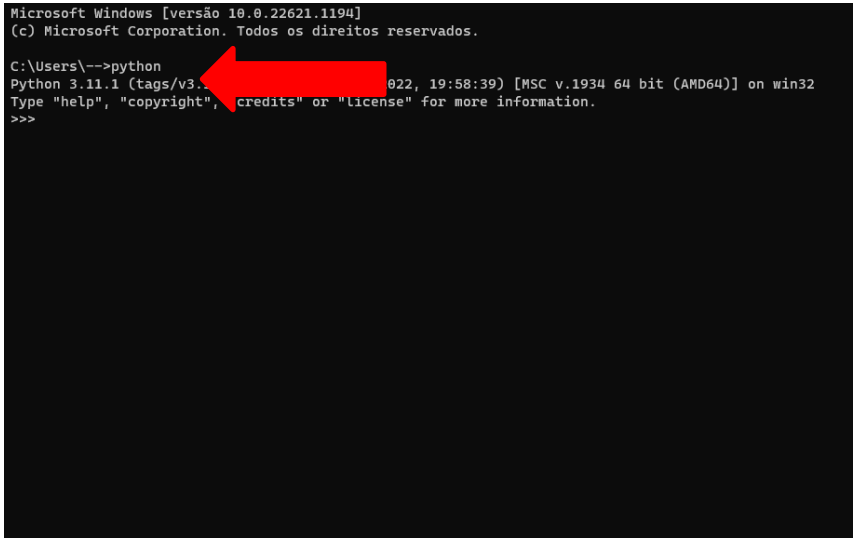
O interpretador Python pode ser usado de forma interativa, na qual as linhas de código são digitadas em um prompt (linha de comando) semelhante ao shell do sistema operacional. Para evocar o modo interativo basta executar o interpretador (se ele estiver no path):



Modo Interativo

Para evocar o modo interativo basta executar o interpretador

No cmd do Windows digite o comando python



O cmd utilizado para rodar comandos desenvolvido em python

https://youtu.be/kQgqjTC4OEY

Tipo de Dados

As Linguagens de programação, em geral, usam o termo “tipo primitivo” para representar a informação em sua forma mais elementar, tais como inteiro, real, lógico ou caractere.

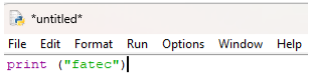
Na documentação oficial da linguagem Python11, o termo “tipo primitivo” não é utilizado, mas sim “tipos built-ins” (ou tipos construídos). O motivo é que, para Python, tudo é um objeto.

Diferentemente de outras linguagens de programação como C, C++ e Java, Python possui tipagem dinâmica. Uma linguagem de programação que possui tipagem dinâmica como Python, PHP ou Perl não exige que o programador declare, explicitamente, o tipo de dado que será armazenado por cada variável. Essa característica permite que, ao longo da execução de um programa, uma mesma variável armazene valores de tipos distintos.

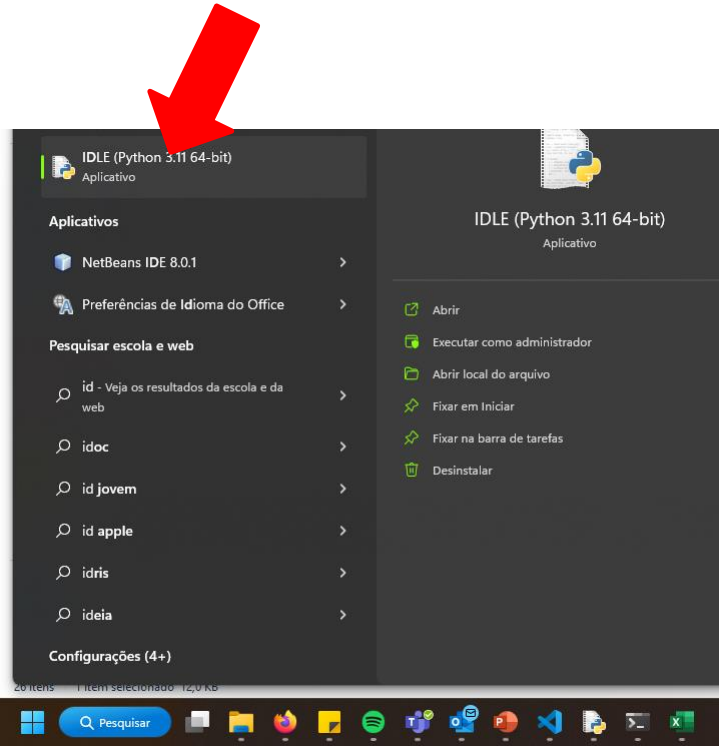
Exemplo

```
C:\Users\-->python
Python 3.11.1 (tags/v3.11.1:a7a450f, Dec  6 2022, 19:58:39) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> exemplo = "FATEC"
>>> exemplo = 2023
>>> exemplo = True
>>> exemplo = 1.99
>>>
```

