
python

Try, except

Tratamento de erro

O **Error Handling** é muito útil para um programador categorizar e entender rapidamente que tipo de erros podem estar ocorrendo durante a execução de seu programa.

Tratamento de erro

Como programador, essa percepção o tornará mais eficiente e mais confiante sobre como escrever código

try except

É comum para um programador, se deparar com erros sejam ele de sintaxe, de divisão por zero, dentre outros. Vamos analisar alguns deles, inclusive erros que acompanhamos durante nosso aprendizado.

try except

```
try:
    código
except:
    mensagem
```

try except

O que acontece se executarmos o código abaixo

```
print(x)
```

try except

```
Traceback (most recent call last):
  File "C:\CDD40\AULA-TRY EXCEPTION\main.py",
line 2, in <module>
    print(x)
NameError: name 'x' is not defined
```

try except

no código podemos verificar que mandamos imprimir x, mas x não existe

```
print(x)
```

NameError: name 'x' is not defined

try except

NameError: **Receberemos este erro, quando o objeto não foi encontrado, como no exemplo do código acima.**

try except

Mas como tratar esses erros?

try except

Você receberá um SyntaxError quando cometer um erro de sintaxe no Python. Pode ser uma aspa ou parênteses ausentes.

try except

Você receberá um TypeError quando aplicar uma operação ou função ao tipo errado de dados, como aplicar operações aritméticas a strings.

try except

Você receberá um TypeError quando aplicar uma operação ou função ao tipo errado de dados, como aplicar operações aritméticas a strings.

try except

Você receberá um IndexError ao tentar alcançar um índice fora dos limites de seus dados.

try except

KeyError é como IndexError para dicionários. Se você tentar acessar uma chave que não está incluída em seu dicionário, receberá um KeyError

try except

O erro de AttributeError: ocorre quando você tenta usar um atributo ou método que não se aplica aos dados específicos em que está trabalhando. Por exemplo, tentar aplicar o método .reverse() em uma string

try except

O **Value Error** ocorre quando você aplica uma função a um tipo de dados corretamente, mas o conteúdo não é adequado para essa operação. Por exemplo, você pode aplicar int() a uma string de números como: int(“9999”)
mas você não pode converter letras em números inteiros, então o seguinte não funcionará: int(“gatinha”)

try except

Quando você recebe um **Syntax Error**, isso significa que você está cometendo um erro de gramática no Python. E Python não consegue entender e não consegue compilar seu código. Geralmente é fácil de corrigir, pois a sintaxe é definitiva e você só precisará corrigir seu erro na linha relacionada ao erro.

Por exemplo:

você está digitando um caractere extra de forma errada, você está digitando uma palavra-chave Python errada,

try except

erros de parênteses também são muito comuns, você pode esquecer ou digitar aspas incorretamente, sua linha pode precisar de recuo (geralmente em instruções if), você pode estar esquecendo de dois pontos (comum com instruções if, definindo funções, construindo classes etc.) às vezes, um espaço extra causará um erro de sintaxe.

try except

Também chamados de Erros Lógicos, os **Semantic Error** são uma questão de significado. Não é um problema de gramática, ou na linguagem de programação, a sintaxe é boa e o programa compila, mas os resultados não são o que pretendíamos ter. Seu programa será executado com sucesso, mas o resultado estará em qualquer lugar entre ligeiramente errado a completamente errado.

Por exemplo: Esquecer de dividir um resultado percentual por 100

Resultado errado causado por ordem errada de operadores aritméticos

Manipulando arquivos

no dia a dia do programador é comum precisarmos abrir arquivos

Manipulando arquivos

```
with open('arquivo.txt', "a" ) as arquivo:

    texto=input("Digite um texto: ")
    arquivo.write(f'{texto} \n')
```