Aula 12 - Arquivos

Técnicas de Programação

Prof. Me. Gustavo Torres Custodio
gustavo.custodio@anhembi.br

Aula 12 - Arquivos

Arquivos

Arquivos

- É necessário uma forma de armazenar dados permanentemente.
- Arquivos são uma maneira eficiente de se comunicar com um programa.
  - São melhores do que o `input()` em muitos casos.
- Podem ser usados para armazenar configurações de programas.

Arquivos

- Um arquivo é uma região do disco onde informações são lidas e gravadas.
  - Essa área é gerenciada pelo próprio sistema operacional.
- Para acessar um arquivo é necessário abri-lo primeiro.
  - Um arquivo é aberto usando seu nome e o diretório (pasta) onde é armazenado.

Abrindo um Arquivo

- Um arquivo é aberto usando o comando open().

```
arquivo=open('teste.txt','r')
print(arquivo.read())
arquivo.close()
```

Modos de Abertura de um Arquivo

Modo	Operações
r	leitura
w	escrita, apaga conteúdo anterior
a	escrita, mas preserva conteúdo anterior
b	modo binário
+	atualização (leitura e escrita)

Escrevendo em um Arquivo

- Um arquivo é aberto para escrita utilizando a operação **w**.

```
arquivo=open('numeros.txt','w')

for linha in range(1,101):
    arquivo.write('%d\n' % linha)
arquivo.close()
```

- Se ao invés de **w**, utilizarmos a operação **a**, é possível escrever no arquivo sem substituir o conteúdo original.
- A função `close()` fecha o arquivo, impedindo o vazamento de memória.

Lendo de um Arquivo

- Um arquivo é aberto para leitura utilizando a operação **r**.

```
arquivo=open('numeros.txt','r')

for linha in arquivo.readlines():
    print(linha)
arquivo.close()
```

- A função `readlines()` lê todas as linhas presentes no arquivo, retornado-as em uma lista onde cada elemento é uma linha.

Modo Binário

- Arquivos binários são formatados sem caracteres legíveis, como arquivos de som, imagem e PDFs.

```
arquivo=open('aula12_handout.pdf','rb')
# rb significa leitura em forma de bytes

byte = arquivo.read(1)
while byte:
    print(byte)
    byte = arquivo.read(1)
arquivo.close()
```

- No exemplo acima, o programa lê cada byte (conjunto de 8 bits) de um PDF individualmente.

Combinando Múltiplos Modos de Abertura

- Adicionar o caractere ‘+’ combina múltiplos modos de acesso.

```
# Podemos ler do arquivo ou escrever nele nesse modo
arquivo = open('numeros.txt','r+')

arquivo.write('Lendo e escrevendo\n')
arquivo.close()
```

- No exemplo, abrimos o arquivo tanto para escrita quanto para leitura.
  - No entanto, o ponteiro está no começo do arquivo, então o conteúdo “Lendo e escrevendo” substituirá parte do arquivo.
  - Se lermos o conteúdo do arquivo antes de escrever, o ponteiro vai para o final.

Instrução `with`

- Uma segunda forma de abrir arquivos é utilizando a instrução `with`.
- Ela é utilizada para uma variedade de situações e deixa o código muito mais limpo e legível.
  - Em arquivos, ela é responsável por aquisição e liberação de recursos.
  - Dessa forma, a função `close()` não é necessária.

Exemplo `with`

```
with open('numeros.txt', 'r') as arquivo:
    print(arquivo.read())
```

- No código acima, o arquivo é aberto, seu conteúdo é impresso e o arquivo é fechado.
  - Tudo em apenas duas linhas.

Aula 12 - Arquivos

Arquivos JSON

JSON

- JSON (*JavaScript Object Notation*) é uma forma comum de estruturar dados.
- Um texto em formato JSON é extremamente fácil de se trabalhar e é frequentemente utilizado para comunicação entre diferentes aplicações.
  - Exemplo: Receber informações de uma API.

JSON

- A linguagem Python possui suporte nativo para JSON.
  - Dessa forma, é fácil adaptar o conteúdo de um JSON para as estruturas de dados do Python.
  - Listas são delimitadas por [].
  - Dicionários são delimitados por {}.

Estrutura JSON

```
[
    {
        'nome': 'João',
        'idade': 30
    },
    {
        'nome': 'Vivian',
        'idade': 21
    }
]
```

Estrutura JSON

- No exemplo anterior, temos um JSON formado por dois dicionários dentro de uma lista.
- Para carregar o conteúdo de um JSON é necessário importar o módulo `json` do Python.
  - `import json`
- O conteúdo do arquivo é lido e depois é convertido para o formato JSON.

Exemplo JSON

- Crie um programa que leia a informação de 3 pessoas (nome, idade, salário) e armazene em um arquivo JSON.
  - As informações de cada pessoa são armazenadas em um dicionário.
  - Os dicionários são armazenados em uma lista.

