Aula 2: Variáveis, Tipos de Dados, Escrita e Leitura

Luís Felipe

UFF

20 de Agosto de 2025

Luís Felipe

20/08/25

Shell Interativa

- Abra um prompt de comando e execute: python3.
- Se Python estiver instalado em seu computador será inicializado a shell de Python.

Execução no terminal:

```
python3
Python 3.13.5 (main, Jun 11 2025, 15:36:57) [Clang 17.0.0 (clang-1700.0.13.3)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

- Você pode executar comandos diretamente na shell

```
python3
Python 3.13.5 (main, Jun 11 2025, 15:36:57) [Clang 17.0.0 (clang-1700.0.13.3)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Olá turma")
Olá turma
>>> 10*4+3
43
>>> 10*(4+3)
70
>>>
```

Luís Felipe

20/08/25

Shell Interativa

- A shell é muito útil durante a criação de um programa pois você pode já testar partes do seu código para saber se está funcionando como esperado.
- Mas na maioria das vezes criaremos um código completo que deve ser salvo em um arquivo como a extensão .py.
- Depois, código poderá ser executado em um terminal:

```
python3 nomeArquivo.py
```

A estrutura básica de um programa é a seguinte:

```
Comando1
.
.
.
ComandoN
```

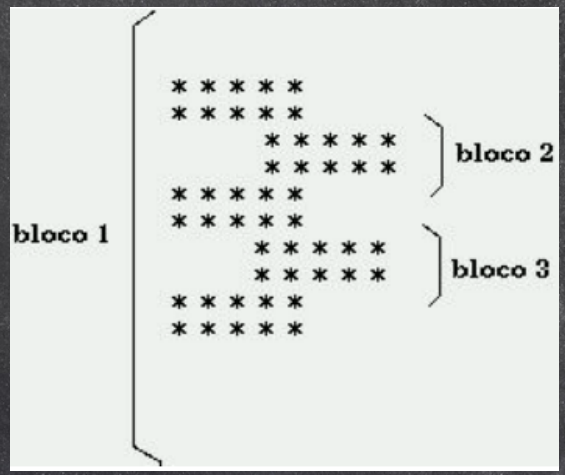
O programa deve ter um comando por linha
Os comandos serão executados de cima para baixo, um por vez.

Luís Felipe

20/08/25

Indentação

- Na escrita, os blocos são determinados por indentação, pelo alinhamento das margens esquerdas. Por exemplo:



- Bloco 1 contém blocos 2 e 3
- Blocos 2 e 3 são disjuntos

Luís Felipe

20/08/25

Exemplo - entendendo intuitivamente

```
1  idade = 20
2  nome = "Fulano"
3
4  if idade >= 18:
5      print("Você é maior de idade.")
6      if nome != "":
7          print("Nome informado:", nome)
8          for i in range(2):
9              if i == 0:
10                 print("Primeira vez no laço.")
11             print("Mensagem", i + 1)
12     else:
13         print("Nome não informado.")
14 else:
15     print("Você é menor de idade.")
16     print("Acesso negado.")
```

O que esse programa está fazendo? Façam o chinês.

Luís Felipe

20/08/25

Variáveis

- Variáveis são espaços reservados na memória do computador para armazenar informações que podem mudar durante a execução do programa.
- Cada variável possui um nome, que é usado para acessar e manipular o valor armazenado.
- Ao criar uma variável, você está dizendo ao computador: "guarde este valor com esse nome".
- Na execução do programa, esse nome está associado a um pedaço da memória RAM.
- Em Python, declaramos (ou atribuímos valor a) uma variável usando o símbolo =
  - nome_variavel = *valor*
- Exemplo:

```
1 idade = 25
2 nome = "Fulano"
```

Luís Felipe

20/08/25

Variáveis

Uma variável também é uma expressão e pode ser atribuída a outra variável.

