

ALGORITMIA E ESTRUTURAS DE DADOS

FICHEIROS DE TEXTO EM PYTHON

LICENCIATURA EM
TECNOLOGIAS E SISTEMAS DE INFORMAÇÃO PARA A WEB
#ESMAD #P.PORTO



1. Ficheiros

- ☐ Conceito
- ☐ Ficheiros: o objeto *file*
 - ☐ Modos de abertura de um ficheiro
 - ☐ Fechar ficheiro
 - ☐ Ler ficheiro
 - ☐ Escrever em ficheiro
 - ☐ Posicionar-se em ficheiro
- ☐ Ficheiros Binários
- ☐ *Case Study*

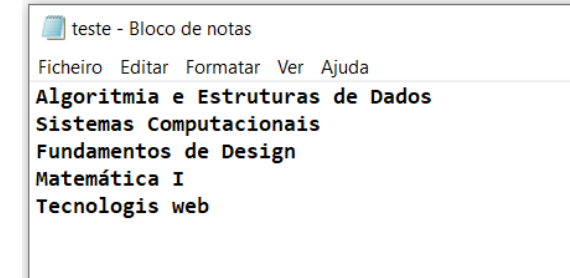


❖ Ficheiros

☐ Tipos de ficheiros:

☐ **Ficheiros de texto:** extensão *.txt*

- Ficheiros que contêm linhas de texto, facilmente legíveis num editor de texto como o Bloco de notas
- Por omissão, os ficheiros manipulados em Python são do tipo ficheiros de texto



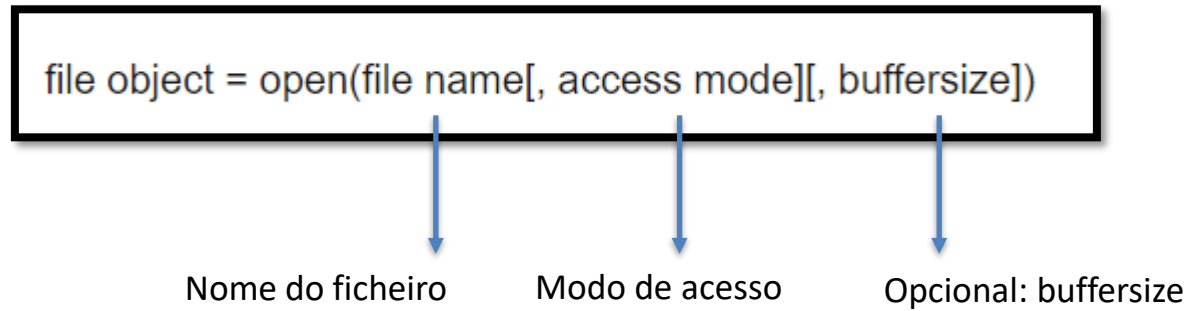
☐ **Ficheiros binários:** extensão *.bin*

- Contêm dados em formato binário. Podem ser úteis para armazenar imagens, ou outros dados em formato binário
- Ao contrário dos ficheiros de texto, os ficheiros binários não são legíveis quando abertos num editor de texto

❖ Ficheiros

❑ Abertura | Acesso a um ficheiro

- ❑ Em Python, um ficheiro físico deve ser mapeado para um objeto do tipo *file*, com recurso ao método **open ()**.



