

Programação II

Ficheiros

Hugo Pacheco

DCC/FCUP

21/22

Dados

- Até agora, os dados utilizados durante a execução de um programa (input ou gerados) são armazenados em memória volátil, apagada quando o programa termina
- Ficheiros externos (imagens, música, texto, ...):
 - Dados armazenados em suportes persistentes (HDD, pen USB, DVD, ...)
 - **Texto**: dados são caracteres (podem ser visualizados com um editor de texto)
 - **Binário**: dados são armazenados mais eficientemente como 0s e 1s (não vamos lidar diretamente com isso)

Organização de ficheiros

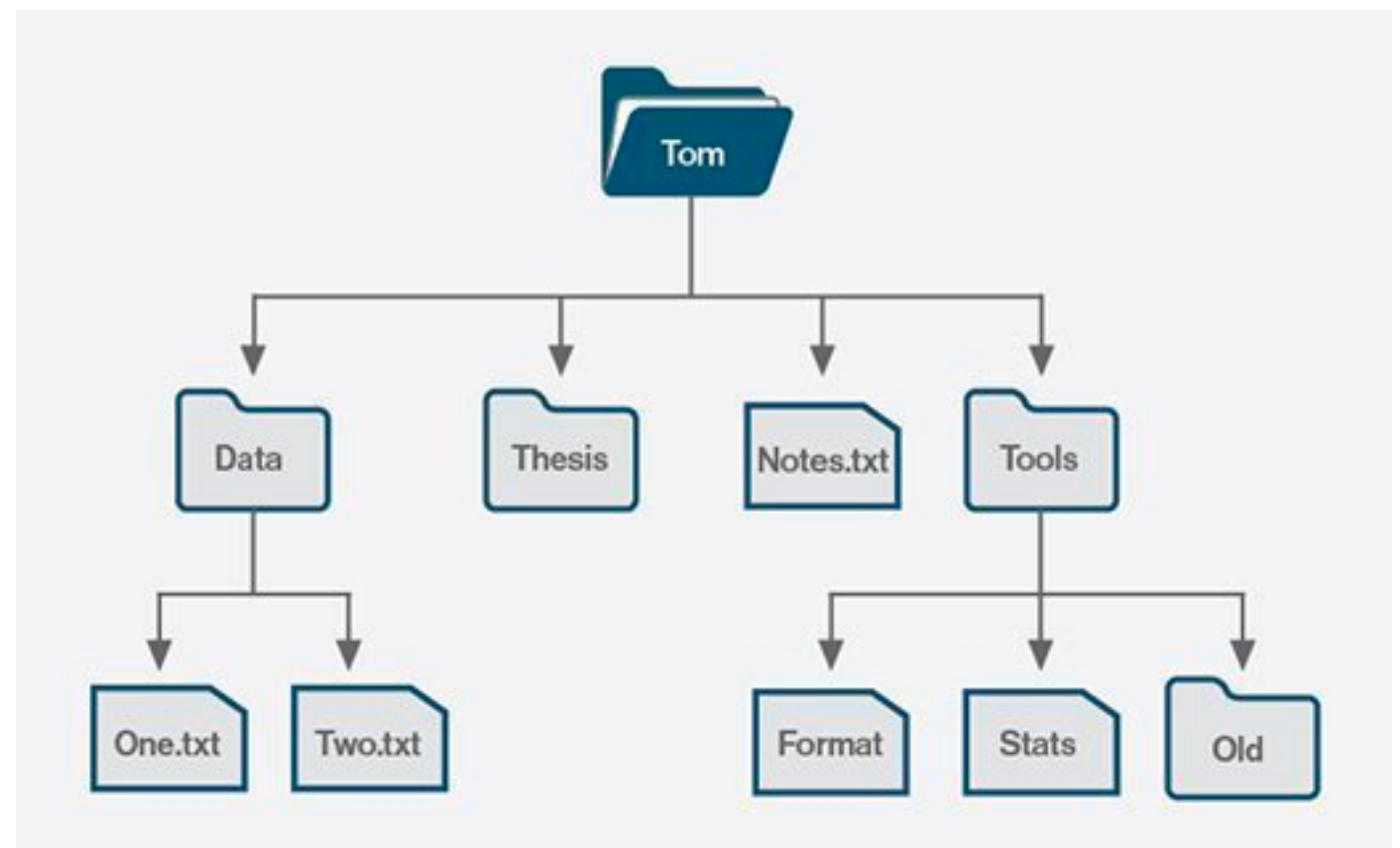
- Identificados por nomes
- Organizados hierarquicamente em pastas