```
>>> a = 10
>>> b = 4
>>> c = 3
>>> a*b+c
43
>>> a = a + b
>>> a
14
>>> a = a + a
>>> a
28
```

Mas, cada variável deve ter uma atribuição.

```
>>> a = 10
>>> a = a + A
Traceback (most recent call last):
  File "<python-input-8>", line 1, in <module>
    a = a + A
            ^
NameError: name 'A' is not defined. Did you mean: 'a'?
```

Luís Felipe

20/08/25

Regras básicas para variáveis

- Não deve ter acento e deve possuir apenas caracteres alfa-numéricos. Usar notação de camelo.
  Ex.: minhaNota, notaTurma, cartoesAmarelos, cartoesVermelhos.
- Deve começar com letra (maiúscula ou minúscula) ou subscrito (_).
  Nunca pode começar com um número.
- Pode conter letras maiúscula, minúscula, números e subscrito.
- Não pode utilizar como parte do nome de uma variável símbolos como:

  `{ ( + - * = / \ ; . , ?`

Luís Felipe

20/08/25

Tipos de Variáveis em Python

- As variáveis em Python possuem um tipo, que corresponde ao valor armazenado naquele momento.
- Python é uma linguagem de tipagem forte e dinâmica.
- Isso significa que você pode atribuir valores de diferentes tipos para uma mesma variável.
- O tipo da variável pode mudar ao longo da execução do programa!

```
1  a = 3                          # a é inteiro
2  print(a)
3
4  a = 90.45                      # a é número real com parte decimal
5  print(a)
6
7  a = "Olá vocês!"               # a é string
8  print(a)
```

Luís Felipe

20/08/25

Tipos de Dados em Python

- Python possui diversos tipos de dados. Os principais são:
  - **int** — números inteiros (ex: `10`, `-4`, `0`)
  - **float** — Números decimais (ex: `3.14`, `-2.5`)
  - **str** — strings, textos (ex: `"Olá"`)
  - **bool** — valores lógicos: `True`, `False`
- Outros tipos importantes:
  - **list** — sequência mutável de valores (ex: `[1, 2, 3, 3]`)
  - **tuple** — sequência imutável (ex: `(1, 2, 3)`)
  - **set** — conjunto de elementos únicos (ex: `{1, 2, 3}`)
  - **dict** — dicionário (pares chave:valor) (ex: `{"a": 1, "b": 2}`)
  - **bytes** — dados binários (ex: `b"ABC"`, "ABC" é automaticamente convertido nos códigos ASCII de A, B, C: A = 65, B = 66, C = 67)
  - **None** — representa ausência de valor
- O tipo de uma variável pode mudar durante a execução, pois Python é dinamicamente tipado.
- Os tipos são definidos dinamicamente pelo próprio Python no momento da criação. Não é preciso dizer de que tipo é cada variável

Luís Felipe

20/08/25

Variáveis inteiras – **int**

- Inteiro (**int**):
  - Pode ter centenas de dígitos, limitado apenas pela memória do computador;
  - O padrão é decimal, mas pode-se usar outras bases como binária (iniciada com 0b), octal (iniciada com 0o) ou hexadecimal (iniciada com Ox)

```
>>> a = 0b01010
>>> a
10
>>> type(a)
<class 'int'>
>>> a = 1023
>>> a
1023
>>> type(a)
<class 'int'>
>>> a = 0o01010
>>> a
520
>>> type(a)
<class 'int'>
```

Luís Felipe

20/08/25

Variáveis de tipo ponto flutuante — **float**

- **Ponto flutuante (`float`):**
  - Armazenam valores reais.
  - Possuem problemas de precisão, pois há uma quantidade limitada de memória para representar números reais.