❖ Ficheiros

❑ Abertura | Acesso a um ficheiro

nome do ficheiro modo



```
# abrir o ficheiro teste.txt
fileText1 = open("teste.txt", "r")

# Nome e path do ficheiro
nomeFicheiro = "teste.txt"
filePath = ".\\files\\" + nomeFicheiro
```

❖ Ficheiros

❑ Modos de abertura de um ficheiro

Modo	Acesso	Descrição
r	Read	Abre ficheiro apenas para leitura
w	Write	Cria ficheiro apenas para escrita Se já existir apaga o seu conteúdo
a	Append	Abre ficheiro para acrescentar dados Cria o ficheiro se não existir
x	Create	Cria o ficheiro Devolve um erro se ficheiro já existir
r+	Read & Write	Abre ficheiro para leitura e escrita
w+	Read & Write	Abre ficheiro para leitura e escrita Se já existir apaga o seu conteúdo

❖ Ficheiros

❑ Modos de abertura de um ficheiro

- A qualquer dos modos de abertura anteriormente apresentados, podemos especificar se se trata de um ficheiro de texto ou de um ficheiro binário

```
# Nome e path do ficheiro
nomeFicheiro = "teste.txt"
filePath = ".\\files\\"+ nomeFicheiro

# Abrir um ficheiro de texto
fileText1 = open(filePath, "r")

# Abrir um ficheiro de binário
fileText1 = open(filePath, "rb")
```

❖ Ficheiros

- ❑ Fecho de um ficheiro: método **close()**
 - Depois de ler ou escrever num ficheiro devemos sempre fechar o ficheiro.
 - De contrário, ficará aberto, e quando mais tarde voltarmos a tentar abrir o ficheiro ocorrerá um erro!

```
# Abrir um ficheiro de texto
fileText1 = open(filePath, "r")
# ler o ficheiro, possivelmente
#.....
# fechar o ficheiro
fileText1.close()
```


❖ Ficheiros de Texto

❑ Ler ficheiro: método **readline()**

Método	Descrição
readline()	Lê uma linha do ficheiro de texto para uma <u>string</u>
readlines()	Lê todas as linhas do ficheiro para uma <u>lista</u>
read()	Lê todo o ficheiro para uma <u>string</u>
read(n)	Lê N caracteres (se ficheiro de texto)
read(N)	Lê N bytes do ficheiro (se ficheiro binário)

❖ Ficheiros de Texto

❑ Ler ficheiro de texto

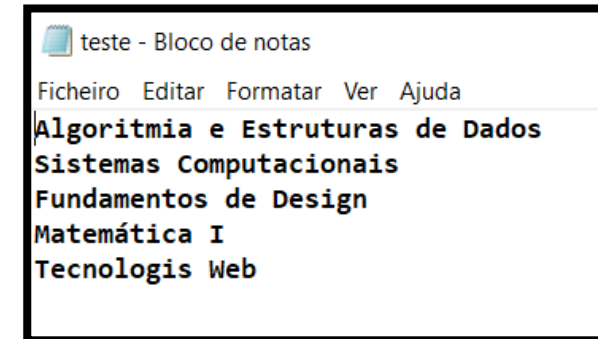
lê uma linha do ficheiro

```
# Nome e path do ficheiro
nomeFicheiro = "teste.txt"
filePath = ".\\files\\"+ nomeFicheiro

# Abrir um ficheiro de texto
fileText1 = open(filePath, "r")

linha = fileText1.readline()
print(linha)

# fechar o ficheiro
fileText1.close()
```



```
Algoritmia e Estruturas de Dados
Press any key to continue . . . |
```

❖ Ficheiros de Texto

❑ Ler ficheiro de texto

Iterar as diversas linhas do ficheiro de texto

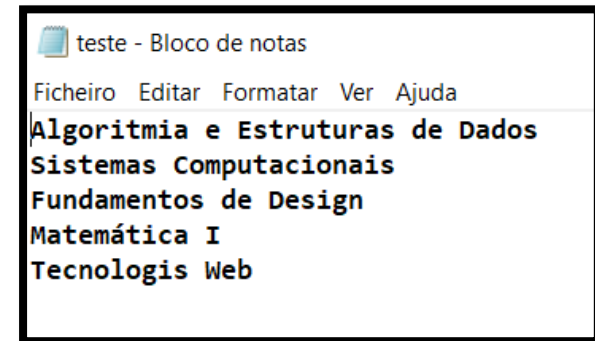
Versão 1:

```
# Nome e path do ficheiro
nomeFicheiro = "teste.txt"
filePath = ".\\files\\"+ nomeFicheiro

# Abrir um ficheiro de texto
fileText1 = open(filePath, "r")

for linha in fileText1:
    print(linha)

# fechar o ficheiro
fileText1.close()
```