Primeiro Projeto



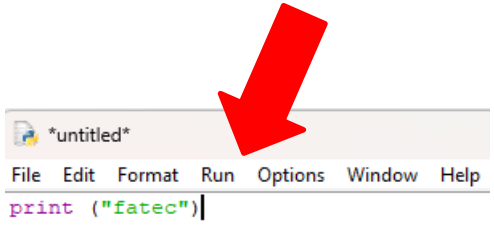
Arquivo de texto.



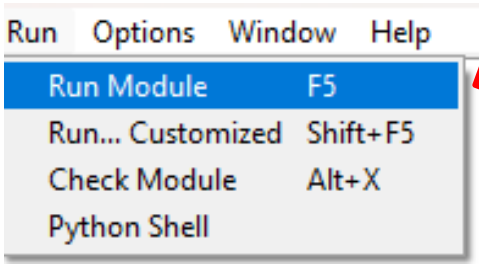
Obs. Será salvo com a extensão .py

Execução do Projeto

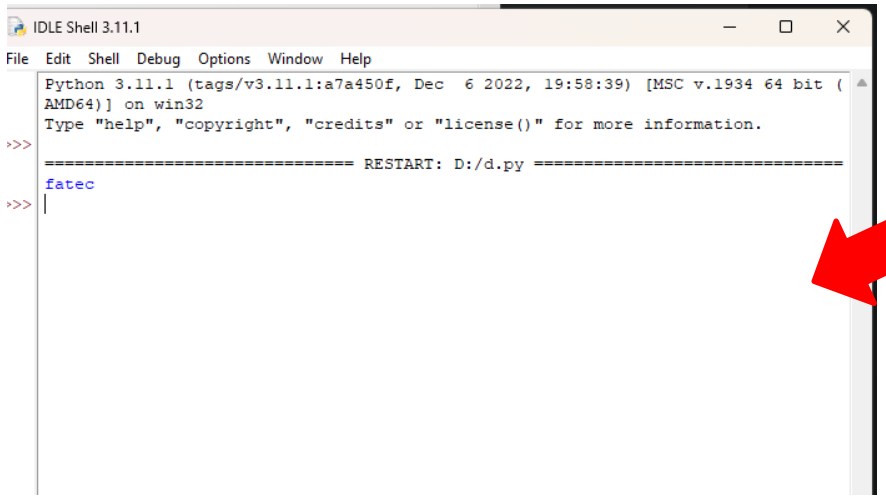
Opção – Run



Run Module



Execução do Arquivo



Variáveis e Constantes

Em Python, podemos declarar uma variável simplesmente definindo um nome para ela e atribuindo-lhe um valor. Atribuir valores a variáveis nada mais é do que armazenar um valor (ou um conjunto deles) em um determinado endereço de memória referenciando-o por meio de um nome.

```
professor = "Ramon Alves Trigo"
periodo_corrente = 3
valor = 9.10
status = True
```

Variáveis e Constantes

Além disso, deve-se observar também que as palavras reservadas da liguagem Python não podem ser usadas como nomes de variáveis. Portanto, não é permitido declarar variáveis com os seguintes nomes.

and	as	assert	break	class	continue	def
del	elif	else	except	False	finally	for
from	global	if	import	in	is	lambda
None	nonlocal	not	or	pass	raise	return
True	try	while	with	yield		

Operadores Aritméticos

Python oferece diversos conjuntos de operadores que podem ser utilizados em um programa. Para realizar operações matemáticas, podemos utilizar os operadores aritméticos.

OPERADOR	DESCRIÇÃO	EXEMPLO DE APLICAÇÃO
+	Adição	`print(4 + 2)` `#resulta em 6`
-	Subtração	`print(4 - 2)` `#resulta em 2`
*	Multiplicação	`print(4 * 2)` `#resulta em 8`
/	Divisão	`print(4 / 3)` `#resulta em 1.3333`
//	Quociente inteiro da divisão	`print(4 // 3)` `#resulta em 1`

Operadores Aritméticos

Python oferece diversos conjuntos de operadores que podem ser utilizados em um programa. Para realizar operações matemáticas, podemos utilizar os operadores aritméticos.

