youth! space

PROJETO </> EDUCAÇÃO

DO FUTURO

MÓDULO 03

TRY EXCEPT

Tratamento de Erros

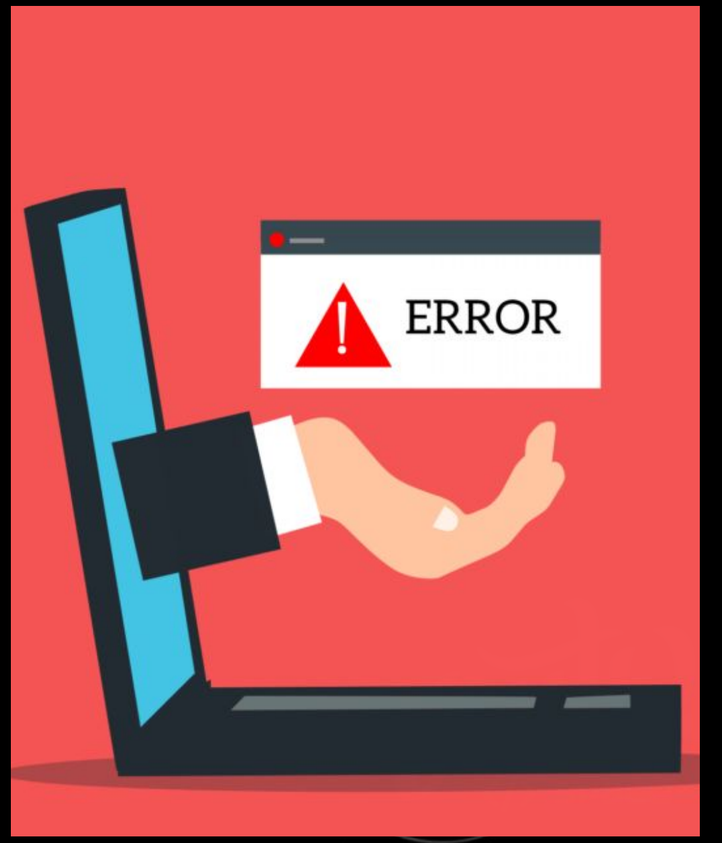
AULA 03

Erros no Python

Em Python, uma variedade de erros pode ocorrer durante a execução do programa. Estes erros podem ser causados por problemas de sintaxe, indentação inadequada, variáveis ou funções não definidas, tentativas de divisão por zero e etc.

Compreender esses erros é essencial para identificá-los e corrigi-los adequadamente durante o desenvolvimento do programa, e o uso de blocos try-except pode ajudar a lidar com essas situações de forma eficaz, garantindo a robustez e a integridade do código Python.

O Python fornece uma estrutura chamada try-except que permite lidar com exceções durante a execução do programa. Isso ajuda a tornar o código mais robusto e a lidar com possíveis erros de forma elegante.



TRY EXCEPT

O bloco try-except em Python permite que o código tente executar uma série de instruções que podem gerar exceções, e em caso de falha, captura essas exceções e executa um bloco de código alternativo, em vez de interromper abruptamente a execução do programa.

Essa estrutura é usada para lidar com situações em que erros podem ocorrer de forma imprevisível, como operações de entrada de dados do usuário, acesso a arquivos ou conexões de rede.

```
try:
    # Código que pode gerar uma exceção
    # ...
except ExcecaoTipo:
    # Código a ser executado se a
exceção do tipo ExcecaoTipo for lançada
    # ...
```

TRY EXCEPT - Erro Genérico

O bloco try-except em Python pode ser usado para capturar e tratar qualquer tipo de exceção que ocorra dentro do bloco try. Ele permite que você especifique quais tipos de exceções deseja capturar e como deseja lidar com elas no bloco except correspondente.

```
try:
    # Código que pode gerar exceções
    x = 10 / 0  # Tentativa de divisão por zero
    y = int("abc")  # Tentativa de converter uma string em um número inteiro
except Exception :  # Captura qualquer outra exceção não especificada anteriormente
    print("Ocorreu um Erro")
```

Neste exemplo, o bloco except não especifica nenhum tipo de exceção, o que significa que ele capturará qualquer tipo de exceção que ocorra dentro do bloco try. Quando um erro é encontrado (como divisão por zero ou conversão de uma string inválida para um número inteiro), o bloco except é executado e a mensagem "Ocorreu um erro." é exibida.

