PÓS-GRADUAÇÃO EM ENGENHARIA DE DADOS

IEC PUC MINAS

Disciplina:

PYTHON

Professor: Nelson Júnior

Arquivos

Como é fácil manipular um arquivo!

Arquivos

- Apenas uma linha para abrir um arquivo!
  - file = open("data", 'r') tipos: r, a, w
- Alguns métodos para operações em arquivos:
  - file.read(), readline(), readlines(),
  - file.write(), writelines(),
  - file.close()

Arquivos Textos

- Até agora, nós vimos no curso exemplos de programas que obtiveram os dados de entrada de usuários via teclado.

```
nomes = input()
```

- A maioria desses programas pode receber seus dados de entrada de **arquivos texto** também.
- Um arquivo texto armazena caracteres que podem ser mostrados diretamente na tela ou modificados por um editor de textos simples. Exemplos: código python, documento texto simples, páginas HTML.

Arquivos Textos

- Quando comparado à entrada de dados via teclado, as principais vantagens de se obter dados de entrada de um arquivo são:
  - O conjunto de dados pode ser muito maior.
  - Os dados podem ser inseridos muito mais rapidamente e com menos chance de erro.
  - Os dados podem ser usados repetidamente com o mesmo programa ou com diferentes programas.

Arquivos Textos

- Um **nome** e **caminho** únicos são usados por usuários ou em programas ou scripts para acessar um arquivo texto para fins de leitura e modificação.
- As tarefas básicas envolvidas na manipulação de arquivos são **ler** dados de arquivos e **escrever** ou anexar dados em arquivos.
- Leitura e escrita em arquivos em Python são muito fáceis de gerenciar.

Arquivos Textos

- Para se trabalhar com arquivos devemos abri-lo e associá-lo com uma variável.
- A variável será um objeto do tipo `file` que contém métodos para ler e escrever no arquivo.
- O primeiro passo então é abrir o arquivo com o comando `open`:

```
variavel_arquivo = open("nome do arquivo", "modo")
```

Arquivos Textos

- O primeiro passo então é abrir o arquivo com o comando `open`:

```
variavel_arquivo = open("nome do arquivo", "modo")
```

- O `"nome do arquivo"` pode ser relativo ou absoluto.
- O `"modo"` pode ser `"r"`(leitura), `"r+"`(leitura e escrita), `"w"`(escrita), `"a"`(append).

Abrindo um Arquivo Texto para Leitura

- Ao se trabalhar com arquivos é bom colocar a abertura do arquivo no bloco `try`, e o tratamento da exceção no bloco `except`.

```
try:
    arquivo = open("tarefas.txt", "r")
    print("Abri arquivo com sucesso.")
except:
    print("Não foi possível abrir o arquivo.")
```

Lendo Dados de um Arquivo Texto

- Para ler dados do arquivo aberto, usamos o método `read`.
  - `read(num bytes)`: Retorna uma string contendo os próximos `num bytes` do arquivo.
  - `read()`: Sem parâmetro é retornado uma string contendo **todo** o arquivo!

```
try:
    arquivo = open("tarefas.txt", "r")
    conteudo = arquivo.read()
except:
    print("Não foi possível abrir o arquivo.")
```

Lendo Dados de um Arquivo Texto

- O programa pode ser alterado para ler todo o arquivo de uma vez.
  - Lembre-se que se o arquivo for muito grande isto pode acarretar em uma sobrecarga da memória do seu computador fazendo com que este fique lento ou mesmo trave.

```
try:
    arquivo = open("tarefas.txt", "r")
    s = arquivo.read()
    print(s, end="")
    arquivo.close()
except:
    print("Não foi possível abrir o arquivo.")
```

```
try:
    arquivo = open("tarefas.txt", "r")
    while True:
        s = arquivo.read(1)
        print(s, end="")
        if (s == ""):
            break
    arquivo.close()
except:
    print("Não foi possível abrir o arquivo." )
```

```
try:
    arquivo = open("tarefas.txt", "r")
    s = arquivo.read()
    print(s, end="")
    arquivo.close()
except:
    print("Não foi possível abrir o arquivo." )
```

Lendo Dados de um Arquivo Texto

- Uma maneira mais eficiente do que se ler um byte por vez e menos arriscada do que se ler todo o arquivo de uma única vez, é ler uma linha por vez.
- Para isso usamos o método `readline()` que devolve uma linha do arquivo em formato string.

Lendo Dados de um Arquivo Texto

```
try:
    arquivo = open("tarefas.txt", "r")
    while True:
        s = arquivo.readline()
        print(s, end="")
        if (s == ""):
            break
    arquivo.close()
except:
    print("Não foi possível abrir o arquivo.")
```

Lendo Dados de um Arquivo Texto

- Notem que ao realizar a leitura de um caractere, ou uma linha, automaticamente o indicador de posição do arquivo se move para o próximo caractere (ou linha).
- Ao chegar no fim do arquivo o método `read(readline())` retorna a string vazia.
- Para voltar ao início do arquivo novamente você pode fechá-lo e abri-lo mais uma vez, ou usar o método `seek`.

