

PROGRAMAÇÃO DE COMPUTADORES

TIPAGEM

COPYRIGHT © 2023 DIATINF/CNAT/IFRN
JORGIANO VIDAL



- Tipos de dados
- Conversão de tipos
- Mais strings
- Formatação





- Inteiro
 - 1234
- Real
 - 1234,56
- String
 - "Isto é um texto"
 - 'Isto também é um texto'



A B C D E F G H I J K L
M N O P Q R S T U V W
X Y Z À Á Ê Ë Ì abcdefghij
klmnopqrstuvwxyz à á â ã
& 1 2 3 4 5 6 7 8 9 0 (\$ % , . ! ?)

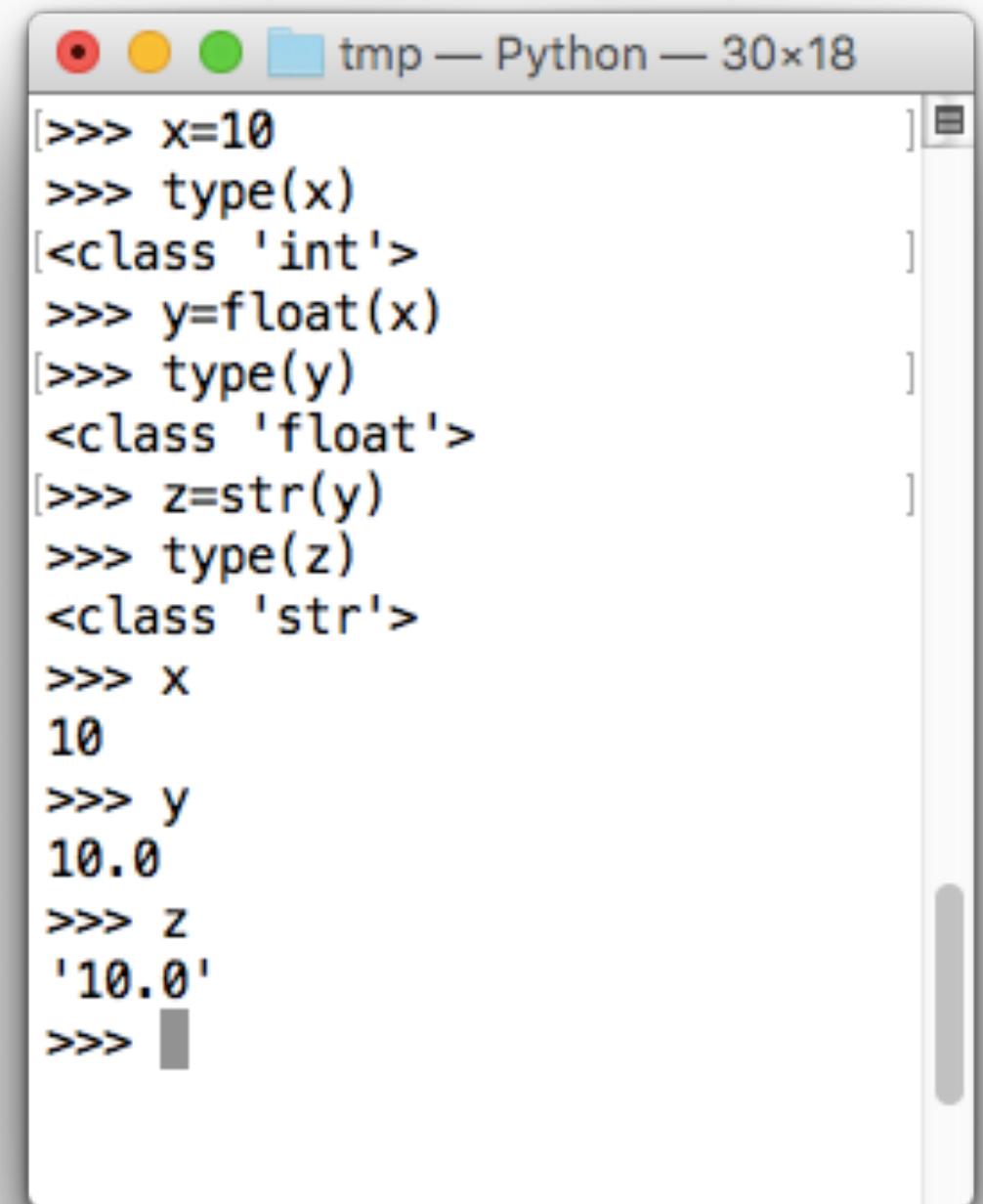


- Para saber o tipo
`type(valor)`
- Converte um para outro
 - Para inteiro
 - função `int()`
 - Para real
 - função `float()`
 - Para string
 - função `str()`



CONVERSÃO DE TIPOS

- Para saber o tipo
`type(valor)`
- Converte um para outro
 - Para inteiro
