B0 - HUB

B-INN-000

Cesar

Atelier de Cryptographie

Cesar

language: python
build tool: / / /

- The totality of your source files, except all useless files (binary, temp files, obj files,...), must be included in your delivery.
- All the bonus files (including a potential specific Makefile) should be in a directory named *bonus*.
- Error messages have to be written on the error output, and the program should then exit with the 84 error code (0 if there is no error).

Bienvenue à ce coding club sur le thème de la cryptographie !
Aujourd'hui, nous allons explorer l'un des algorithmes de cryptographie les plus anciens et les plus simples, le chiffrement de César. Nous allons découvrir comment ce chiffrement fonctionne, comment il peut être implémenté en utilisant un langage de programmation et comment il a été utilisé dans l'histoire pour sécuriser des messages importants.
En fin de compte, nous espérons que cette session de coding club vous donnera une meilleure compréhension de la cryptographie et de la manière dont les algorithmes de chiffrement sont utilisés pour protéger les données sensibles.

Objectifs :
Comprendre les bases de la cryptographie et du chiffrement.
Apprendre à implémenter l'algorithme de chiffrement de César en code.
Développer des compétences en programmation (Python).
Renforcer la logique algorithmique et la résolution de problèmes.

Pour effectuer l'algorithme César vous avez besoin de vous assurer que python est installé sur votre ordinateur, ainsi que Visual Studio Code

Terminal

```
~/B-INN-000> python3 --version
```

Introduction à la cryptographie :

La cryptographie est l'art et la science de sécuriser les communications en convertissant des informations en un format illisible pour quiconque n'ayant pas la clé de déchiffrement appropriée.
Son objectif principal est de garantir la confidentialité, l'intégrité et l'authenticité des données, ainsi que de permettre des opérations sécurisées dans un monde de plus en plus interconnecté.

Fonctionnement du chiffrement Cesar :

Le chiffrement de César est une méthode très simple de cryptographie. Voici comment ça marche :

Choix de la clé (décalage) : Vous choisissez un nombre qui indique combien chaque lettre du message original sera décalée dans l'alphabet. Par exemple, avec un décalage de 3, A devient D, B devient E, et ainsi de suite.
Conversion du message : Prenez le message à chiffrer et appliquez le décalage à chaque lettre. Si le décalage vous fait sortir de l'alphabet, revenez au début. Par exemple, si le décalage est de 3 et que vous arrivez à Z, retournez à C.
Chiffrement : Remplacez chaque lettre du message par la lettre correspondante après le décalage.
Les autres caractères restent inchangés. Message chiffré : Le résultat est le message chiffré, composé des lettres décalées.

Exemple : Si le mot original est "HELLO" et que le décalage est de 3, chaque lettre est déplacée de 3 places, donnant le message chiffré "KHOOR". Il est important de savoir que le chiffrement de César est vulnérable aux attaques par force brute, car il n'y a que 25 décalages possibles. Cela signifie qu'il ne peut pas protéger efficacement les informations secrètes dans le monde moderne. Malgré cela, il est enseigné pour comprendre les bases de la sécurité informatique et de la cryptographie.

Introduction

Le jour d'Halloween, les membres du Coding Club ont décidé de se rendre à un manoir hanté, connu pour les différents trésor qu'il cacherait… Du moins c'est ce qu'on en dit, car personne n'en est jamais revenu.
Witch Garden n'est pas un endroit comme les autres, comme son nom le suggère, des sorcières ont ensorcellé les lieux et après un tour des lieux avec vos camares une seule solution semble s'offrir à vous pour entrer dans le manoir : La Grande Porte.
Mais celle-ci est ensorcellée par un puissant sortilège, et le code magique qui vous empêche de l'ouvrir est bien trop complexe… Est-ce que votre technologie sera suffisante pour vaincre la magie des sorcières ?

Lecture d'un message et decalage

Etape 1 : Lecture, stockage, et affichage d'un message.

Vous venez d'observer d'un coup d'oeil un indice sur le côté de la grande porte, vous devez écrire un programme informatique pour le déchiffrer. Pour cela, vous allez devoir apprendre comment chiffre un message, puis comment
le déchiffrer avec un décalage connu et enfin comment le déchiffrer sans connaitre le décalage.
1. Stocker le message suivant dans une variable.
Message : "Commencez par réussir à afficher ce message !"