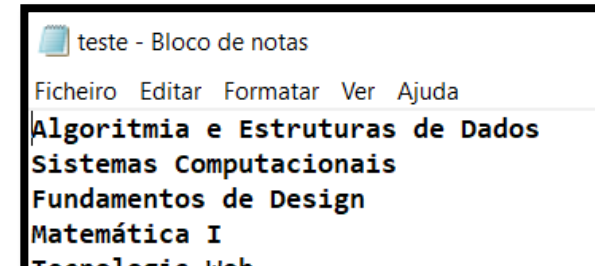
```
Algoritmia e Estruturas de Dados
Sistemas Computacionais
Fudamentos de Design
MatemÃ;tica I
Tecnologias Web
Press any key to continue . . . |
```

❖ Ficheiros de Texto

- ❑ Ler ficheiro de texto

Iterar as diversas linhas do ficheiro de texto
Versão 2:

```
Algoritmia e Estruturas de Dados
Sistemas Computacionais
Fundamentos de Design
Matemática I
Tecnologias Web
Press any key to continue . . . |
```



```
# Nome e path do ficheiro
nomeFicheiro = "teste.txt"
filePath = ".\\files\\"+ nomeFicheiro

# Abrir um ficheiro de texto
fileText1 = open(filePath, "r")
linha = fileText1.readline()
while linha != "":
    print(linha)
    linha = fileText1.readline()
# fechar o ficheiro
fileText1.close()
```

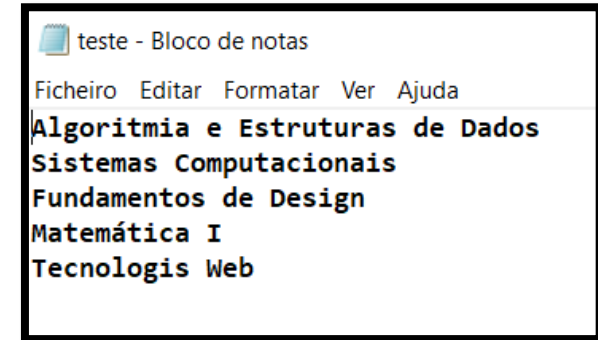
❖ Ficheiros de Texto

❑ Ler ficheiro de texto

Versão 3:

Lê todas as linhas para uma lista

Seguidamente vai iterar as várias posições da lista



```
f = open("teste.txt", "r") # Abre ficheiro para leitura: read
linhas = f.readlines()    # lê todas as linhas do fx para uma lista
f.close()
```

```
print(linhas)
for lin in linhas:
    print(lin)
```

C:\WINDOWS\py.exe

['Algoritmia e Estruturas de Dados\n', 'Sistemas Computacionais\n', 'Fundamentos de Design\n', 'Tecnologias Web\n']

Algoritmia e Estruturas de Dados

Sistemas Computacionais

Fundamentos de Design

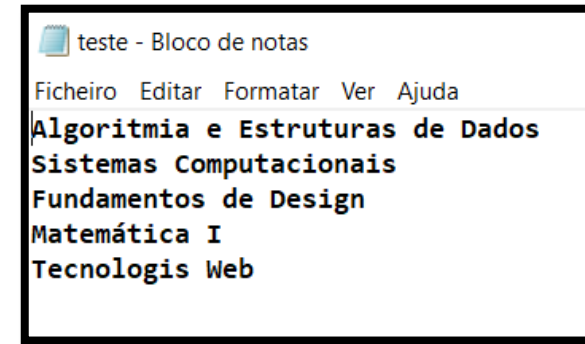
Matemática I

Tecnologias Web

❖ Ficheiros de Texto

❑ Ler ficheiro de texto

Usando a codificação UTF-8



```
f = open("teste.txt", "r", encoding="utf-8") # Abre ficheiro para leitura: read
linhas = f.readlines() # lê todas as linhas do fx para uma lista
f.close()
for lin in linhas:
    print(lin)

input()
```

