L’attaque de César

version 2.0.2





1. Introduction

Nous sommes en l’an 50 avant Jésus-Christ, toute la Gaule est occupée par les Romains... Toute ? Non ! Un village peuplé d'irréductibles Gaulois résiste encore et toujours à l’envahisseur Romains ! Le village gaulois, se prépare à une future attaque de l’empire de Julios César.

Codingtamix habite ce village, c’est un jeune homme talentueux ayant intercepté un pigeon voyageur qui détenait une note chiffrée à destination des légionnaires romains des camps retranchés d’Epidorium. Ce dernier, souhaiterait déchiffrer ce message pour protéger son village et sa famille des envahisseurs.

Pour cela, il va utiliser le code César, qui permet de déchiffrer le message, avant que l’attaque se produise.



Carte de la Gaule

1. Consignes

* Lisez tout avant de commencer !
* Pour ce projet, il vous sera demandé de choisir comme nom de repository : cc\_attaque\_cesar
* Si tu bloques, rappelles-toi que tu es accompagné(e) ! Demande de l’aide à tes camarades ou à un Cobra. L’union fait la force !
* Internet est un outil formidable pour découvrir le fonctionnement des choses, Google est ton ami sers-t ’en régulièrement !
* Pour ce projet, il faut extraire l’archive fournie par les cobras.

Explication du code :

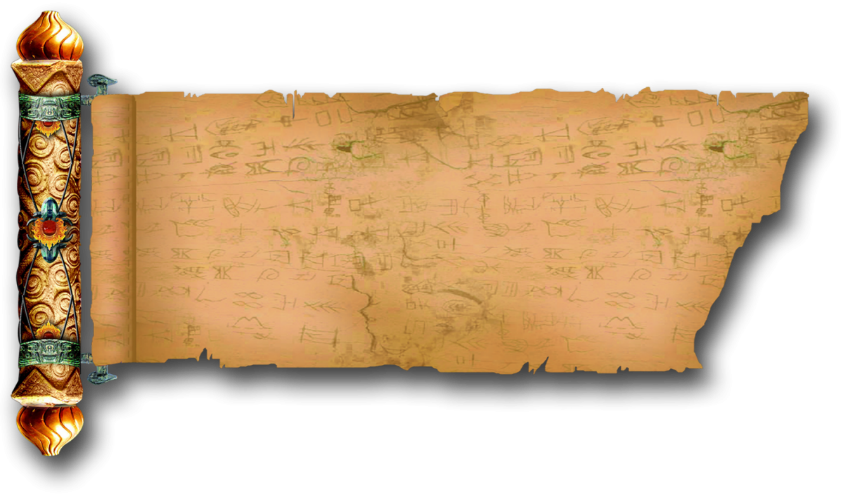
* Un … dans le code signifie que tu dois compléter le code par toi-même en utilisant les informations du sujet.
* Tout ce qui est écrit après un // est un commentaire pour t’aider à comprendre.
* Les exemples de code fourni ci-dessous sont incomplets, vous devrez rajouter ou modifier quelques éléments pour que cela fonctionne.

1. Sauvons la Gaule !
   1. Un peu d’histoire

En cryptographie, le code de César, est une méthode de chiffrement très simple utilisée par Julios César dans ses correspondances secrètes.

Cette méthode de chiffrement vient donc d'avant Jésus Christ ! Ainsi, déjà à cette époque, la volonté de protéger les données était d'une importance non négligeable. Au fur et à mesure du temps, les méthodes de chiffrement se sont bien évidemment modernisées, et sont devenues de plus en plus complexes.

Entrons dans le monde de la cryptographie avec Codingtamix !



La note trouvée par Codingtamix.

* 1. Après la théorie, c’est l’heure de la pratique !

Avant de commencer à déchiffrer le message, Codingtamix va préparer tout son matériel pour bien travailler dans “Installations Python et ses outils”.

Après cette mise en place, Codingtamix prend un parchemin qui se nomme “decrypt.py” ainsi que le message intercepté : “message-chiffre.txt”.

Sur ce parchemin il va créer une fonction “decrypt” qui prend en paramètre deux variables :

* Text
* Step

def decrypt(text,step)

A la suite de cela, plusieurs paramètres d’erreurs doivent être pris en compte pour avoir le message déchiffré. Tout d’abord, il aura besoin d’une variable “result” qui va récupérer le résultat du déchiffrement.