OPERADOR	DESCRIÇÃO	EXEMPLO DE APLICAÇÃO
%	Resto da divisão inteira	`print(4 % 2)` `#resulta em 0`
**	Potenciação	`print(4 ** 2)` `#resulta em 16`

Operadores de Atribuição

os operadores aritméticos, agora apresentaremos os operadores de atribuição, desde aquele mais simples e direto – como o sinal de igual (=) que, nós, professores, gostamos de chamar de “recebe” – até as abreviações envolvendo operadores aritméticos e de atribuição.

OPERADOR	DESCRIÇÃO	EXEMPLO DE APLICAÇÃO
=	Atribuição simples	`x = 2` `#x recebe 2`
+=	Atribuição de adição	`x += 2` `#equivale a x = x + 2`
-=	Atribuição de subtração	`x -= 2` `#equivale a x = x - 2`
*=	Atribuição de multiplicação	`x *= 2` `#equivale a x = x * 2`
/=	Atribuição de divisão	`x/= 2` `#equivale a x = x / 2`
%=	Atribuição de resto inteiro da divisão	`x%= 2` `#equivale a x = x % 2`
**=	Atribuição de potência	`x**= 2` `#equivale a x = x ** 2`

Tipos de Dados

int – armazena valores numéricos inteiros

float – armazena valores numéricos com ponto flutuante

complex – armazena valores numéricos complexos

bool – armazena valores lógicos (True ou False). O valor True pode ser representado por 1 e o False por 0 e, por isso, alguns autores consideram valores do tipo bool como sendo do tipo inteiro.

str – armazena cadeias de caracteres

list – armazena conjuntos de elementos que podem ser acessados por meio de um índice

dic – armazena um conjunto de elementos que podem ser acessados por meio de uma chave

ENTRADA E DADOS E CONVERSÃO DE TIPOS

No entanto, em um cenário real, geralmente o usuário interage com o programa informando dados de entrada. Em Python, utiliza-se a função input()

```
aluno = input ("Digite seu nome: ")
periodo_semestre = input ("Digite o semestre atual: ")
print (aluno)
print(periodo_semestre)
```

CONVERSÃO DE VALORES

```
base = float(input("Digite a base "))
altura = float(input ("Digite a altura"))
area = base * altura
print (area)
```

FORMATAÇÃO DE STRING

```
from datetime import datetime
ano_atual = datetime.now().year
faculdade = "Fatec"
turma = 3
ano_fundacao = 2022
print(f"{faculdade} possui {turma} turmas do curso de DSM.")
```

Através da letra f é possível realizar concatenação entre a frase e a variável

Exercício

1. Construa um programa no qual um usuário informe a sua estatura em metros e o programa converta-a para centímetros.

2. Construa um programa que receba do usuário a variação do deslocamento de um objeto (em metros) e a variação do tempo percorrido (em segundo). Ao fim, o programa deve calcular a velocidade média, em m/s, do objeto.

3. Construa um programa para calcular a área de convivência de uma escola cujo formato é circular. Para isso, o usuário deve informar o valor do raio.

Exercício

1. Construa um programa no qual um usuário informe a sua estatura em metros e o programa converta-a para centímetros.

```
estatura = float(input("Digite a sua estatura (em metros): "))
estatura = estatura * 100
print(f"Sua estatura é de {estatura} cm.")
```

Exercício

2. Construa um programa que receba do usuário a variação do deslocamento de um objeto (em metros) e a variação do tempo percorrido (em segundo). Ao fim, o programa deve calcular a velocidade média, em m/s, do objeto.

```
delta_s = float(input("Digite o deslocamento (em metros): "))
delta_t = float(input("Digite o tempo (em segundos): "))
velocidade = delta_s / delta_t
print(f"Vm = {velocidade:.2f} m/s")
```

Exercício

3. Construa um programa para calcular a área de convivência de uma escola cujo formato é circular. Para isso, o usuário deve informar o valor do raio

```
from math import pi
raio = float(input("Digite o raio da área: "))
area = pi * raio ** 2 # mesmo que area = pi * pow(raio, 2)
print(f"Área = {area:.2f}.")
```

Exercício

4. Um aluno iniciou seus estudos em geometria plana e, para validar se suas respostas estão corretas, solicitou sua ajuda. Sabendo que área = (base*altura)/2, construa um programa para auxiliar esse aluno

```
base = float(input("Base do triângulo (cm): "))
alt  = float(input("Altura do triângulo (cm): "))
area = (base * alt) / 2
print(f"Área = {area:.2f} cm²")
```