Exemplo JSON

```
import json

pessoas = []
for i in range(3):
    nome = input(f'Digite o nome da pessoa {i+1}: ')
    idade = int(input(f'Digite a idade da pessoa {i+1}: '))
    salario = float(input(f'Digite o salário da pessoa {i+1}: '))
    pessoa = {'nome': nome, 'idade': idade, 'salario': salario}
    pessoas.append(pessoa)

# Salvando arquivo
with open('nomes.json', 'w') as arquivo:
    json.dump(pessoas, arquivo, indent=4)

# Lendo de arquivo
with open('nomes.json', 'r') as arquivo:
    conteudo = json.load(arquivo)
    print(conteudo)
```

Aula 12 - Arquivos

API

API

- API é um conjunto de definições e protocolos usado no desenvolvimento e na integração de software de aplicações.
- API é um acrônimo em inglês que significa interface de programação de aplicações.
- Elas facilitam a comunicação com produtos e serviços de terceiros.

API

- Normalmente o conteúdo de uma API é acessado por meio de uma URL.
- Em alguns casos, é necessário ter uma chave de aplicação para acessar a API.
- Exemplo: API do mapbox.

Exemplo API

- Vamos criar uma conta no site do mapbox (https://www.mapbox.com/).
  - Grave o token de acesso.
  - Vamos colocar o token de acesso em um arquivo “accesstoken.txt”.
- Criaremos um programa que usa a API do mapbox para encontrar as coordenadas de um ponto específico.

Exemplo API

- Antes de tudo, vamos instalar o módulo requests do python:
  - Esse módulo serve para fazer requisições online.
  - Dentro do terminal do VSCode, use o comando:
    - `python -m pip install requests`.

Exemplo API

```
import urllib.parse
import requests
import json

def ler_token() -> str:
    with open('accesstoken.txt', 'r') as arqtoken:
        token_acesso = arqtoken.read()
        # Remove o pulo de linha no fim do arquivo
        return token_acesso.strip('\n')
```

Exemplo API

```
def buscar_json(endereco_url: str, token_acesso: str) -> dict:
    # Url que fará a requisição para a API.
    url_api = f'''
        https://api.mapbox.com/geocoding/v5/mapbox.places/
        {endereco_url}.json?access_token={token_acesso}'''
    conteudo = requests.get(url_api).text
    # Json contendo os endereços que batem com a busca
    json_enderecos = json.loads(conteudo)
    return json_enderecos
```

Exemplo API

```
def devolver_coordenadas(json_enderecos: dict) -> 'tuple[float, float]':
    primeiro_endereco = json_enderecos['features'][0]
    longitude = primeiro_endereco['geometry']['coordinates'][0]
    latitude = primeiro_endereco['geometry']['coordinates'][1]
    # Devolve as coordenadas do endereço em forma de tupla
    return (latitude, longitude)
```

Exemplo API

```
endereco = "Rua Casa do Ator, 275, Vila Olímpia, São Paulo"
# Converte o endereço para o formato de url
endereco_url = urllib.parse.quote_plus(endereco)

token_acesso = ler_token()
json_enderecos = buscar_json(endereco_url, token_acesso)
print(devolver_coordenadas(json_enderecos))
```

Exemplo API

- O programa se comunica com a API do mapbox por meio do módulo `requests`.
- A requisição feita por nosso programa é validada por meio da chave de acesso.
  - Se tudo estiver certo, a API nos devolve as informações para o endereço buscado.

Aula 12 - Arquivos

Exercícios

Exercício 1

- Crie um programa que verifica se existe um arquivo `ultimoacesso.json`.
  - O arquivo contém dois campos:
    - **nome**: contendo o nome da pessoa.
    - **ultimo_acesso**: contendo a hora e data do último acesso.
- Se o arquivo existir mostre na tela:
  - “Bem-vindo <nome>, você entrou pela última vez em <data_hora>”.
  - e atualize a hora do último acesso.
- Se o arquivo não existir, pergunte o nome do usuário e crie o arquivo.

Exercício 2

- Faça um programa que leia um arquivo contendo o nome e o preço de diversos produtos (separados por linha), e calcule o total da compra.

Exercício 3

- Modifique o exemplo do mapbox reaizado anteriormente.
  - Faça com que o programa peça para o usuário digitar 3 localizações diferentes.
  - Em seguida, encontre as coordenadas de cada localização.
  - Mostre para o usuário a localização mais próxima.

Exercício 4

- Faça um programa no qual o usuario informa o nome do arquivo e uma palavra.
  - Retorne o número de vezes que aquela palavra aparece no arquivo.

Exercício 5

- Veja como a API `http://deckofcardsapi.com/` funciona.
- Faça um programa que:
  - Crie um baralho.
  - Compre 3 cartas desse baralho.

Referências

- Menezes, Nilo Ney Coutinho: *Introdução à programação com Python*, capítulo 9.
  Novatec Editora, 2ª edição, 2010.
- *With Statement in Python.*
  `https://www.geeksforgeeks.org/with-statement-in-python/`.
  [Acessado 28-10-2021].
- *APIs.*
  `https://www.redhat.com/pt-br/topics/api/what-are-application-programming-interfaces`.
  [Acessado 28-10-2021].

Obrigado

gustavo.custodio@anhembi.br