C:\WINDOWS\py.exe

Algoritmia e Estruturas de Dados

Sistemas Computacionais

Fundamentos de Design

Matemática I

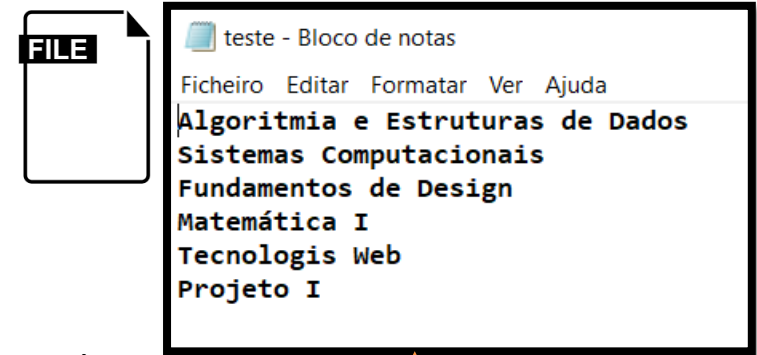
Tecnologias Web

UTF-8 é um tipo de codificação binária (Unicode) de comprimento variável criado por [Ken Thompson](#) e [Rob Pike](#). Pode representar qualquer caracter universal padrão do [Unicode](#), sendo compatível com a tabela ASCII

❖ Ficheiros de Texto

❑ Escrever em ficheiro de texto: método **write()**

- Modo **'a'** : acrescenta linhas ao ficheiro
- Modo **'w'** : cria novamente o ficheiro, para escrita (se já existir apaga-o)

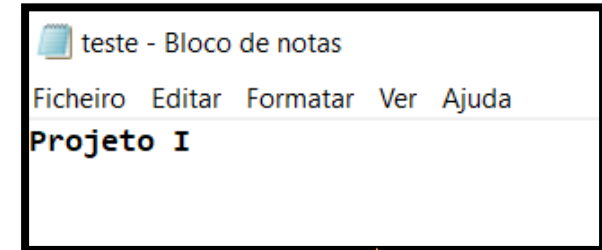


```
44
45 f = open("teste.txt", "a") # Abre ficheiro em modo append
46 linha = "Projeto I"
47 f.write(linha)
48 f.close()
49
50
```

❖ Ficheiros de Texto

❑ Escrever em ficheiro de texto: método **write()**

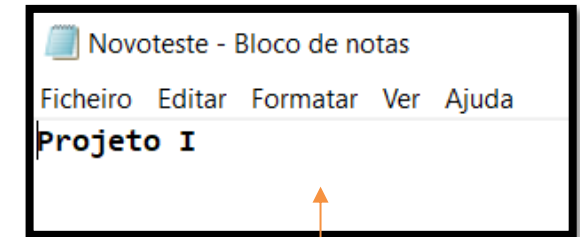
- Modo '**a**' : acrescenta linhas ao ficheiro
(não apaga o seu conteúdo, caso ficheiro já exista)
- Modo '**w**' : cria novamente o ficheiro, para escrita
(se já existir apaga o seu conteúdo). O modo w trata sempre como se fosse um novo ficheiro



```
44
45 f = open("teste.txt", "w") # Abre ficheiro em modo write
46 linha = "Projeto I"
47 f.write(linha)
48 f.close()
49
```


❖ Ficheiros de Texto

- ❑ Criar um novo ficheiro: método **write()**
 - Modo **'x'** : Cria um novo ficheiro
(devolve um erro se o ficheiro já existir)