 - função `int()`
 - Para real
 - função `float()`
 - Para string
 - função `str()`



```
[>>> x=10
>>> type(x)
<class 'int'>
>>> y=float(x)
>>> type(y)
<class 'float'>
>>> z=str(y)
>>> type(z)
<class 'str'>
>>> x
10
>>> y
10.0
>>> z
'10.0'
>>> ]
```



- Quantidade de caracteres (tamanho)
- Função `len(str)`

```
nome=input()  
t=len(nome)  
print("Seu nome tem",t,"caracteres")
```



- Quantidade de caracteres (tamanho)
- Função `len(str)`

```
nome=input()  
t=len(nome)  
print("Seu nome tem",t,"caracteres")
```

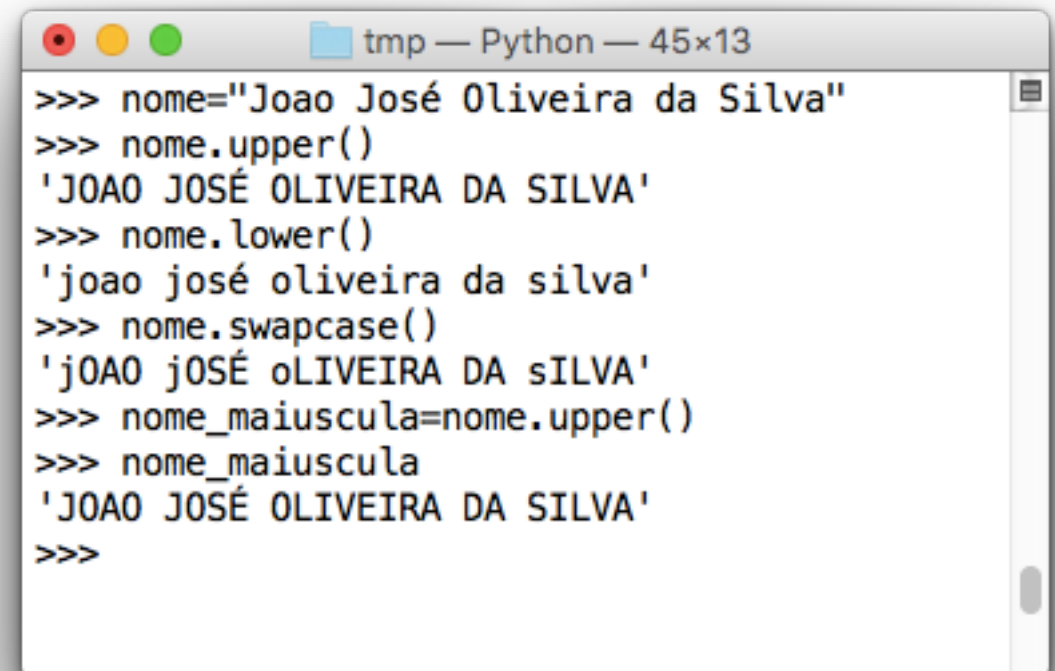


- Quantidade de caracteres (tamanho)
- Função `len(str)`

```
nome=input()  
t=len(nome)  
print("Seu nome tem",t,"caracteres")
```




