



Taki Academy
www.takiacademy.com

Algo & Prog

Classe : Bac Scientifique

Série : **Algorithmme ET Interface**

Nom du Prof : Salem Ounis

📍 Sousse (Khezama - Sahloul) Nabeul / Sfax / Bardo / Menzah El Aouina /
Ezzahra / CUN / Bizerte / Gafsa / Kairouan / Medenine / Kébili / Monastir /
Gabes / Djerba / Jendouba / Sidi Bouzid / Siliana / Béja / Zaghouan



www.takiacademy.com



73.832.000



Exercice 1



Soit la séquence python suivante :

```
R = ""
i = p
while (i != q):
    R=R+ch[i]
    i=i+1
```

Questions :

1) Exécuter à la main cette séquence pour les valeurs suivantes :

ch="Algorithmique", p=3 et q=6 R=

ch="Informatique", p=0 et q=4 R=

2) donner l'appel de la fonction prédéfinie en algorithmique qui fournit le même résultat.

.....

Exercice 2



Une société se propose d'envoyer, à travers le réseau Internet, les informations de ses transactions aux membres du conseil d'administration sous forme de messages.

Afin de garder la confidentialité de ces messages, le responsable de la société compte utiliser une version allégée d'un algorithme de cryptage appelé "**masque jetable**".

Le principe de cryptage est décrit comme suit :

1^{ère} étape : Créer une **Clé**, de façon que pour chaque lettre de l'alphabet on lui fait correspondre un nombre aléatoire entre **33 et 99 sans redondance**.

2^{ème} étape : Remplacer chaque caractère du message à crypter par le nombre correspondant dans la **Clé**, sachant que l'espace sera codé par "**32**". On obtient ainsi un deuxième message **RES**.

3^{ème} étape : Découper le message obtenu en blocs de **deux** chiffres où chaque bloc correspond au code ASCII d'un caractère.

Le message crypté **MSG** sera formé en associant à chaque bloc le caractère qui lui correspond.

Exemple :

Le message à envoyer est : **VERS LA VICTOIRE**

Le message crypté reçu sera : **#&M! 0G # %VF? %M&**

En effet, on commence par créer la **Clé** comme suit :

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Clé	71	41	86	81	38	67	39	68	37	43	92	48	75	84	63	47	91	77	33	70	72	35	47	25	82	45

- On fait correspondre pour chaque lettre du message envoyé le nombre correspondant dans **Clé** en tenant compte des espaces. On obtient le message **RES** suivant :

35387733324871323537867063377738

- On découpe ce message par bloc de deux chiffres et on fait correspondre pour chaque nombre un caractère :

RES	35	38	77	33	32	48	71	32	35	37	86	70	63	37	77	38
MSG	#	&	M	!		0	G		#	%	V	F	?	%	M	&

On obtient le message crypté **MSG** suivant : **#&M! 0G #%%VF?%M&**

On se propose d'écrire un programme qui permet de crypter un message donné, formé uniquement par des lettres majuscules où les mots sont séparés par un seul espace, en utilisant le procédé de cryptage décrit précédemment et d'afficher le message obtenu.

Travail demandé :

- Écrire l'algorithme du programme principal, solution à ce problème, en le décomposant en modules.
- Écrire l'algorithme de chaque module.

N. B. : Chaque algorithme doit être accompagné des tableaux de déclaration nécessaires.

Exercice 3



Important :

- Une solution modulaire au problème posé est exigée.
- Dans le répertoire **Bac2023**, créez un dossier de travail ayant comme nom votre numéro d'inscription (6 chiffres) et dans lequel vous devez enregistrer, au fur et à mesure, tous les fichiers solution à ce sujet.

Miroirs de mots

On se propose de concevoir une interface graphique permettant de saisir une chaîne de caractères **ch** et de la crypter en formant une nouvelle chaîne par les miroirs des mots de **ch** dans leur ordre d'apparition. On rappelle que le miroir d'un mot consiste à permuter le premier caractère avec le dernier, le deuxième caractère avec l'avant dernier et ainsi de suite.

Exemple :

Pour **ch** = "La vie est une aventure merveilleuse"

On obtient la chaîne cryptée : "aL eiv tse enu erutneva esuellievrem"

L'interface graphique à concevoir contient les éléments suivants, comme l'illustre la capture d'écran ci-dessous :

- Un label contenant le texte "**Miroirs de mots**"
- Un label contenant le texte "**Introduire une chaîne :** "
- Une zone de saisie pour la saisie d'une chaîne
- Un label pour afficher le résultat
- Un bouton intitulé "**Miroir**"

Miroirs de mots

Introduire une chaîne :

Travail demandé :

- 1) Concevoir l'interface graphique présentée précédemment et l'enregistrer sous le nom **InterfaceMiroirsMots**.
- 2) Créer un programme Python et l'enregistrer sous le nom **MiroirsMots**, dans lequel, il est demandé :
 - a) de développer une fonction nommée **Miroir (M)** qui permet de retourner le miroir d'un mot **M**.
 - b) de développer un module **Play**, qui s'exécute suite à un clic sur le bouton "**Miroir**", permettant :
 - de récupérer la chaîne **ch** saisie. La chaîne **ch** doit être non vide et de longueur inférieure à 50, contient seulement des lettres alphabétiques en minuscule et chaque deux mots consécutifs sont séparés par un seul espace.
 - de déterminer la chaîne cryptée en utilisant la fonction **Miroir (M)** et d'afficher le résultat via le **label** dédié à l'affichage dans l'interface graphique **InterfaceMiroirsMots**.
 - c) d'ajouter les instructions permettant d'exploiter l'interface graphique intitulée **InterfaceMiroirsMots** en se référant à l'annexe ci-après.

N.B. : l'affichage doit être conforme aux exemples d'exécutions suivants :

Exemples d'exécutions :

Miroirs de mots

Introduire une chaîne :

Veuillez introduire chaîne

Miroirs de mots

Introduire une chaîne :

Entre 2 mots un seul espace est autorisé

Miroirs de mots

Introduire une chaîne :

aL eiv tse enu erutneva esuellievrem

Annexe

```

from PyQt5.uic import loadUi
from PyQt5.QtWidgets import QApplication
.....
.....
app = QApplication([])
windows = loadUi("Nom_Interface.ui")
windows.show()
windows.Nom_Bouton.clicked.connect (Nom_Module)
app.exec_()
```

Grille d'évaluation

Tâches	Nombre de points
Conception de l'interface " InterfaceMiroirsMots "	4 pts
Création et enregistrement du programme " MiroirsMots "	1 pt
Développement de la fonction " Miroir "	4 pts
Développement du module " Play "	6 pts
Ajout des instructions de l'exploitation de l'interface	3 pts
Modularité et cohérence	2 pts