



# Arquivos e Dicionários

[fmasanori@gmail.com](mailto:fmasanori@gmail.com)

# Arquivos

- Até agora nossos dados desapareciam ao sair do IDLE
- Arquivos servem para armazenamento permanente
- Um arquivo é uma área em disco onde podemos ler ou gravar informações
- Acessamos o arquivo pelo seu nome
- Para acessar um arquivo é preciso abri-lo

# Arquivos

- Ao abrir o arquivo informamos seu nome, diretório onde fica (se necessário) e que operações iremos executar: leitura e/ou escrita
- A função que abre os arquivo é `open` e os modos são: `r` – leitura, `w` – escrita, `a` – append, `b` – binário, `+` (atualização)
- Os métodos para ler ou escrever são `read` e `write`
- Os arquivos devem ser fechados com `close`

# Arquivos

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

- Caso você execute este programa nada aparecerá na tela
- Procure no diretório c:\Python3x o arquivo números.txt
- O modo w cria o arquivo se ele não existir, caso exista ele será apagado e reescrito

# Arquivos

```
arquivo = open('números.txt', 'r')
for linha in arquivo.readlines():
    print(linha)
arquivo.close()
```

- `readlines` gera uma lista onde cada elemento é uma linha lida
- Arquivos textos são simples e possuem um caracter de controle no final para pular linha
- Se quisermos tirar esse caracter do final podemos usar `print(linha.rstrip())`

# Pythonic way

```
with open('números.txt') as f:  
    print (f.read())
```

- O código acima faz o mesmo da forma pythônica
- No slide anterior vimos como programadores normais fazem a leitura
- Python é legal, pois sempre você pode se aprofundar mais
- Python é simples, mas difícil de esgotar 😊



# Cripto

- Leia mensagem.txt e grave cripto.txt com todas as vogais trocadas por '\*'

```
texto = open('mensagem.txt')
saida = open('cripto.txt', 'w')
for linha in texto.readlines():
    for letra in linha:
        if letra in 'aeiou':
            saida.write('*')
        else:
            saida.write(letra)
texto.close()
saida.close()
```

# Validate IP address

## **IPS.txt**

200.135.80.9

192.168.1.1

8.35.67.74

257.32.4.5

85.345.1.2

1.2.3.4

9.8.284.5

192.168.0.256

## **Válidos.txt**

200.135.80.9

192.168.1.1

8.35.67.74

1.2.3.4

## **Inválidos.txt**

257.32.4.5

85.345.1.2

9.8.284.5

192.168.0.256



# Validate IP address

```
def ip_ok(ip):
    ip = ip.split('.')
    for byte in ip:
        if int(byte) > 255:
            return False
    return True


arq = open('IPS.txt')
validos = open('Válidos.txt', 'w')
invalidos = open('Inválidos.txt', 'w')
for linha in arq.readlines():
    if ip_ok(linha):
        validos.write(linha)
    else:
        invalidos.write(linha)
arq.close()
validos.close()
invalidos.close()
```

# HTML

- Páginas web são escritas em HTML (Hypertext Mark-up Language)
- Tags HTML começam com `<` e terminam com `>`
- A página web é escrita entre `<html>` e `</html>` que é a tag de maior nível
- Normalmente inserimos código javascript
- Javascript não é um subconjunto de Java

# HTML

```
arquivo = open('ola.html', 'w', encoding='utf-8')
arquivo.write(''<!DOCTYPE html>
<html lang="pt-BR">
<head>
<meta charset="utf-8">
<title>Título da Página</title>
</head>
<body>
Olá!
</body>
</html>'')
arquivo.close()
```



Note o parâmetro de codificação utf-8  
Sem ele os acentos não sairão

# Dictionaries

- O dicionário em si consiste em relacionar uma chave a um valor específico
- Diferentemente das listas, onde o índice é um número, dicionários utilizam suas chaves como índice
- Para adicionar novos elementos não preciso de append, basta fazer a atribuição
  - Se a chave já existe: o valor associado é alterado
  - Se a chave não existe: a nova chave é adicionada