- Transformação
 - Método `upper()`
 - Gera uma nova string com todos os caracteres maiúsculos
 - Método `lower()`
 - gera um nova string com todos os caracteres minúsculos
 - Método `capitalize()`
 - Gera uma nova string com a primeira maiúscula e as outras minúsculas
 - Método `swapcase()`
 - Gera uma nova string com as letras minúsculas convertidas para maiúsculas e vice-versa



```
tmp — Python — 45x13
>>> nome="Joao José Oliveira da Silva"
>>> nome.upper()
'JOAO JOSÉ OLIVEIRA DA SILVA'
>>> nome.lower()
'joao josé oliveira da silva'
>>> nome.swapcase()
'jOAO jOSÉ oLIVEIRA DA sILVA'
>>> nome_maiuscula=nome.upper()
>>> nome_maiuscula
'JOAO JOSÉ OLIVEIRA DA SILVA'
>>>
```



- Cria *string* com junção de outras *strings*
- Operador +

```
s1="IFRN"  
s2="CNAT"  
s3="DIATINF"  
lugar=s1+s2+s3  
print(lugar)  
lugar=s1+'/' +s2+'/' +s3  
print(lugar)
```

```
tmp — -bash — 44x5  
IFRNCNATDIATINF  
IFRN/CNAT/DIATINF  
$
```



OPERAÇÕES COM TIPOS DIFERENTES



OPERAÇÕES COM TIPOS DIFERENTES

- Qual o resultado?

`x = "12" + "23"`

`x = "12 + 23"`

`x = 12 + 23`



OPERAÇÕES COM TIPOS DIFERENTES

- Qual o resultado?

`x = "12" + "23"` `"1223"`

`x = "12 + 23"`

`x = 12 + 23`



OPERAÇÕES COM TIPOS DIFERENTES

- Qual o resultado?

`x = "12" + "23"` `"1223"`

`x = "12 + 23"` `"12 + 23"`

`x = 12 + 23`



OPERAÇÕES COM TIPOS DIFERENTES

- Qual o resultado?

`x = "12" + "23"` `"1223"`

`x = "12 + 23"` `"12 + 23"`

`x = 12 + 23` `35`



OPERAÇÕES COM TIPOS DIFERENTES

- Qual o resultado?

`x = "12" + "23"` `"1223"`

`x = "12 + 23"` `"12 + 23"`

`x = 12 + 23` `35`

- Não há soma/subtração entre números e strings

`x="12"+23`



OPERAÇÕES COM TIPOS DIFERENTES

- Qual o resultado?

`x = "12" + "23"` `"1223"`

`x = "12 + 23"` `"12 + 23"`

`x = 12 + 23` `35`

- Não há soma/subtração entre números e strings

`x="12"+23`

- Necessário converter um dos operandos

- `x = "12"+str(23)`

- `x = int("12")+23`



- Multiplicação de string por inteiro

`x="Texto"*5`

O valor de x é:

`"TextoTextoTextoTextoTexto"`



FORMATAÇÃO DE NÚMEROS



FORMATAÇÃO DE NÚMEROS

- Gera uma *string* com formato definido
 - Sintaxe: `"FORMATO".format(VALOR)`
 - Gera uma string de VALOR de acordo com o FORMATO



FORMATAÇÃO DE NÚMEROS

- Gera uma *string* com formato definido
 - Sintaxe: `"FORMATO".format(VALOR)`
 - Gera uma string de VALOR de acordo com o FORMATO
- Exemplo
`media_formatada="{:.2f}".format(media)`
 - Cria uma string do valor armazenado na variável `media` com duas casas decimais reais e armazena na variável `media_formatada`