```
>>> a = 10.0/3.0
>>> a
3.3333333333333335
>>> type(a)
<class 'float'>
```

- O float em Python segue o padrão IEEE 754, que usa 64 bits (1 para o sinal, 11 para o expoente e 52 para a mantissa: $(-1)^s \times 1, f_{52} \times 2^e$), permitindo representar com precisão de cerca de 15 a 16 dígitos decimais apenas.
- A representação de um float é em binário. Assim, só frações que são potências de 2 (ex.: 0.5, 0.25) podem ser guardadas exatamente, enquanto outras (ex.: 0.1, 1/3) viram dízimas infinitas.
- Na conversão, o computador corta a dízima após os 52 bits da mantissa e guarda a aproximação binária mais próxima possível.

Luís Felipe

20/08/25

Limitações do tipo float

- **Faixa de valores representáveis:**
  - $\simeq$ entre $2.225074 \times 10^{-308}$ e $1.797693 \times 10^{308} \simeq 2^{1023}$.
  - O expoente possui 11 bits, logo há $2^{11} = 2048$ combinações possíveis. O valor armazenado é deslocado por um *bias* de 1023, permitindo representar expoentes reais de -1022 até $+1023$. Assim, temos $\sim 10^{\pm 308}$ como limites de magnitude.
- **Overflow:**
  - Quando um número ultrapassa o valor máximo representável.
  - Resultado: `inf` (infinito). Ex: `1e309` $= 1 \times 10^{309} \Rightarrow$ `inf`
- **Underflow:**
  - Quando um número é tão próximo de zero que não pode ser representado.
  - Resultado: é arredondado para `0.0`. Ex: `1e-400` $= 1 \times 10^{-400} \Rightarrow$ `0.0`
- **Precisão:**
  - Cerca de 15 a 16 (as vezes 17) dígitos decimais significativos.
  - Exemplo: `0.1234567891234567`$9 \Rightarrow$ `0.1234567891234568`
- **Erro de representação binária:**
  - Nem todos os decimais têm representação exata. Ex: `0.1 + 0.2 = 0.3000000000000000`4. Alternativa: round(0.1 + 0.2, 1) $\Rightarrow$ 0.3

Luís Felipe

20/08/25

Variáveis do tipo string

- String (**str**):
  - Armazenam texto. Uma constante do tipo string deve estar entre aspas simples ou aspas duplas. Ex.: 'Olá Brasil' ou "Olá Brasil".

```
>>> a = "Ola Brasil"
>>> a
'Ola Brasil'
>>> a = 'Ola Brasil'
>>> a
'Ola Brasil'
>>> type(a)
<class 'str'>
```

Luís Felipe

20/08/25

Variáveis do tipo booleano — bool

- Booleano (**bool**):
  - Representam valores lógicos: True ou False.
  - São muito usados em estruturas de decisão e repetição.

```
>>> a = True
>>> b = False
>>> type(a)
<class 'bool'>
>>> a and b
False
>>> a or b
True
>>> not a
False
>>> idade = 20
>>> maior_de_idade = idade >= 18
>>> maior_de_idade
True
>>> type(maior_de_idade)
<class 'bool'>
```

A	B	A and B	A or B
True	True	True	True
True	False	False	True
False	True	False	True
False	False	False	False

A	B	if A then B	not A or B
True	True	True	True
True	False	False	False
False	True	True	True
False	False	True	True

Luís Felipe

20/08/25

Simulando if ... then:

Obs.: Ainda veremos comandos condicionais com mais detalhes em aulas futuras.

```
1  a = True
2  b = False
3
4  # Simulando "se a então b"
5  if not a or b:
6      print("A implicação 'se a então b' é verdadeira.")
7  else:
8      print("A implicação 'se a então b' é falsa.")
```

Luís Felipe

20/08/25

Tipagem forte e mudança de tipo

- Uma vez que uma variável tenha um valor de um tipo, ele não pode ser usado como se fosse de outro tipo.

