



Numérique et Science Informatique

Spécialité de première

Projet 1 : cryptographie

Travail à rendre :

Vous devrez rendre un dossier par binôme (**à vos noms et prénoms**) contenant un fichier python par exercice.

La date limite pour la restitution du projet est fixée au vendredi 19 novembre 2021.

Vous veillerez à avoir plusieurs sauvegardes de votre projet (sur ordinateur et sur clé) afin de pouvoir le rendre à la date limite.

Aucune excuse relative à un dysfonctionnement matériel (wifi, clé, etc) ne sera acceptée en cas de non rendu de projet.

Vous déposerez un script python dans une conversation Teams avec votre professeur de NSI.

Le projet :

Vous répondrez aux questions posées dans les exercices.

Le minimum attendu est de deux fichiers correspondants aux exercices 1 et 3.

Les autres exercices seront des bonus.

Les fonctions seront bien séparées, le code sera commenté, chaque fonction aura un **docstring** et sera commentée avec des # ; de plus, un exemple d'utilisation sera implémenté.

Vous utiliserez uniquement les structures de Python qui ont été vues dans les TD précédents.

Aucune bibliothèque n'est autorisée sauf Tkinter.

Chaque script python devra proposer une utilisation conviviale de votre code : l'utilisateur devra pouvoir faire des choix et saisir lui-même du texte (et les clés utilisées).

La moitié de la note concernera le travail en cours et la gestion du projet.

Etape 1			*
----------------	---	--	---

Le code de César.

En cryptographie, un chiffrement de César, également connu sous le nom de *code de César* (par abus de langage), est l'une des techniques les plus simples et les plus connues de cryptage.

Il s'agit d'un type de chiffrement par substitution, dans lequel on décale juste d'un certain nombre, les lettres de l'alphabet.

Il s'agit d'une *permutation circulaire* de l'alphabet.

Par exemple, avec un décalage de 3: (c'est le code de César original)

- A serait remplacé par D,
- B deviendrait E,
- [...]
- Y devient B,
- et Z devient C.

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
D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

1. Codez la fonction `Cesar3(text)` qui encode la chaîne de caractères `text` (caractères majuscules sans espaces) à l'aide du chiffre de César avec un décalage de 3.
2. Améliorez la fonction (vous nommerez la fonction améliorée `Cesar3Plus(text)`) afin qu'elle renvoie un message d'erreur si la chaîne de caractères `text` n'est pas au format souhaité. On pourra créer la fonction `checkMajuscules(text)` qui renvoie `True` uniquement si son argument est un texte en majuscules.
3. Codez la fonction `Cesar(text,n)` qui encode la chaîne de caractères `text` (caractères majuscules sans espaces) à l'aide du chiffre de César avec un décalage de `n`.

Etape 2 (optionnel)			* * *
--------------------------------	---	--	-------

Utilisez le dernier exercice du TD Fonctions Python pour formater le texte avant de le chiffrer.

Etape 3			* *
----------------	---	--	-----

Attaque par force brute :

Codez `attaqueForceBruteCesar(text)` qui retourne les 25 messages originaux possibles pour `text` préalablement chiffré avec un code de César ainsi que les décalages utilisés.

Pour aller plus loin			* * *
-----------------------------	---	--	-------



Un autre algorithme de chiffrement historique assez connu est le **chiffre de Vigenère**. Comme pour le code de César on va décaler les lettres dans l'alphabet, sauf qu'ici la clé n'est pas un nombre mais un autre mot, et on va décaler chaque lettre du message en additionnant le numéro (de 0 à 25 pour nous) dans l'alphabet de la lettre ayant la même position que celle du message dans la clé. On répète ensuite la clé autant de fois que nécessaire.

Si je prends la chaîne « abcde » avec la clé 'aab' ou plutôt 'aabaa' pour que la longueur corresponde :

Ma chaîne chiffrée est « abdde »

Vous allez devoir implémenter ce chiffrement en Python après avoir complété la clé pour qu'elle soit aussi longue que le message.

Créez la fonction `vigenere(texteclair,cle)` qui renvoie le message chiffré.

Pour aller beaucoup plus loin			* * * *
--------------------------------------	---	---	---------

Deux attaques possibles du chiffre de Vigenère :

Par un mot probable.

Attaque utilisant le test de Kasiski.

Vous pouvez implémenter l'une de ces 2 attaques en Python.