FORMATAÇÃO DE NÚMEROS

- Gera uma *string* com formato definido
 - Sintaxe: `"FORMATO".format(VALOR)`
 - Gera uma string de VALOR de acordo com o FORMATO
- Exemplo
 - `media_formatada="{:.2f}".format(media)`
 - Cria uma string do valor armazenado na variável `media` com duas casas decimais reais e armazena na variável `media_formatada`
 - `seq='{05d}'.format(num)`
 - Cria uma string do valor armazenado na variável `num` de um inteiro com zeros a esquerda, ocupando pelo menos cinco espaços, e armazena na variável `seq`



- Muitas opções
- Detalhes em <https://pyformat.info>
- Formatar números inteiros

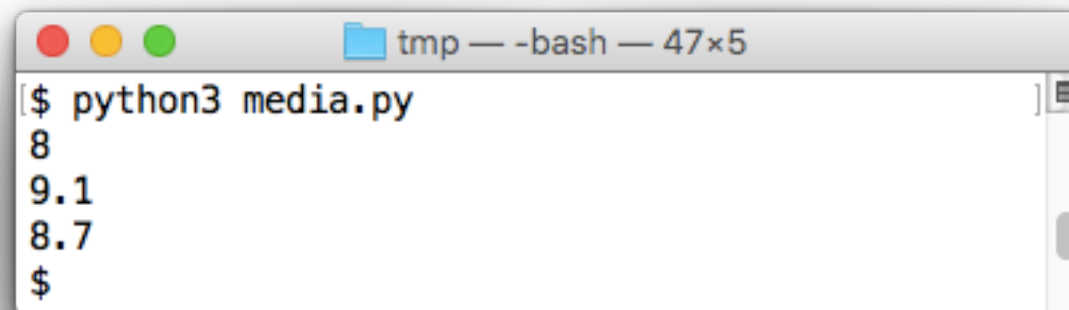
Expressão	resultado
'{:d}'.format(123)	"123"
'{:5d}'.format(123)	" 123"
'{:05d}'.format(123)	"00123"
'{:+d}'.format(123)	"+123"

- Formatar números reais

Expressão	resultado
'{:f}'.format(123)	"123.000000"
'{:12f}'.format(123)	" 123.000000"
'{:5.2f}'.format(10/3)	" 3.33"
'{:,.2f}'.format(10/3)	"3.33"



```
nota1=float(input())
nota2=float(input())
media=(nota1*2+nota2*3)/5.0
media_str="{:.1f}".format(media)
print(media_str)
```



A terminal window titled "tmp — -bash — 47x5" showing the execution of a Python script. The prompt is "\$". The user enters "python3 media.py". The script outputs "8", "9.1", and "8.7" on separate lines. The prompt "\$" appears again.

```
tmp — -bash — 47x5
[$ python3 media.py
8
9.1
8.7
$]
```




```
nota1=float(input())
nota2=float(input())
media=(nota1*2+nota2*3)/5.0
media_str="{:.1f}".format(media)
print(media_str)
```

```
tmp — -bash — 47x5
$ python3 media.py
8
8.1
8.7
$
```

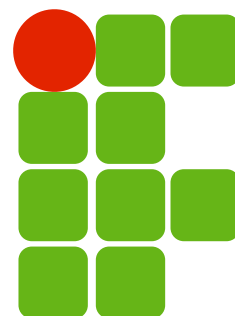


```
nota1=float(input())
nota2=float(input())
media=(nota1*2+nota2*3)/5.0
media_str="{:.1f}".format(media)
print(media_str)
```



```
nota1=float(input())
nota2=float(input())
media=(nota1*2+nota2*3)/5.0
media_str="{:.1f}".format(media)
print(media_str)
```

```
nota1=float(input())
nota2=float(input())
media=(nota1*2+nota2*3)/5.0
print("{:.1f}".format(media))
```



PROGRAMAÇÃO DE COMPUTADORES

TIPAGEM

COPYRIGHT © 2023 DIATINF/CNAT/IFRN
JORGIANO VIDAL