Manipulação de Arquivos em Python

Índice

Explicações Adicionais

- **Abrindo Arquivos:** O processo de abrir um arquivo para leitura ou escrita.
- **Lendo Arquivos:** O processo de obter dados de um arquivo.
- **Escrevendo em Arquivos:** O processo de adicionar dados a um arquivo.
- **Fechando Arquivos:** O processo de liberar o arquivo após o uso.
- **Gerenciador de Contexto:** Usar a declaração `with` para garantir que o arquivo seja fechado corretamente.
- **Tratamento de Exceções:** Lidar com erros que podem ocorrer durante a manipulação de arquivos.

Abrindo Arquivos

Para abrir um arquivo em Python, você usa a função `open()`. Você pode especificar o modo de abertura, como leitura (`'r'`), escrita (`'w'`), ou anexação (`'a'`).

```
# Código
# Abrir um arquivo para leitura
arquivo = open('meuarquivo.txt', 'r')

# Abrir um arquivo para escrita (cria o arquivo se não existir)
arquivo = open('meuarquivo.txt', 'w')

# Abrir um arquivo para anexação (adiciona ao final do arquivo)
arquivo = open('meuarquivo.txt', 'a')
```

Lendo Arquivos

Você pode ler o conteúdo de um arquivo usando métodos como `read()`, `readline()`, ou `readlines()`.

```
# Código
# Lendo todo o conteúdo do arquivo
arquivo = open('meuarquivo.txt', 'r')
conteudo = arquivo.read()
print(conteudo)

# Lendo linha por linha
arquivo = open('meuarquivo.txt', 'r')
linha = arquivo.readline()
while linha:
    print(linha, end='')
    linha = arquivo.readline()

# Lendo todas as linhas em uma lista
arquivo = open('meuarquivo.txt', 'r')
linhas = arquivo.readlines()
print(linhas)
```

Escrevendo em Arquivos

Para escrever em um arquivo, você usa o método `write()`. Se o arquivo não existir, ele será criado.

```
# Código
# Escrevendo uma única string no arquivo
arquivo = open('meuarquivo.txt', 'w')
arquivo.write('Olá, Mundo!')

# Escrevendo várias linhas no arquivo
arquivo = open('meuarquivo.txt', 'w')
linhas = ['Linha 1\n', 'Linha 2\n', 'Linha 3\n']
arquivo.writelines(linhas)
```

Fechando Arquivos

Após concluir as operações com um arquivo, é importante fechá-lo usando o método `close()` para liberar os recursos do sistema.

```
# Código
arquivo = open('meuarquivo.txt', 'r')
# Operações com o arquivo
arquivo.close()
```

Gerenciador de Contexto

Usar o gerenciador de contexto `with` garante que o arquivo seja fechado automaticamente após a execução do bloco de código.

```
# Código
with open('meuarquivo.txt', 'r') as arquivo:
    conteudo = arquivo.read()
    print(conteudo)
# O arquivo é fechado automaticamente ao sair do bloco
```

Tratamento de Exceções

Você pode usar a declaração `try...except` para lidar com exceções que podem ocorrer durante a manipulação de arquivos, como quando o arquivo não é encontrado.

```
# Código
try:
    arquivo = open('meuarquivo.txt', 'r')
    conteudo = arquivo.read()
except FileNotFoundError:
    print('Arquivo não encontrado!')
except IOError:
    print('Erro de entrada/saída!')
finally:
    if 'arquivo' in locals():
        arquivo.close()
```