- Linux/Mac

/Tom/Data/One.txt

- Windows

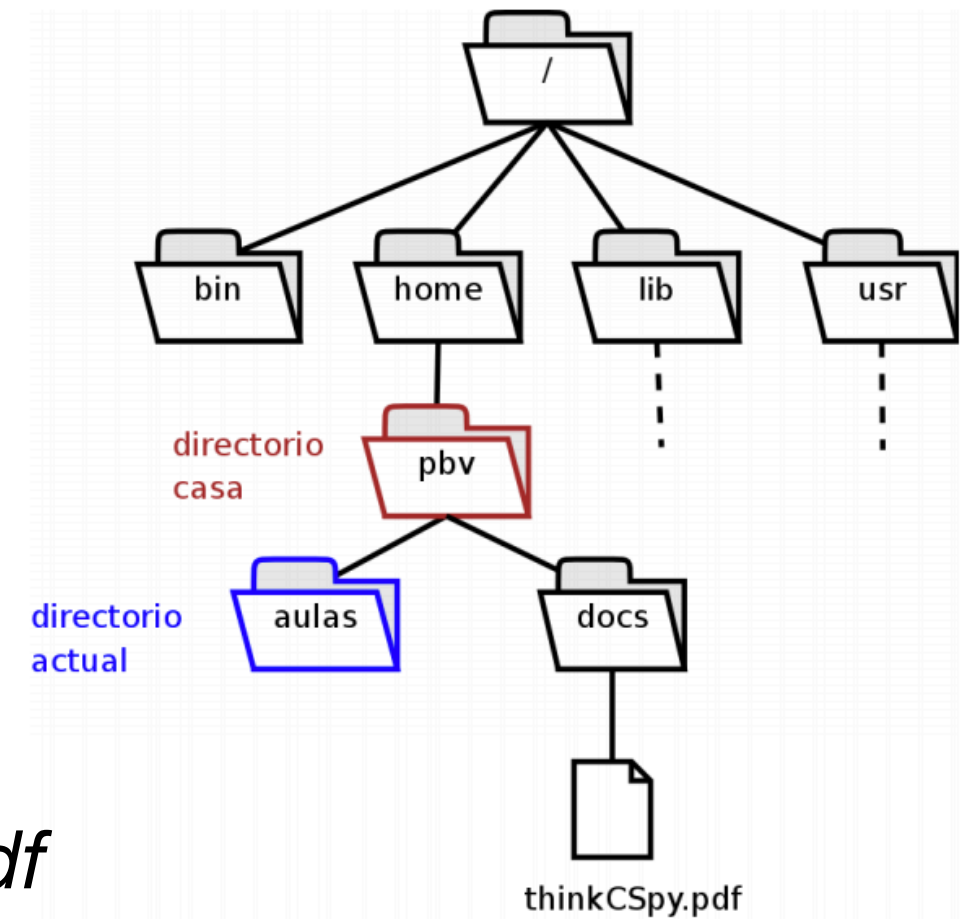
C:\\Tom\\Data\\One.txt



Caminhos no sistema de ficheiros

.	Pasta atual
..	Pasta mãe
~	Pasta home

- Caminhos:
 - */home/pbv/aulas/docs/thinkCSpy.pdf* (absoluto)
 - *../docs/thinkCSpy.pdf* (relativo)
 - *~/docs/thinkCSpy.pdf* (relativo à home)



Navegar no sistema de ficheiros

<i>Alguns comandos UNIX</i>	
<i>ls</i>	Lista ficheiros na pasta atual
<i>pwd</i>	Imprime o caminho atual
<i>cd</i>	Muda a pasta atual
<i>mkdir</i>	Cria uma nova pasta
<i>rmdir</i>	Remove uma pasta vazia
<i>cp</i>	Copia ficheiros
<i>mv</i>	move/renomeia ficheiros
<i>rm</i>	Remove ficheiros
<i>man <comando></i>	Mostra manual de um <comando>

Manipular ficheiros

- Funcionamento como um bloco de notas:
 - abrir/fechar
 - modo leitura/escrita
 - localização do ficheiro:
 - relativa à pasta onde o ficheiro do programa se encontra
 - absoluta (relativa à raiz do sistema de ficheiros)

Abrir/fechar ficheiros

- Abrir um ficheiro cria um *file handle* que permite ler/escrever no ficheiro
- Tem que se fechar o *file handle* no fim
- Padrão (abre o ficheiro e fecha automaticamente no fim):

```
with open(ficheiro, modo) as f:  
    f.operacao  
    ...
```

- Dica: ficheiro =
 - apenas nome (e.g., '*lusiadas.txt*') se na mesma pasta que o programa
 - caminho absoluto/relativo (e.g., '*/home/progii/lusiadas.txt*' ou '*../lusiadas.txt*') se numa pasta diferente