2. Afficher le message sur la console.

```
Terminal
~/B-INN-000> python3 cesar_exemple.py "Commencez par réussir à afficher ce message
!"
```

3. Demander à l'utilisateur de saisir un message et le nombre pour le décalage. Stockez les informations dans des variables, puis affichez les informations

Pour demander à l'utilisateur de saisir un message en Python, vous pouvez utiliser la fonction input().

message = input (" Entrez le message a chiffrer : ")
decalage = int (input (" Entrez le decalage : "))
Voici un exemple d'affichage :

```
Terminal
~/B-INN-000> python3 cesar_exemple.py
Saissisez le message que vous souhaitez chiffrer :
Commencez par réussir à afficher ce message !
Saissisez le décalage que vous souhaitez utiliser :
3
Vous avez saisi le message suivant : Commencez par réussir à afficher ce message !
Ce message sera chiffré avec un décalage de 3.
```

ETAPE 2 : CHIFFRER LE MESSAGE.

Félicitations, vous avez réussi à stocker le message et le décalage dans des variables. Maintenant, vous allez devoir chiffrer le message en utilisant le décalage.

1. Parcourir le message lettre par lettre et afficher chaque lettre sur la console en prennant en compte que les caractères alphanumérique.

Pour récupérer les entrées de l'utilisateur au moment de l'éxécution du programme, utilisez argv. Exemple : argv[1] représente "test" dans : ./cesarcode.py "test"

```
Terminal
~/B-INN-000> python cesarcode.py "Bonjour, je suis Marvin, votre robot super
intelligent."
B
o
n
j
o
u
r
j
e
... etc
```

Pour parcourir une chaîne de caractères, vous pouvez utiliser une boucle for ou while en Python et dans la plus part des langages. Pour connaître la longueur d'une chaîne de caractères en Python, vous pouvez utiliser la fonction len().

2. Pour chaque lettre, ajouter le décalage au code ASCII de la lettre.

Vous devez ajouter le décalage au code ASCII de la lettre afin d'obtenir le code ASCII de la lettre chiffrée.

ASCII est un code de caractères qui associe un entier à chaque symbole. Par exemple, le code ASCII de "A" est 65, le code ASCII de "B" est 66, etc. Vous pouvez utiliser la fonction ord() en Python pour obtenir le code ASCII d'un caractère. La table ASCII est disponible sur Wikipedia ou via la commande man ascii sur un terminal Linux.

Exemple avec le message "Bonjour" et un décalage de 3 :
Message original: BONJOUR
Décalage: 3
Message chiffré: ERQMRXU

```
Terminal
~/B-INN-000> python3 cesarcode.py "Bonjour" 3
ERQMRXU
```

3. Gérez les cas des caractères non alphanumériques. Vous devez ignorer les caractères non alphanumériques et les afficher tels quels.

Message original: Hello, World!
Décalage: 3
Message chiffré: Khoor, Zruog!

4. Gérez les cas où le code ASCII dépasse la valeur de 122 (code ASCII de la lettre z). Dans ce cas, vous devez revenir au début de l'alphabet.

Etape 3 : Dechiffrer le message.

Bravo, vous avez réussi à chiffrer le message. Maintenant, vous allez devoir déchiffrer le message en utilisant le décalage.

1. Parcourir le message lettre par lettre et afficher chaque lettre sur la console en prennant en compte que les caractères alphanumérique.

2. Pour chaque lettre, soustraire le décalage au code ASCII de la lettre.
Exemple avec le message “ERQMRXU” et un décalage de 3 :
Message chiffré: Khoor, Zruog!
Décalage: 3
Message original: Hello, World!

Etape 4 : Dechiffrer le message sans connaitre le decalage.

Lorsque vous interceptez un message, vous ne connaissez pas le décalage utilisé pour le chiffrer. Vous allez devoir trouver le décalage en explorant d’autres méthodes. Voici une liste non-exhaustive de méthodes que vous pouvez utiliser pour trouver le décalage :

Fréquence des Lettres : Les langues ont des lettres fréquentes. En analysant la fréquence des lettres dans le message chiffré, vous pourriez repérer des correspondances avec les fréquences typiques de la langue.

Mots Courants : Les mots courants apparaissent régulièrement. En identifiant des mots fréquents comme “the”, “and” ou “is”, vous pourriez déduire le décalage à partir des lettres correspondantes.

Dictionnaires de Mots : Les dictionnaires peuvent aider à trouver des mots reconnus dans le message chiffré. En itérant à travers les mots du dictionnaire, vous pourriez repérer des correspondances.

Approche de Force Brute : Si le décalage est petit, vous pourriez essayer toutes les possibilités de décalage jusqu’à ce que le message ait du sens.

Etape 5 : Tu penses avoir fini ?

Ok ! Maintenant que tu es arrivé à la fin, il est temps de tenter de déchiffrer le message sur cette porte !

“r thi a wtjgt sj zpwddi gsk pj adjcvt p w edjg at vdjitg.”

Alors tu as réussi ? Préviens un Cobra !