Arquivos

Programação 1

Dr. Vinicius Hartmann Ferreira

Sobre arquivos

- Arquivos, sobretudo arquivos de texto, são utilizados como forma de manter informações de uma aplicação.

- Por mais que seja possível utilizar bancos de dados, os arquivos de texto ainda são uma alternativa rápida e fácil de armazenar informações de uma aplicação.

- Em Python não é necessário utilizar recursos externos para manipular arquivos.

Utilizando um arquivo

```
#Usando um arquivo
arquivo = open("arquivo.txt", 'a')
```

- **A função open precisa basicamente de 2 parâmetros:**
  - **file:** o nome do arquivo que será utilizado;
  - **mode:** de que forma ele será utilizado:
    - **r:** quando o arquivo será utilizado apenas para leitura;
    - **w:** quando o arquivo será utilizado apenas para escrita;
    - **a:** quando será adicionada informação ao final do arquivo; e
    - **r+:** quando pretende-se trabalhar com escrita e leitura.

Inserindo informação em um arquivo

```
#Com o arquivo
with open("arquivo.txt", 'w') as arquivo:
    nome = input("Informe um nome:")
    #Escreve uma linha no arquivo
    arquivo.write(nome)
```

```
#Com o arquivo
with open("arquivo.txt", 'a') as arquivo:
    nome = input("Informe um nome:")
    #Escreve uma linha no arquivo
    arquivo.write(nome)
```

Inserindo informação em um arquivo

```
#Neste arquivo especificamos a codificação dos caracteres
with open("arquivo.txt", 'w', encoding='utf-8') as arquivo:
    times = ['Grêmio','Juventude','Caxias','Internacional']
    for time in times:
        arquivo.write(time)
```

- A codificação utf-8 abrange os caracteres da língua portuguesa.

Inserindo informação em um arquivo

```
#Podemos escrever uma lista direto no arquivo
with open("arquivo.txt", 'w', encoding='utf-8') as arquivo:
    times = ['Grêmio','Juventude','Caxias','Internacional']
    arquivo.writelines(times)
```

- Podemos utilizar o método **writelines** para inserir uma lista diretamente no arquivo texto.

Inserindo informação em um arquivo

```
#Neste arquivo especificamos a codificação dos caracteres
with open("arquivo.txt", 'w', encoding='utf-8') as arquivo:
    times = ['Grêmio','Juventude','Caxias','Internacional']
    for time in times:
        arquivo.write(time+'\n')
```

- Para separar as linhas é necessário incluir a quebra de linha (‘\n’)

Lendo informações de um arquivo

```
with open('arquivo.txt','r', encoding='utf-8') as arquivo:
    #Lê todo conteúdo como uma string
    conteudoTodo = arquivo.read()
    #Lê o arquivo de linha em linha
    umaLinha = arquivo.readline()
    #Lê todas as linhas de um arquivo
    todasLinhas = arquivo.readlines()
```

Lendo informações de um arquivo

```
with open('arquivo2.txt','r',encoding='utf-8')as arquivo:
    #Aqui foram lidas todas as linhas, criando uma lista
    conteudo = arquivo.readlines()
    #Para cada linha da lista
    for linha in conteudo:
    #Imprime o conteúdo sem a quebra de linha
        print(linha.strip('\n'))
```

- Neste caso estamos imprimindo cada uma das linhas do conteúdo do arquivo2.txt.

Lendo informações de um arquivo

```
with open('arquivo2.txt','r',encoding='utf-8')as arquivo:
    conteudo = arquivo.readlines()
    for linha in conteudo:
        #Aqui quebramos a linha pelo '#'
        linguagens = linha.strip().split("#")
        print(linguagens[0])
        print(linguagens[1])
```

- Para separarmos as informações devemos usar a padronização e função **split**.

Exemplo de uma lista de times

```
with open('arquivo.txt','a',encoding='utf-8') as arquivo:
    while True:
        time=input("Nome de um time:")
        if time=='nenhum':
            break
        arquivo.write(time+"\n")
```