Ler de ficheiros

- É possível controlar como e quantos caracteres se lêem de um ficheiro

```
with open('test.txt', 'r') as f:
    f.read() # lê até ao fim do ficheiro
with open('test.txt', 'r') as f:
    f.readline() # lê uma linha
with open('test.txt', 'r') as f:
    f.readlines() # lê todo como uma lista de linhas
with open('test.txt', 'r') as f:
    f.read(5) # lê 5 caracteres
with open('test.txt', 'r') as f:
    digit = int(f.read(1)) # lê dígito num linhas
    for _ in range(digit):
        print(f.readline()) # lê várias linhas
```


Escrever em ficheiros

- Modo *write* cria novo ficheiro e sobrepõe se existente
- Modo *append* acrescenta conteúdo no fim do ficheiro

```
# cria/apaga ficheiro e escreve string
with open('test.txt', 'w') as f:
    f.write('Hello\n')
```

```
# acrescenta duas strings no fim do ficheiro
with open('test.txt', 'a') as f:
    f.write('Big\n')
    f.write('World\n')
```

```
# acrescenta lista de strings no fim
with open('test.txt', 'a') as f:
    f.writelines(['Really\n', 'Big\n'])
```

Ler e escrever ficheiros

- E.g., ler um ficheiro de um programa python e copiar todas as linhas não comentário para outro ficheiro

```
# abrir 2 ficheiros de uma vez
with open('test.py', "r") as fin\
    , open('test2.py', "w") as fout:
    for line in fin: # readline() implícito
        if not line.startswith('#'):
            fout.write(line)
```

Download de ficheiros web

- Faz download de *Os Lusíadas* de um endereço web e guarda num ficheiro '*lusiadas.txt*' na pasta atual
- Lê, guarda numa lista de linhas e imprime no ecrã

```
import urllib.request

url = 'http://www.gutenberg.org/cache/epub/3333/pg3333.txt'
urllib.request.urlretrieve(url, 'lusiadas.txt')

with open('lusiadas.txt', 'r') as f:
    lines = f.readlines()
for line in lines:
    print(line)
```

Exemplo Os *Lusíadas*

- Extrai estrofes de Os *Lusíadas*

```
# lê ficheiro como lista de linhas sem \n
with open('lusiadas.txt', 'r') as f:
    lines = f.read().splitlines()

# cria lista de estrofes
estrofes = []
for i, line in enumerate(lines):
    if line.isnumeric():
        estrofes.append(lines[i+1:i+9])

# imprime estrofes
for estrofe in estrofes:
    for verso in estrofe: print(verso)
    print()
```

Exemplo Os *Lusíadas*

- Vamos verificar se está tudo bem.
- Deve ter 1102 estrofes, e 8 816 versos

```
#numero de estrofes  
print(len(estrofes))
```

```
#numero de versos  
numversos = 0  
for estrofe in estrofes:  
    numversos += len(estrofe)  
print(numversos)
```

```
#numero de versos (ordem superior)  
print(sum(map(len, estrofes)))
```

Exemplo Os *Lusíadas*

- Remover caracteres especiais (não alfabeto nem espaço)
- modificar listas in-place, cópia de strings

```
def rem_verso(verso):  
    for c in verso:  
        if not c.isspace() and not c.isalpha():  
            verso = verso.replace(c, '')  
    return verso
```

```
for estrofe in estrofes:  
    for i, verso in enumerate(estrofe):  
        estrofe[i] = rem_verso(verso)
```

Exemplo Os *Lusíadas*

- Guarda estrofes num novo ficheiro 'estrofes.txt'

```
# escreve estrofes para um ficheiro
with open('estrofes.txt', 'w') as f:
    for estrofe in estrofes:
        for verso in estrofe:
            # um verso por linha
            f.write(verso+"\n")
        # linha de espaço entre estrofes
        f.write("\n")
```

Exemplo Os *Lusíadas*

- Contar palavras

```
# versão imperativa
npalavras = 0
for estrofe in estrofes:
    for verso in estrofe:
        npalavras += len(verso.split())
```

```
# versão funcional
def sum_verso(verso):
    return len(verso.split())
def sum_estrofe(estrofe):
    return sum(map(sum_verso, estrofe))
npalavras = sum(map(sum_estrofe, estrofes))
```


Exemplo Os *Lusíadas*

- Média de palavras por estrofe

```
import statistics
```

```
# versão imperativa
```

```
npalavras = []
```

```
for estrofe in estrofes:
```

```
    npalavras.append(sum_estrofe(estrofe))
```

```
avg = statistics.mean(npalavras)
```

```
# versão funcional
```

```
avg = statistics.mean(map(sum_estrofe, estrofes))
```