Ensuite, il faut vérifier si dans la note envoyée par Julios César, se trouve que [des lettres et pas des chiffres](https://www.w3schools.com/python/ref_string_isalpha.asp), puis si ce sont des [majuscules](https://www.w3schools.com/python/ref_string_isupper.asp) ou des minuscules. Le tout en lisant la note [ligne par ligne](https://www.w3schools.com/python/python_for_loops.asp).

result = ""

for i in range(len(text)):

char = text[i]

if not char. … (): //Vérifier si ce sont des lettres

result += char

elif char. ...(): // Vérifier si ce sont des majuscules

* 1. Déchiffrer des lettres

Codingtamix se fait interrompre lors de son travail par Maggiramix, grand magicien de la Gaule. Ce dernier lui donne un livre qui lui permettra de comprendre pourquoi les chiffres peuvent aussi être des [lettres](https://www.w3schools.com/charsets/ref_html_ascii.asp).

Le livre lui a permis de mieux comprendre comment aborder sa tâche. Il va falloir ajouter à la suite de la vérification des majuscules, la partie la plus importante, le déchiffrement. Pour cela, on aura besoin d’une variable “[chr](http://programiz.com/python-programming/methods/built-in/chr)()”, mais également des valeurs de [A](https://www.w3schools.com/charsets/ref_html_ascii.asp) et aussi de [a](https://www.w3schools.com/charsets/ref_html_ascii.asp), car dans une phrase on peut retrouver des majuscules et des minuscules.

elif char.isupper():

result += chr((ord(char) + step - ...) % 26 + ...) // Majuscules

else:

result += chr((ord(char) + step - ...) % 26 + ...) // Minuscules

return result

* 1. Les mots vont tomber !

Ça y est, Codingtamix est à deux doigts de découvrir le message. Il va devoir maintenant déchiffrer le message avec la méthode qu’il a mis en place précédemment.

A la suite de ce qu’il a écrit, il va mettre en place, un système de [contrôle](https://www.w3schools.com/python/python_try_except.asp) pour être sûr que le message a bien été déchiffré. Il aura également besoin de plusieurs variables pour la bonne lecture de ce dernier. Il faudra :

* + Une variable qui permet d’[ouvrir](https://www.w3schools.com/python/ref_func_open.asp) le message
  + Une variable qui permet de [lire](https://www.w3schools.com/python/ref_file_read.asp) le message
  + Une variable qui permet d’[écrire](https://www.w3schools.com/python/python_file_write.asp) sur ce message
  + Une variable qui permet de [refermer](https://www.w3schools.com/python/ref_file_close.asp) le tout en sécurité

try:

f = … (sys.argv[1], 'r') // Ouvrir le message

f1 = … ("message-dechiffre.txt", 'w') // Ouvrir le message

step = int(input("Entrez le nombre de décalage des lettres : "))

text = f. ...() // Lire le message

f1. ...(decrypt(text, 26-step)) // Ecrire le message

f. … () // Refermer le message

f1. … () // Refermer le message

print("Le fichier chiffré est disponible sous le nom message-dechiffre.txt")

except Exception:

print("Le fichier est pas existant.")

Cependant, Maggiramix avait laissé une note dans le livre, cette note nous stipule que les réponses sont dans le message, tout est une question de décalage... La note est la suivante :

“ Celui qui signe est aussi chiffré mais si nous le soustrayons alors il sera décalé”.

* 1. 

Stop Julios César

🌟 Bravo ! Nous avons réussi à stopper César ! 🌟

1. L'empire gaulois contre-attaque !
   1. Prenez vos affaires et rentrez chez vous !

Suite à cette découverte, Codingtamix est parti en courant prévenir le chef du village de sa découverte. Tous les villageois ont été appelés à se réunir près du grand chêne pour un conseil général. Lors de la réunion, Codingtamix explique aux gaulois sa découverte sur la future attaque des romains sur leur village.

Il leur propose une petite vengeance contre les envahisseurs pour avoir voulu attaquer le beau village d’Epidorium. Le stratagème est le suivant : renvoyer un message chiffré avec la même méthode mais cette fois-ci avec des fausses directives.

Les nouvelles instructions ont été prises en note par Codingtamix sur un nouveau parchemin “message-jules-cesar.txt”. Il va falloir refaire la même chose pour déchiffrer le message original mais sur un nouveau manuscrit “encrypt.py”, sauf qu’il faut juste changer un élément.

f1.write(encrypt(text, step))

🎊 Félicitations ! Les romains ne viendront plus ! 🎊



Le chef du camp Epidorium appelle ces hommes pour repartir à Rome.

1. Conclusion

Grâce à vous, Codingtamix a réussi à sauver son village gaulois mais également toute l’Europe des envahisseurs romains. Julios César ne reviendra pas sur ce territoire avant longtemps ! C’est ainsi que les irréductibles gaulois du village ont dressé un banquet sous les étoiles pour fêter cette nouvelle victoire.



Banquet des gaulois

Pour aller plus loin, vous pouvez :

* Ajouter une interface graphique en python