

ALP

EXERCISE SHEET
ACADEMIC ACTIVITY

Algorithms and Data Structures

CURRICULAR UNIT

Sheet 05 - Strings

1. Write a program (not a function) that reads a sentence and writes its characters in reverse order.

```
C:\WINDOWS\py.exe
Indique um texto:Algoritmia e Estruturas de dados
sodad ed saruturtsE e aimtirogLA_
```

2. Write a program (not a function) that reads a sentence and determines:

- Number of characters
- Number of spaces
- Number of vowels

```
C:\WINDOWS\py.exe
Indique um texto:Algoritmia e Estruturas de Dados
Nº de caracteres: 32
Nº de vogais : 13
Nº de espaços : 4
```

3. Create a **capicua** function that receives a text as an input parameter and **returns** True or False, depending on whether the text is a capicua or not.

A capicua consists of a text that can be read both from left to right and from right to left.

Capicua examples: *osso, asa, ana*

Examples of using your capicua function:

capicua('osso') => returns True
capicua('roma') => returns False

C:\WINDOWS\py.exe

```

Insira um texto:arara

O texto é capicua

```

4. Write the **removeSpaces** function that takes a text and replaces sequences of two or more spaces with a single space. The function should print the resulting text.

C:\WINDOWS\py.exe

```

Texto:Algoritmia e Estruturas de Dados

Texto: Algoritmia e Estruturas de Dados

```

5. Write the **shortName** function that must receive a full name and **returns** a string with the first name and last name.

Example:

shortName('Manuel Jorge da Costa Pereira') => Manuel Pereira

C:\WINDOWS\py.exe

```

Nome:Manuel Jorge da Costa Pereira

Manuel Pereira

```

6. Create the **standardName** function that must receive a full name and **returns** a string with the normalized name: it includes the first and last name and abbreviations of all other intermediate names.

Example:

standarName('Carlos Alberto Costa Pereira') => Carlos A. C. Pereira

C:\WINDOWS\py.exe

```

Nome:Carlos Alberto Costa Pereira

Carlos A. C. Pereira

```

7. Elaborate the **generatePassword** function that works as a password generator: the function must receive a username, and based on that name it must generate and **return** a password that is formed as follows:

- Password consists of the characters from the even positions in the username, combined with a random number between 1 and 9 (inclusive).
- The password ends with the number of characters indicated in the username.

Example:

```
username = Carlos
```

```
generatePassword(userName) => Password: a3l2s76 (it's an example, the random numbers could be different!)
```

If the username includes any spaces, the function must return the message "username is invalid".

8. Write the **reverseWords** function that receives a text and **returns** the same text, but with the words in reverse order.

Example:

```
Text= This is an AED test
```


```
reverseWords(text) => test AED an is This
```

9. Write the **countWord** function that takes a text and a search word. The function should return the number of occurrences of this word in the text, and the positions where it occurs.

One example:

Text: *Using a GPS/GPRS unit installed in each vehicle, a monitoring device records real-time information about the location of each vehicle.*

```
countWord(text, 'vehicle') => returns 2 (2 occurrences)
```

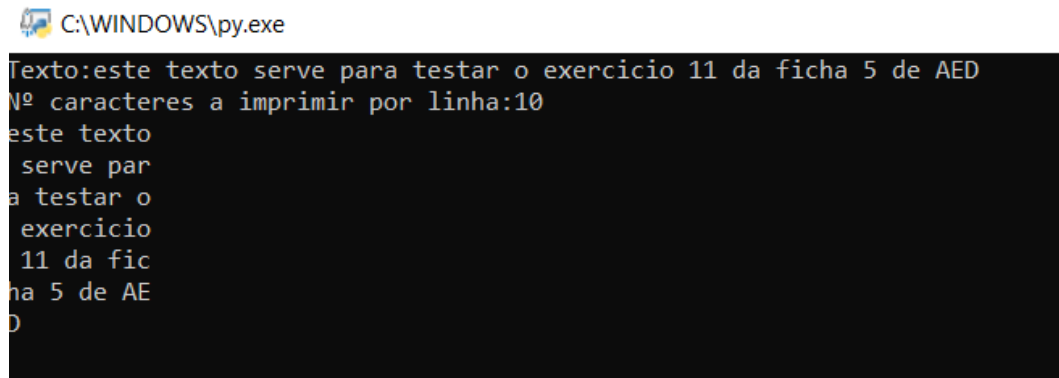
 C:\WINDOWS\py.exe

```
Texto: teste do ex 9 teste sobre strings
Pesquisa: teste
A palavra teste ocorre 2 vezes no texto. Nas posições 1 15
```

(note: you can assume that the search text does not contain ",", and ".". Or you can start by replacing these characters by spaces).

10. Implement the **printCharLine** function that takes two arguments: a text and the number of characters you want to print per line.

Your function should print the text depending on this number of characters, as shown in the image below.



```
C:\WINDOWS\py.exe
Texto:este texto serve para testar o exercicio 11 da ficha 5 de AED
Nº caracteres a imprimir por linha:10
este texto
serve par
a testar o
exercicio
11 da fic
ha 5 de AE
D
```