Lendo Dados de um Arquivo Texto

- `seek(offset, from_what)`: o primeiro parâmetro indica quantos bytes se move a partir do valor inicial `from_what`.
- Os valores de `from_what` podem ser:
  - 0: indica início do arquivo.
  - 1: indica a posição atual no arquivo.
  - 2: indica a posição final do arquivo.
- O programa a seguir imprime duas vezes o conteúdo do arquivo `"tarefas.txt"`.

```
try:
    arquivo = open("tarefas.txt", "r")
    while True:
        s = arquivo.readline()
        print(s, end="")
        if (s == ""):
            break
    arquivo.seek(0,0) #mover indicador de posição
              #0 bytes a partir do início
    while True:
        s = arquivo.readline()
        print(s, end="")
        if (s == ""):
            break
    arquivo.close()
except:
    print("Não foi possível abrir o arquivo.")
```

EXERCÍCIOS

Escrevendo Dados de um Arquivo Texto

- Para escrever em um arquivo, ele deve ser aberto de forma apropriada usando o modo `"w"`, `"a"` ou `"r+"`.
- `arquivo = open("nome do arquivo", "modo")`
  - `"w"`: se o arquivo existir ele será sobreescrito, ou seja todo o conteúdo anterior será apagado.
  - `"a"`: o indicador de posição ficará no fim do arquivo, e dados escritos serão adicionados no fim do arquivo.
  - `"r+"`: o indicador de posição ficará no início do arquivo, e dados serão escritos sobre dados anteriores.

Escrevendo Dados de um Arquivo Texto

- **Sobreescreve** o início do arquivo `"tarefas.txt"`:

```
try:
    arquivo = open("tarefas.txt", "r+")
    arquivo.write("Alterei o começo do arquivo\n")
    arquivo.close()
except:
    print("Erro no arquivo.")
```

Escrevendo Dados de um Arquivo Texto

- **Sobreescreve** o início do arquivo `"tarefas.txt"`:

```
try:
    arquivo = open("tarefas.txt", "r+")
    arquivo.write("Alterei o começo do arquivo\n")
    arquivo.close()
except:
    print("Erro no arquivo.")
```

PUC Minas

EXERCÍCIOS

PUC Minas

Resumindo .... open

```
arquivo = open("nome do arquivo", "modo")
```

modo	operador	indicador de posição
r	leitura	início do arquivo
r+	leitura e escrita	início do arquivo
w	escrita	início do arquivo
a	(append) escrita	final do arquivo

Resumindo .... `open`

- Se um arquivo for aberto para leitura (`r`) e ele não existir, `open` gera um erro.
- Se um arquivo for para escrita (`w`) e existir, ele é sobrescrito. Se o arquivo não existir, um novo arquivo é criado.
- Se um arquivo for aberto para leitura/escrita (`r+`) e existir, ele não é apagado. Se o arquivo não existir, `open` gera um erro.

Alterando um Texto

- Podemos ler todo o texto de um arquivo e fazer qualquer alteração que julgarmos necessária.
- O texto alterado pode então ser sobrescrito sobre o texto anterior.
- Como exemplo vamos fazer um programa para alterar um texto substituindo toda ocorrência da letra ’a’ por ’A’.
- Como uma string é imutável primeiro transformaremos esta em lista, alteramos o que precisar, depois transformamos a lista em string novamente para então escrever em arquivo.

Alterando um Texto

- Transformando strings em listas e vice-versa.

```
string = "abc"
string = list(string)
string
```

```
['a', 'b', 'c']
```

```
string = "".join(string)
string
```

```
'abc'
```

Alterando um Texto

- Programa que altera arquivo texto trocando ocorrências de 'a' por 'A'.

```
try:
    arquivo = open("tarefas.txt", "r+")
    t = arquivo.read()
    t = list(t) #transformamos em lista
    for i in range(len(t)):
        if(t[i] == 'a'):
            t[i] = 'A'
    arquivo.seek(0,0)
    t = "".join(t)
    arquivo.write(t)
    arquivo.close()
except:
    print("Erro no arquivo.")
```

EXERCÍCIOS

PUC Minas

PUC Minas

PUC Minas

ESBOÇO DA APRESENTAÇÃO

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letterset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum. Lorem Ipsum is simply dummy text of the printing and typesetting industry.

PUC Minas

MODELO DE TEXTO

MODELO DE TEXTO SUBTÍTULO

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

PUC Minas

MODELO DE TEXTO

REDESENHANDO A EXPERIÊNCIA

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type

SIMPLIFICANDO PLATAFORMAS

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type

MAXIMIZANDO A TECNOLOGIA

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type

PUC Minas

MODELO DE TEXTO

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type