```
>>> a = 10
>>> b = '20'
>>> c = a + b
Traceback (most recent call last):
  File "<python-input-26>", line 1, in <module>
    c = a + b
        ~~^~~
TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

- A não ser que alteremos o tipo da variável. Usar int() ou float() para converter o tipo para numérico; bool() para tipo booleano; str() para tipo string; list() para tipo lista.

```
>>> a = 10
>>> b = '20'
>>> b = int(b)
>>> a + b
30
```

```
>>> idade = "25"
>>> idade = int(idade)
>>> metade = idade / 2
>>> print("Medade da idade é: ", metade)
Medade da idade é:  12.5
>>> type(metade)
<class 'float'>
```

Luís Felipe

20/08/25

Entrada

- Para entrada de dados, usamos o comando input().
- É possível informar um texto que será exibido na tela, ajudando o usuário a entender que o programa está aguardando uma entrada.
- O comando input realiza a leitura de dados a partir do teclado: ele pausa a execução do programa até que o usuário digite um valor.
- O valor digitado é então atribuído a uma variável.
- Importante: todos os dados lidos pelo input são do tipo string, mesmo que pareçam números.

```
>>> a = input()
20
>>> type(a)
<class 'str'>
```

```
a = input("Digite algo:")
print("Você digitou:", a)
print("Tipo de variável", type(a))

Digite algo:Prog é legal
Você digitou: Prog é legal
Tipo de variável <class 'str'>
```

- Assim, para alterar um dado de entrada para um tipo numérico, utilizemos int() ou float():

```
altura = int(input('Digite a altura do triangulo: '))
print(type(altura))
```

Luís Felipe

20/08/25

Saída

- Para saída de dados, usamos print.
- Para imprimir um texto, utilizamos o comando print. O texto pode ser uma constante do tipo string.

```
print('Prog é muito legal')
print(123)
altura = 10
print(altura)
print('O nome do aluno é', nome)
print('Vamos pular uma linha \n')
```

O símbolo especial \n é responsável por pular uma linha na saída

```
print("Olá Pessoal!\n Olá Pessoal")

Saída:
Olá Pessoal!
Olá Pessoal!
```

Luís Felipe

20/08/25

Comandos de saída padrão

- `print()`
  Pula para a próxima linha.

- `print(exp1)`
  Escreve na saída o resultado da avaliação da expressão `exp1`. Ao final, pula para a próxima linha.

- `print(exp1, end = "conteudo_final")`
  Escreve na saída o resultado da avaliação da expressão `exp1`. Ao final, escreve a string que consta após `end =` .

- `print(exp1, exp2, ..., expN, end = "conteudo_final")`
  Escreve na saída o resultado da avaliação de cada expressão `exp1`, `exp2`, ..., `expN` separando os resultados por um espaço em branco. Ao final, escreve a string que consta após `end =` .

Luís Felipe

20/08/25

Expressões Formatadas com %

- **Sintaxe Geral:**

  `". . . %[largura].[precisão][tipo] . . . " % (valores)`

- **Componentes:**
  - `largura (opcional)`: número mínimo de caracteres a serem exibidos
  - `.precisão (opcional)`: número de casas decimais (para floats) ou caracteres (para strings)
  - `tipo`: tipo do dado:
    - `d` → inteiro
    - `f` → ponto flutuante
    - `s` → string

- **Exemplo:**

```
1 valor = 3.14159
2 print("Resultado = %6.2f" % valor)
```

- **Saída:** `Resultado = __3.14`
  Obs.: `_` representa espaço em branco; total de 6 caracteres, incluindo ponto e decimais