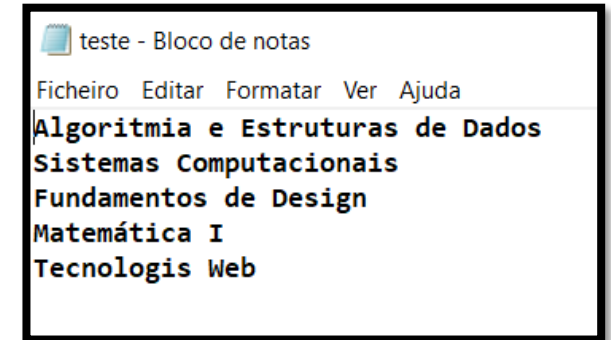
```
4  
5 f = open("Novoteste.txt", "x") # Abre ficheiro em modo Create New File  
6 linha = "Projeto I"  
7 f.write(linha)  
8 f.close()  
9
```

Tip !


Para remover um ficheiro em disco (também devolve um erro se ficheiro não existir):
`import os`
`os.remove("fileName.txt")`

❖ Ficheiros de Texto

- ❑ Posicionar no ficheiro: método ***seek(position, from)***
 - Posiciona numa determinada posição do ficheiro
 - ***from***:
 - 0: posição determinada a partir do início do ficheiro
 - 1: posição determinada a partir da posição atual
 - 2: posição determinada a partir do fim do ficheiro



```
f = open("teste.txt", "r") # Abre ficheiro para leitura: read
f.seek(13,0)
linha = f.readline()
print(linha)
f.close()
```

 C:\WINDOWS\py.exe
Estruturas de Dados

❖ Ficheiros de Texto

❑ Síntese de métodos relacionados com o objeto *file*:

Método	Descrição
<code>close()</code>	Fecha o ficheiro
<code>readline()</code>	Lê uma linha do ficheiro de texto
<code>readlines()</code>	Lê todas as linhas do ficheiro para uma lista
<code>read()</code>	Lê todo o ficheiro para uma string
<code>read(N)</code>	Lê N bytes/caracteres do ficheiro
<code>write(str)</code>	Escreve uma string no ficheiro
<code>seek(pos, from)</code>	Posiciona num determinado caracter do ficheiro

❖ Ficheiros Binários

❑ Escrever e Ler em **ficheiros binários**

```
exemplos1.py > ...  
1  
2 f=open("teste.bin","wb")  
3  
4 linha ="Algoritmia e Estruturas de Dados\n"  
5 linha1 = bytes(linha, encoding="utf-8")  
6 f.write(linha1)  
7  
8 linha ="Fundamentos de Design\n"  
9 linha1 = bytes(linha, encoding="utf-8")  
0 f.write(linha1)  
1 f.close()  
2
```

