AULA 13 - Manipulação de diretórios

Manipulação de arquivos Leitura Gravação Renomeação Exclusão

4.6. Manipulação de arquivos

A palavra-chave with em Python é usada para criar um contexto controlado para um bloco de

código. É usado com objetos que precisam ser gerenciados, como arquivos, conexões de banco de

dados ou recursos do sistema que precisam ser abertos e fechados adequadamente. A principal

vantagem do uso de with é que ele garante que os recursos sejam liberados automaticamente

quando não são mais necessários, mesmo se ocorrerem exceções no bloco de código.

A sintaxe básica do with é a seguinte:

```
with open('exemplo.txt', 'r') as arquivo:
    conteudo = arquivo.read()
    print(conteudo)

# O arquivo é automaticamente fechado
após sair do bloco "with"
```

Neste exemplo, o with é usado com open() para abrir um arquivo chamado "exemplo.txt" no modo de leitura ('r'). O arquivo é automaticamente fechado quando saímos do bloco with, garantindo que os recursos do sistema sejam liberados corretamente.

```
import os

# Criar um arquivo  "with"
with open('novo_diretorio', 'w') as novo_arquivo:
```

```
   os.mkdir('novo_arquivo')

# Um arquivo é criado e automaticamente fechado após
# sair do bloco "with"

with open('exemplo.txt', 'r') as arquivo:
   conteúdo = arquivo.read()
   print(conteúdo)

import os

# Renomeando o arquivo
os.rename('arquivo.txt', 'novo_nome.txt')
```

MANIPULANDO O DIRETÓRIO

Exemplo 1: Criar um novo Arquivo

```
import os
with os.scandir('c:/Users/aluno/Desktop/aula12/') as entrada:
    for arquivo in entrada:
      print(f'Diretório encontrado: {arquivo.name}')
```

Exemplo 2: Visualizar o nome dos arquivos do diretório

```
import os
 with os.scandir('C:/caminho da pasta(barra ao contrário)') as entrada:
   for arquivo in entrada:
      if arquivo.is_file():
```

```
        print(f'Arquivo encontrado: {arquivo.name}')

with os.scandir('C:/Users/aluno/Downloads/teste') as entrada :
    for n in entrada:
      if 'teste.txt':
        with open('C:/Users/aluno/Downloads/teste/teste.txt', 'r')  as t:
            content = t.read()

print(content) # fora do with, caso contrário ele irá se comportar
# como o loop
```

Exemplo 3: Renomear um Diretório

```
import os
os.rename('exemplo.txt', 'test5.txt')
```

Exemplo 4: Remover um Diretório

```
import shutil
shutil.rmtree('c:/Users/aluno/Desktop/aula12/')
```

Exemplo 5: Listar Arquivos em um Diretório

```
import os
with os.scandir('meu_diretorio') as entrada:
  for arquivo in entrada:
    if arquivo.is_file():
      print(f'Arquivo encontrado: {arquivo.name}')
```

Exemplo 6: Copiar o diretório todo

```
import shutil
shutil.copytree('original', 'nome da copia')
# serve para pastas -> diretórios

#Esses exemplos são mais concisos e realizam operações comuns de
#manipulação de diretórios usando a estrutura with.
```

Certifique-se de que o código seja executado em um ambiente
em que você tenha as permissões necessárias para as operações
de sistema de arquivos.

Funções utilizadas:

open() abrir

rename() renomear

read() ler

copy() copiar

write () escrever (criar)

‘a’ inserir novos dados no final do arquivo

‘w’ Escrita, vai substituir o arquivo existente

‘r’ Leitura

‘x’ retorna erro caso o arquivo já exista

‘t’ modo de texto padrão

‘+’ atualizar leitura e escrita

EXERCÍCIOS:

Utilizando Python:

Exercício 1: Criar e ler um Arquivo

Exemplo 2: Entrar em um Diretório

Exercício 3: Renomear um Diretório

Exercício 4: Remover um Diretório

Exercício 5: Listar Arquivos em um Diretório

Exercício 6: Copiar Arquivos em um Diretório

Subam Github **

Leitura Gravação Renomeação Exclusão

Resolução

Match Case

Semelhante as condicionais o Match Case, faz o computador fazer escolhas.

Conhecido também como Strutural Pattern Matching. Estrutura Padrão Correspondente.

exemplo 1

```
n = 10
if n == 0:
  print("zero")
else:
  print('Menor ou maior')

n = 0
match n:
   case 0:
```

```
      print("é zero")
   case 1:
      print("Menor ou maior")
   case 2:
      print('É dois...')
   case _:
      print('Algo desconhecido')
```

exemplo 2

```
a  = 20
b  = 30
match a,b:
   case 0:
      print('É zero')
   case 20:
      print('é 20')
   case _:
       print('Não é nada')
```

2.b

```
a  = int(input())
b  = int(input())
match a,b:
   case 0 , 1:
      print('Acesso ok')
```

```
    case _:
        print('Tente novamente')
```

exemplo 3

Com funções:

Se no if temos Isso:

```
def check_with_if(value):
    if isinstance(value, str):
        print("É uma string")
    elif isinstance(value, int):
        print("É um inteiro")
    elif isinstance(value, float):
        print("É um número de ponto flutuante")
    else:
        print("Tipo não reconhecido")

check_type_with_if("Olá")  # Saída: É uma string
check_type_with_if(42)     # Saída: É um inteiro
check_type_with_if(3.14)   # Saída: É um número de ponto flutuante
```

Com Match Case:

```
def check_with_match(value):
    match value:
        case str():
            print("É uma string")
```

```
    case int():
        print("É um inteiro")
    case float():
        print("É um número de ponto flutuante")
    case bool():
        print("É um número de boleano")
    case _:
        print("Desconhecido")

check_type_with_match("Olá")  # Saída: É uma string
check_type_with_match(42)    # Saída: É um inteiro
check_type_with_match(3.14)   # Saída: É um número de ponto flutuante
```

exemplo 4:

```
num = int(input())

match num:
  case 0:
    print ("Zero")
  case x if x > 0:
    print ("Positivo")
  case _:
    print ("Negativo")
```

EXERCÍCIOS:

1: Verificando se o número é par ou ímpar

2: Verificando se um número é positivo, negativo ou zero

3: Verificando se uma string é vazia ou não

4: Verificando se um número é maior, menor ou igual a 10

5: Classificando uma idade em faixas etárias - criança, jovem, adulto, idoso