28 - Documentacao, alguns string methods

Para ver a documentação das funções built-in do python veja o link:

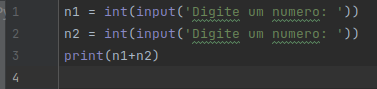
<https://docs.python.org/3/library/stdtypes.html>

Obs: já de agora, aprenda a ler a documentação, isso é muito importante para linguagem e frameworks.

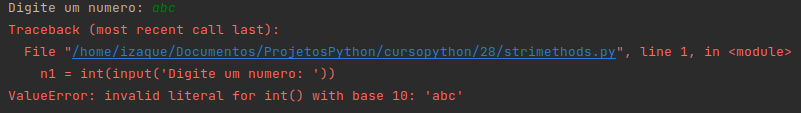
Já aprendemos a realizar **type casting**, mas sabemos que se realizarmos int(<str>) e <str> não for um número inteiro, o interpretador retorna um ERRO interrompendo a execução do programa.

É claro que não queremos erro em um programa, então como boa prática de programação deveríamos evitar esses erros.

Por exemplo, veja esse código:

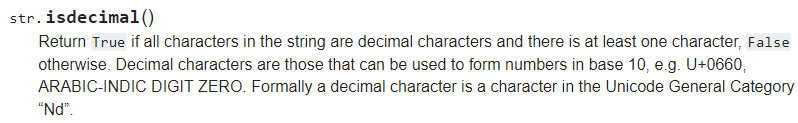


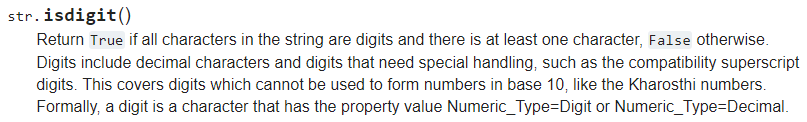
Se o usuário não digitar um número inteiro o programa retornará um erro:

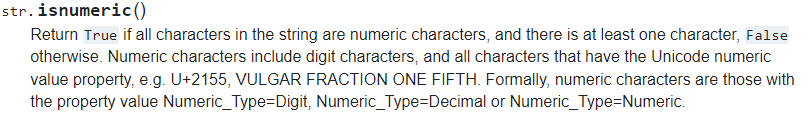


Devemos evitar isso, uma solução possível (e para já nos introduzirmos a alguns strings methods) é utilizar os métodos: **<str>.isdecimal() , <str>.isdigit() , <str>.isnumeric()**

Leia a documentação deles:







Links que podem ajudar com o que foi dito nas documentações deles:

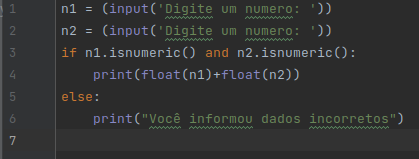
<https://en.wikipedia.org/wiki/Template:General_Category_(Unicode)#cite_note-fn3-4>

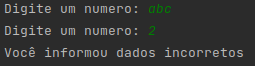
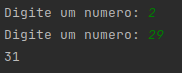
<https://en.wikipedia.org/wiki/Unicode_character_property#Numeric_values_and_types>

Especificamente no tópico Decimal deste segundo link ele faz a distinção do que seria os conjunto dos caracteres decimal, digit e numeric.

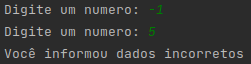
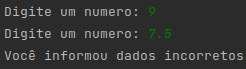
The characters that do have a numeric value are separated in three groups: Decimal (De), Digit (Di) and Numeric (Nu, i.e. all other). "Decimal" means the character is a straight decimal digit. Only characters that are part of a contiguous encoded range 0..9 have numeric type Decimal. Other digits, like superscripts, have numeric type Digit. All numeric characters like fractions and Roman numerals end up with the type "Numeric"

Portanto, uma solução seria:

^

Contudo essa solução não está completamente correta, pois não cobre os números negativos e com ponto flutuante:

Uma maneira de resolver isso seria utilizar **try except** (não se preocupe tanto com isso agora pois veremos novamente mais pre frente)

Sintaxe:

**try:**

**<bloco do try>**

**except:**

**<bloco do except>**

Ele tenta fazer executar os comandos que estão no bloco do try, caso algum deles der erro ele executará apenas os comandos do bloco do except.

Uma maneira de resolver isso seria utilizar **try except** (não se preocupe tanto com isso agora pois veremos novamente mais pre frente)

Sintaxe:

**try:**

**<bloco do try>**

**except:**

**<bloco do except>**

Ele tenta fazer executar os comandos que estão no bloco do try, caso algum deles der erro ele executará apenas os comandos do bloco do except.