w- write
b- binary file

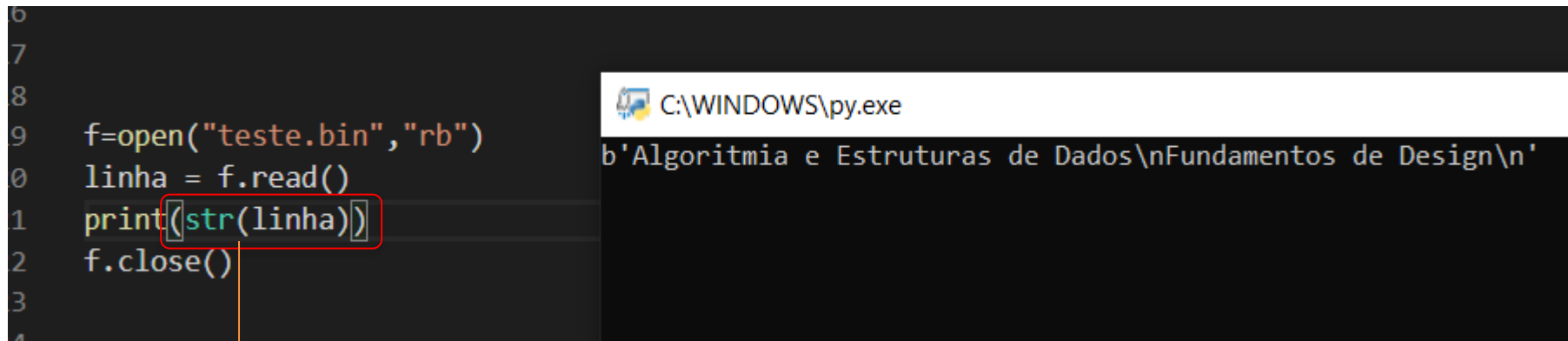
Converte a string para bytes

❖ Ficheiros Binários

❑ Ler e escrever em ficheiros binários

Lê todo o conteúdo do ficheiro binário

```
6
7
8
9 f=open("teste.bin","rb")
0 linha = f.read()
1 print(str(linha))
2 f.close()
3
4
```



```
C:\WINDOWS\py.exe
b'Algoritmia e Estruturas de Dados\\nFundamentos de Design\\n'
```

Converte para string

❖ Ficheiros Binários

- ❑ Ler e escrever em ficheiros binários

```
18  
19 f=open("teste.bin","rb")  
20 linha = f.read(32)  
21 print(str(linha))  
22  
23 linha = f.read(21)  
24 print(str(linha))  
25 f.close()  
26  
27
```

```
C:\WINDOWS\py.exe  
b'Algoritmia e Estruturas de Dados'  
b'Fundamentos de Design'
```

Lê N bytes de cada vez,
do ficheiro binário

❖ Ficheiros Binários

❑ Ler e escrever em ficheiros binários (listas, arrays)

Converte a lista para binário

```
lista = [1,2,3,4,5,6,7,8,9,10]    # Lista com numeros
lista=bytearray(lista)            # converte lista para binário

fLista=open("teste.bin","wb")      # Abre ficheiro binário em modo 'w'
fLista.write(lista)                # Grava em ficheiro
fLista.close()

fLista=open("teste.bin","rb")
nova_lista=list(fLista.read())
fLista.close()

print(nova_lista)
for item in nova_lista:
    print(item)
```

c:\Users\mario\OneDrive\AED\4 - Exercicios\Ficha 09 - VS Code Cons

[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

1
2
3
4
5
6
7
8
9
10
Press any key to continue . . .

❖ Um exemplo de um simulador de temperatura

APLICAÇÃO 1: SIMULADOR DE SENSOR DE TEMPERATURA

Pretende-se implementar um programa que permita registar a temperatura verificada num determinado espaço, simulando a aquisição de dados a partir de um sensor de temperatura.

O programa deve funcionar como um simulador do comportamento do sensor de temperatura, gerando valores inteiros e aleatórios de temperatura, entre 10º e 25º, e com uma periodicidade de 1 segundo. A data e a hora devem ser as de sistema.



C:\WINDOWS\py.exe

Data	Hora	Temperatura
2020-12-01	16:57:19	13
2020-12-01	16:57:20	23
2020-12-01	16:57:21	15
2020-12-01	16:57:22	14
2020-12-01	16:57:23	23
2020-12-01	16:57:24	20
2020-12-01	16:57:25	17
2020-12-01	16:57:26	23
2020-12-01	16:57:27	11
2020-12-01	16:57:28	14