Luís Felipe

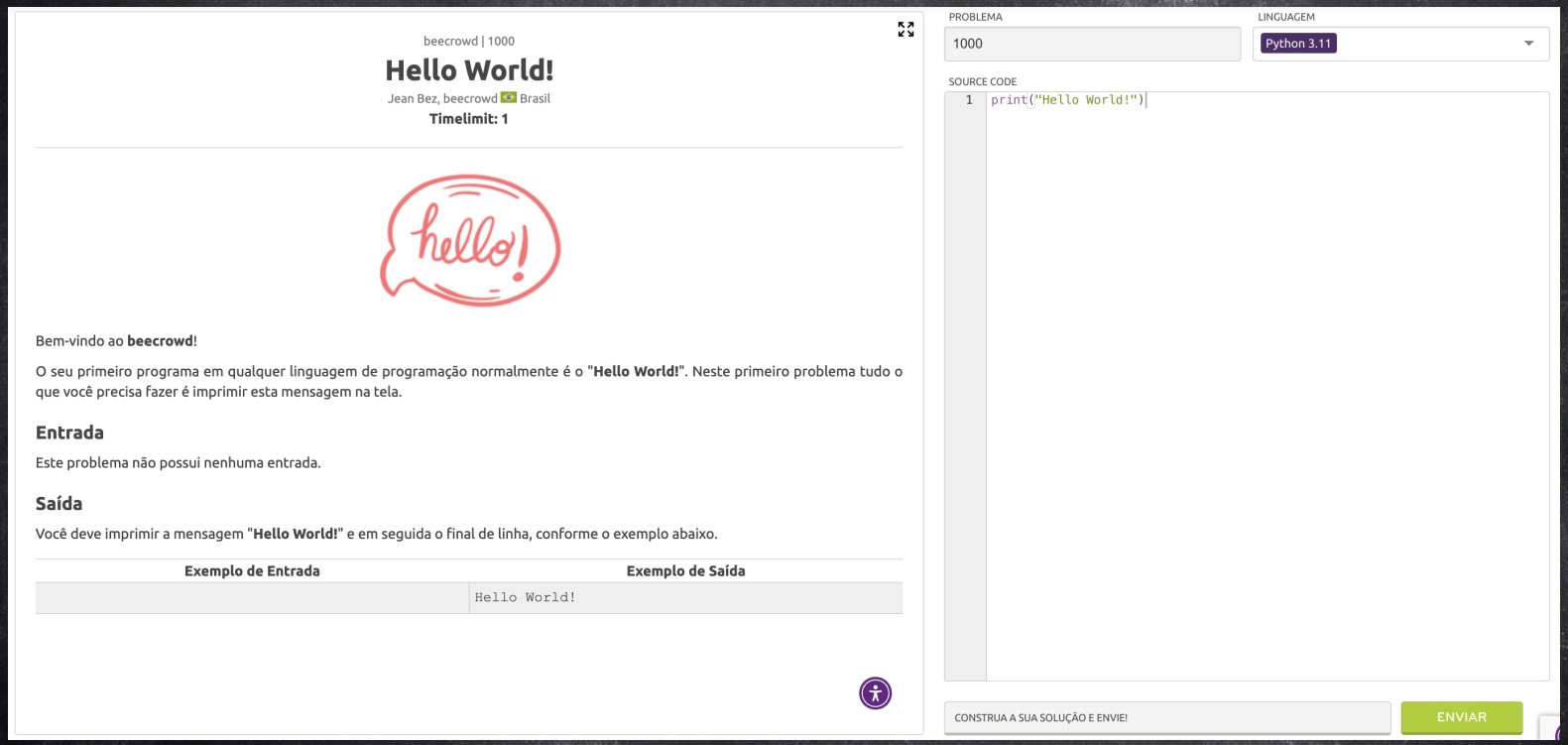
20/08/25

Exercícios

Problemas 1000 ao 1009 do Beecrowd

https://judge.beecrowd.com/pt/problems/index/1

https://judge.beecrowd.com/pt/problems/view/1000



Luís Felipe

20/08/25

Exercícios Práticos

1. Faça um programa que:
   - Leia o **nome**, **idade**, **altura**, **peso** e **nacionalidade** do usuário;
   - Escreva essas informações em um **parágrafo de apresentação**.

2. Faça um programa que:
   - Leia o **raio** de uma circunferência;
   - Exiba o **perímetro** da circunferência.

3. Faça um programa que:
   - Leia dois **pontos** em um plano bidimensional;
   - Calcule e exiba a **distância** entre esses pontos.