Atividades

1. Crie um programa que solicite ao usuário dois números e calcule a soma deles. Use try e except para tratar quaisquer erros que possam ocorrer (como entradas não numéricas)

1. Crie um programa que solicite ao usuário um índice e tente acessar um elemento em uma lista. Use try e except para tratar quaisquer erros que possam ocorrer (como o índice estar fora do intervalo ou não ser um número).

Atividades

3. Crie um programa que solicite ao usuário uma chave e tente acessar o valor correspondente em um dicionário. Use try e except para tratar o caso de a chave não existir no dicionário.

4. Crie um programa que define uma classe simples e instancia um objeto dessa classe. Solicite ao usuário um nome de atributo e tente acessar esse atributo no objeto. Use try e except para tratar o caso de o atributo não existir.

TRY EXCEPT - Classes de Erros

Aqui estão algumas das exceções integradas mais comuns em Python que você pode capturar e tratar em um bloco try-except:

- ZeroDivisionError: Ocorre quando uma tentativa de divisão por zero é feita.
- ValueError: Ocorre quando uma função recebe um argumento com o tipo correto, mas um valor inadequado.
- TypeError: Ocorre quando uma operação é realizada em um tipo de dado incompatível.
- IndexError: Ocorre quando um índice está fora do intervalo em uma sequência (por exemplo, uma lista, tupla ou string).
- KeyError: Ocorre quando uma chave específica não está presente em um dicionário.
- FileNotFoundError: Ocorre quando um arquivo não pode ser encontrado no sistema de arquivos.
- IOError: Ocorre quando ocorre um erro relacionado à operações de entrada/saída (I/O), como falha na leitura ou gravação de um arquivo.
- AttributeError: Ocorre quando uma tentativa é feita para acessar um atributo que não existe.

Essas são apenas algumas das exceções integradas em Python que você pode capturar e tratar em um bloco try-except. Lembre-se de que você também pode criar suas próprias exceções personalizadas para situações específicas do seu código, estendendo a classe Exception

Atividades

5. Faça as atividades anteriores incluindo as classes de exceções que se aplicam aos erros mais específicos de cada caso.

TRY EXCEPT - Exemplo de Uso

```
try:
    numero = int(input("Digite um número: "))
    resultado = 10 / numero
    print("O resultado da divisão é:", resultado)
except ZeroDivisionError:
    print("Erro: Divisão por zero não é
permitida.")
except ValueError:
    print("Erro: Entrada inválida. Você deve
digitar um número.")
```

TRY EXCEPT - Erro Personalizado

Para estender a classe Exception em Python e criar suas próprias exceções personalizadas, você pode definir uma nova classe que herda de Exception. Aqui está um exemplo básico:

```
class ExcecaoPersonalizada(Exception):
    def __init__(self, mensagem="Ocorreu um erro personalizado."):
        self.mensagem = mensagem
        super().__init__(self.mensagem)
```

```
try:
    # Código que pode gerar a exceção personalizada
    raise ExcecaoPersonalizada("Esta é uma mensagem de erro personalizada.")
except ExcecaoPersonalizada as e:
    print("Erro personalizado capturado:", e)
```

Neste exemplo, uma instância de ExcecaoPersonalizada é criada e lançada dentro do bloco try. O bloco except corresponde a essa exceção personalizada e imprime a mensagem de erro associada. Este é apenas um exemplo básico de como estender a classe Exception em Python para criar suas próprias exceções personalizadas.

(85) 98524-9935 youthidiomas

contato@youthidiomas.com.br

www.youthidiomas.com.br

2023