Université Paris Dauphine **Licence MIE 1re année**
Algorithmique et programmation 1 **2025-2026**

5. Chaînes de caractères en Python (TP)

Exercice 5.5

1. Écrire un programme Python qui lit une chaîne de caractères `s`, affiche sa longueur, puis affiche `s` avec tous les espaces du début, intermédiaires et de la fin, supprimées.
2. Écrire un nouveau programme Python qui permet, non pas simplement d'afficher `s`, mais de modifier `s` en enlevant tous les espaces du début, intermédiaires et de la fin.

Exercice 5.6

Écrire un programme Python qui lit une chaîne de caractères, puis lit une ou plusieurs valeurs entières (tant que l'utilisateur le souhaite) et qui, pour chacune de ces valeurs `v`, remplace le caractère situé à l'indice `v` par le caractère `'?'`. Ce programme se termine par l'affichage de la nouvelle chaîne. Attention : il faut bien vérifier que les valeurs `v` correspondent à des indices valides pour la chaîne de caractères lue.

Exercice 5.7

Écrire un programme Python qui lit une chaîne de caractères et qui affiche cette chaîne avec les suites de caractères identiques remplacées par un seul caractère. Par exemple, on donnera la chaîne `"abbcdebbb"` et votre programme devra afficher `"abcdeb"`.

Exercice 5.8

Écrire une fonction Python, appelée `test()`, qui prend 3 arguments : un entier `n` positif, et deux chaînes de caractères (ne contenant pas d'espace) `ch1` et `ch2`. Pour simplifier vos tests ultérieurs, on supposera que l'argument `ch1` a comme valeur par défaut `"habhabab"`. Cette fonction renvoie `True` si les deux conditions suivantes sont vérifiées : `ch1` contient `n` fois `ch2`, et dans `ch2` tout caractère est utilisé au plus une fois ; sinon la valeur booléenne `False` est renvoyée.
Écrire un programme Python qui lit un entier `n` positif, puis une ou deux chaînes de caractères (ne contenant pas d'espace) `ch1` et `ch2`, et appelle la fonction `test()` sur ces variables pour afficher `"oui"` si la valeur de retour de `test()` est `True` et `"non"` sinon. Le tableau ci-dessous donne des exemples de valeurs que le programme doit afficher.

ch1	ch2	n	valeur affichée
"habhabab"	"abh"	1	oui
"habhabab"	"abh"	2	non
"habhabab"	"ab"	3	oui
"habhabab"	"ab"	4	non
"baaabaa"	"aa"	2	non
"baaabaa"	"aa"	3	non
"ababaaaba"	"ab"	3	oui

Exercice 5.9

Écrire une fonction Python, appelée `test2()`, qui prend deux chaînes de caractères `ch1` et `ch2` en argument. Pour simplifier vos tests ultérieurs, on supposera que l'argument `ch2` a comme valeur par défaut `"ab"`. Cette fonction renvoie un entier `N` tel que `ch1` est égale à `N` concaténations successives de `ch2` avec elle-même (ou 0 si ce n'est pas le cas).
Écrire un programme Python qui lit une ou deux chaînes de caractères `ch1` et `ch2`, et appelle la fonction `test2()` pour afficher `N` s'il existe ; dans le cas contraire, le programme affichera `"non"`. On suppose que l'utilisateur saisira toujours des chaînes de caractères `ch1` et `ch2` non vides ne contenant pas d'espace. Le tableau ci-dessous donne des exemples de valeurs que le programme doit afficher.

ch1	ch2	valeur affichée
"ababab"	"ab"	3
"ab"	"ab"	1
"abhabab"	"ab"	non
"aaaa"	"aa"	2
"aaa"	"aa"	non
"aaa"	"a"	3

Exercice 5.10

Écrire un programme Python qui lit une chaîne de caractères et l'affiche sous la forme d'un `X`. Si l'on saisit par exemple `"abcde"`, alors il doit afficher :

```
a   e
 b d
  c
 b d
a   e
```

Si l'on saisit par exemple `"abcd"`, alors il doit afficher :

```
a  d
 bc
 bc
a  d
```

Tester votre programme, entre autre avec les chaînes de caractères `MAGIQUE` et `PYTHON`, et avec la chaîne obtenue en concaténant les deux précédentes.

Exercice 5.11

Écrire un programme Python qui prend en entrée deux chaînes de caractères `s1` et `s2` (supposées sans espace), et affiche `"OK"` s'il est possible, en enlevant certains caractères de `s2`, de retrouver `s1`. Dans le cas contrainte, le programme affiche `"impossible"`.
Par exemple :
- avec `s1='bd'` et `s2='abcde'`, le programme affiche `"OK"`,
- avec `s1='bd'` et `s2='bd'`, le programme affiche également `"OK"`,
- par contre, il affiche `"impossible"` lorsque `s1='db'` et `s2='abcde'`.

Exercice 5.12

Il pîaart que puor la lctreue, l'orrde des lrttees à l'iétunreir des mots n'a acnuue itnpocmare. La sulee chose qui cptmoe est que la pemièrre et la dneèirre lttree seonit à leur pclae.
L'objectif de cet exercice est de concevoir un programme en **Python** permettant de tester cette théorie. Étant donnée une chaîne de caractères saisie par l'utilisateur, il mélangera aléatoirement les lettres à l'intérieur des mots et affichera la phrase modifiée. On supposera que la chaîne de caractères ne comporte pas de signe de ponctuation et que deux mots sont séparés par exactement une espace.

1. Écrire une fonction `permuteMot()` qui prend en argument un mot et permute aléatoirement les lettres à l'intérieur du mot en gardant les première et dernière lettres inchangées.
2. Écrire une fonction `permute()` qui prend en argument une phrase et applique la première fonction sur chacun des mots.
3. Écrire le programme demandé en utilisant les fonctions précédentes.

Exemple : L'utilisation saisit la phrase `"Je vais avoir une bonne note"`, alors le programme peut afficher `"Je vias avior une bnone ntoe"`.

Exercice 5.13

Écrire un programme Python qui lit une chaîne de caractères `ch` et qui affiche la liste des mots de `ch` ainsi que les indices de début de chacun des mots dans `ch` (chaque mot pouvant se répéter plusieurs fois).
Par exemple, pour `ch=" ab cda fg ab  fg cda h cda  "`, le programme doit afficher :

```
"ab" 1, 11
"cda" 4, 19, 25
"fg" 8, 16
"h" 23
```

Attention, plusieurs espaces peuvent se trouver entre deux mots consécutifs, ainsi qu'avant le premier mot et après le dernier mot.