❖ Um exemplo de um simulador de temperatura

1. Simular a aquisição de dados do sensor, através da leitura de dados aleatórios. Guardar em ficheiro.

c:\Users\mario\OneDrive\AED\4 - Exercicios\Ficha 09 - VS Code Console

Data	Hora	Temperatura
2022-11-21	10:47:15	25
2022-11-21	10:47:16	22
2022-11-21	10:47:17	17
2022-11-21	10:47:18	25
2022-11-21	10:47:19	11
2022-11-21	10:47:20	16
2022-11-21	10:47:21	17
2022-11-21	10:47:22	13



temperatura - Bloco de notas

Ficheiro	Editar	Ver
2022-11-21;10:47:15;25		
2022-11-21;10:47:16;22		
2022-11-21;10:47:17;17		
2022-11-21;10:47:18;25		
2022-11-21;10:47:19;11		
2022-11-21;10:47:20;16		
2022-11-21;10:47:21;17		
2022-11-21;10:47:22;13		

❖ Um exemplo de um simulador de temperatura

1. Simular a aquisição de dados do sensor, através da leitura de dados aleatórios.
Guardar em ficheiro.

```
1  import random                # módulo que inclui o método randint (números aleatórios)
2  from datetime import datetime # módulo que inclui métodos para obter data e hora
3  import time                  # módulo que inclui o método sleep
4
5  def saveFile(linha):
6      """
7      Receives a line with temp data and saves in the file
8      """
9      fTemp = open("temperatura.txt", "a") # abre o ficheiro em modo de Append.
10     fTemp.write(linha)
11     fTemp.close()
12
13     # Funciona num do for ever, o ciclo while +e semper veraddeiro. Termina com o fecho da aplicação
14     # Simula a leitura de um sensor de temperatura, gerando valor aleatório com intervalos de 1 segundo
15     print("\t\t Data \t Hora \t Temperatura")
16     print("\t\t-----")
17     fim = False
18     while not fim:
19         temp = random.randint(10,25) # valor aleatorio da temperatura
20         data = datetime.now().date() # obtem data de sistema
21         hora = datetime.now().time().strftime("%H:%M:%S") # Obtem hora de sistema H:M:S
22         print("\t\t", data, "\t", hora, "\t", temp) # Imprime
23         linha = str(data) + ";" + str(hora) + ";" + str(temp) + "\n" # Constroi string linha para guardar dados no fi
24         saveFile(linha)
25         time.sleep(1) # faz pausa de 1 segundo
```

❖ Um exemplo de um simulador de temperatura

2. Leitura do ficheiro de texto e impressão dos dados do ficheiro

```
1 import random # módulo que inclui o método randint (números aleatórios)
2 from datetime import datetime # módulo que inclui métodos para obter data e hora
3 import time # módulo que inclui o método sleep
4 import os
5
6 # -----
7 def readFile():
8
9     if not os.path.isfile("temperatura.txt"):
10         print("O ficheiro não existe!")
11         input()
12         return
13     print("\t\t Data \t Hora\t\tTemperatura")
14     print()
15
16     ftemp = open("temperatura.txt", "r")
17     listaFile = ftemp.readlines() # ler ficheiro todo para uma lista
18     ftemp.close()
19     for linha in listaFile:
20         campos = linha.split(";") # cada linha da lista é dividida em 3 partes, pelos ";"
21         print("\t\t", campos[0], "\t", campos[1], "\t", campos[2][:-1])
22
23
24 readFile()
```

c:\Users\mario\OneDrive\AED\4 - Exercicios\Ficha 09 - VS Code Console

Data	Hora	Temperatura
2022-11-21	10:47:15	25
2022-11-21	10:47:16	22
2022-11-21	10:47:17	17
2022-11-21	10:47:18	25
2022-11-21	10:47:19	11
2022-11-21	10:47:20	16
2022-11-21	10:47:21	17
2022-11-21	10:47:22	13
2022-11-21	10:47:23	11
2022-11-21	10:47:24	13
2022-11-21	10:59:02	13
2022-11-21	10:59:03	10
2022-11-21	10:59:04	18
2022-11-21	10:59:05	14
2022-11-21	10:59:06	12
2022-11-21	10:59:07	12

Press any key to continue . . .