# Dictionaries

```
>>> d = {}
>>> d['a'] = 'alpha'
>>> d['o'] = 'omega'
>>> d['g'] = 'gama'
>>> d
{'a': 'alpha', 'g': 'gama', 'o': 'omega'}
>>> d['a']
'alpha'
>>> d['x']
Traceback (most recent call last):
  File "<pyshell#6>", line 1, in <module>
    d['x']
KeyError: 'x'
```

# Dictionaries

```
>>> d.keys()
dict_keys(['a', 'g', 'o'])
>>> d.values()
dict_values(['alpha', 'gama', 'omega'])
>>> 'g' in d
True
>>> 'x' in d
False
>>> for chave in d: print (chave)

a
g
o
```

# Dictionaries

- Faça um programa que leia o arquivo alice.txt e conte o número de ocorrências de cada palavra no texto. Obs.: para saber os caracteres especiais use `import string` e utilize `string.punctuation`
- <http://www.gutenberg.org/cache/epub/11/pg11.txt>

# Dictionaries

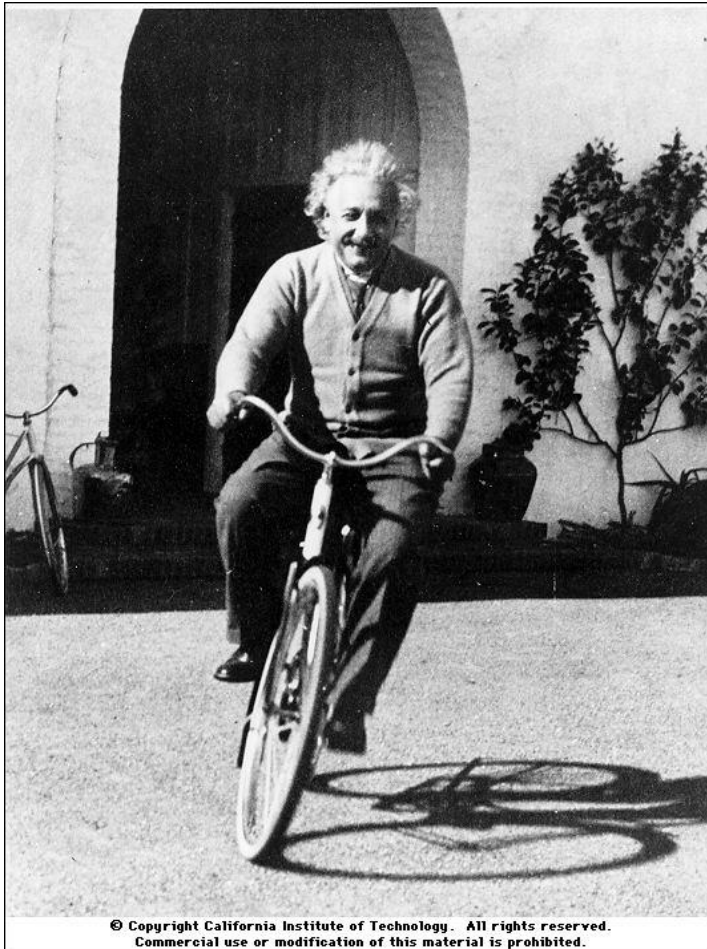
```
arq = open('alice.txt')
texto = arq.read()
texto = texto.lower()
import string
for c in string.punctuation:
    texto = texto.replace(c, ' ')
texto = texto.split()

dic = {}
for p in texto:
    if p not in dic:
        dic[p] = 1
    else:
        dic[p] += 1
print ('Alice aparece %s vezes' %dic['alice'])
arq.close()
```

<https://gist.github.com/4673017>



# Exercício Programa 1



*“A vida é como  
andar de bicicleta.  
Para manter o  
equilíbrio, é preciso  
se manter em  
movimento”.*  
*Einstein.*