

TP2_Guillaume_Sanchez

Expliquer le principe de ROT13

Le **ROT13** est une méthode de chiffrement simple, utilisée principalement pour **obfusquer** (cacher) le texte sans réelle sécurité. Le nom **ROT13** signifie "**rotate by 13 places**", ce qui veut dire "rotation de 13 positions". Le texte est transformé en décalant chaque lettre de l'alphabet de 13 positions.

Réaliser un programme permettant d'implémenter ROT13

Voici mon programme détaillé en python :

```
TP > TP2 > rot13.py > main
1 # TP 2 : Guillaume Sanchez
2
3 def rot13_conversion(message):
4     result = "" # instantiation d'une variable vide qui contiendra le résultat
5     for char in message: # pour tous caractères dans l'argument message
6         if char.isalpha(): # vérification si la chaîne est constituée d'alphabets
7             shift = 13 # instantiation d'une variable qui contient le nombre de décalage
8             base = ord('A') if char.isupper() else ord('a') # instantiation d'une variable qui contient le code ASCII de 'A' ou 'a' selon la casse
9             # Return du calcul du nouveau caractère en appliquant le décalage
10            # ord(char) - base : décalage par rapport à 'A' ou 'a'
11            # (ord(char) - base + shift) % 26 + base: décalage circulaire dans l'alphabet
12            result += chr((ord(char) - base + shift) % 26 + base)
13        else:
14            result += char # Si ce n'est pas un alphabet, on le garde tel quel
15    return result
16
17 def main(): # Fonction Principale
18    print("=== ROT13 : Codage / Décodage ===") # Affichage du nom du programme
19    print("1. Coder un message") # Affichage du choix 1
20    print("2. Décoder un message") # Affichage du choix 2
21    choix = input("Votre choix (1 ou 2) : ").strip() # Récupération de la valeur choisi dans une variable
22
23    if choix not in ['1', '2']: # Gestion d'erreur (si ce n'est pas 1 ou 2)
24        print("Choix invalide. Veuillez entrer 1 ou 2.") # affichage message erreur
25        return
26
27    message = input("Entrez votre message : ") # Récupération du message dans une variable
28    resultat = rot13_conversion(message) # Utilisation de la fonction "rot13_conversion" avec comme argument "message" et récupération du résultat dans une variable
29    print("\nMessage transformé :")
30    print(resultat) # Affichage du résultat de la fonction rot13_conversion.
31
32 if __name__ == "__main__":
33     main()
```

J'ai dans un premier temps créé une fonction nommée "rot13_conversion" qui attend en argument une chaîne de caractères. Cette fonction permet de décaler chaque caractère de 13 places selon sa position dans l'alphabet. La fonction retourne le résultat.

La seconde fonction est la fonction "main" qui est la fonction principale du programme. Elle permet de sélectionner si on veut coder ou décoder en ROT13 et dans un second temps, elle demande le message à décoder. Elle utilise la fonction "rot13_conversion" pour ensuite afficher le résultat de cette dernière.

Exemple avec bonjour :

```

=== ROT13 : Codage / Décodage ===
1. Coder un message
2. Décoder un message
Votre choix (1 ou 2) : 1
Entrez votre message : Bonjour

Message transformé :
Obawbhe

```

```

=== ROT13 : Codage / Décodage ===
1. Coder un message
2. Décoder un message
Votre choix (1 ou 2) : 2
Entrez votre message : Obawbhe

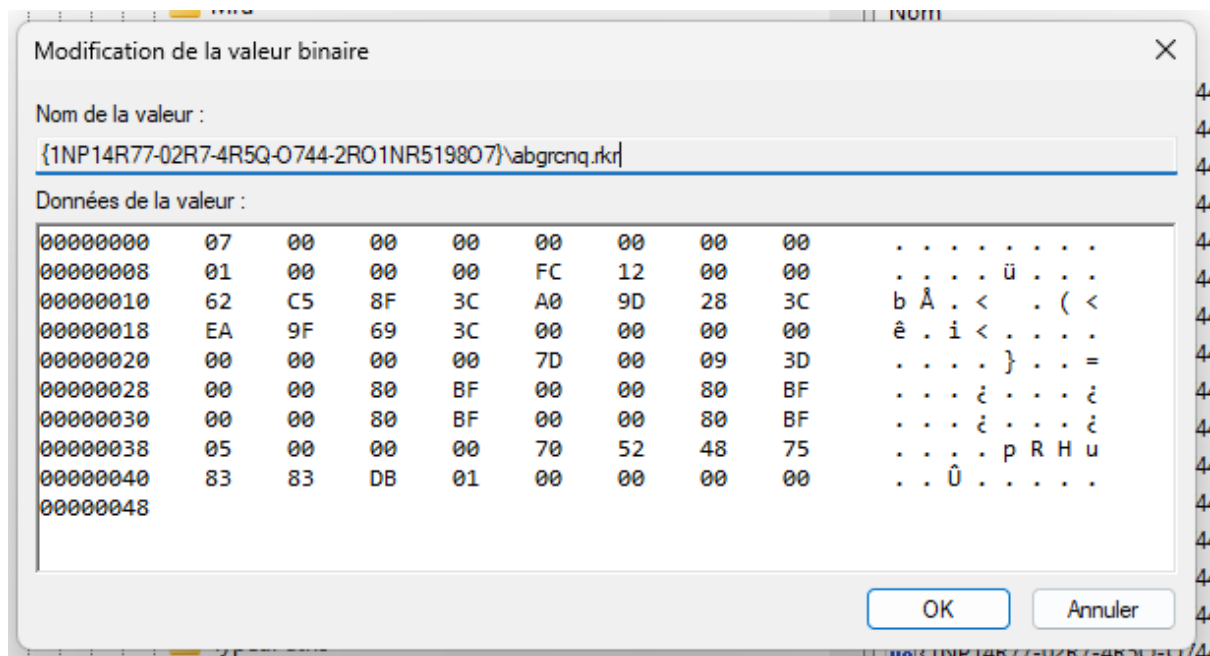
Message transformé :
Bonjour

```

En vrais, le choix du codage n'est pas utile car l'alphabet est composé de 26 caractères.

Décoder une des valeurs de UserAssist

Dans cet exemple, je vais utiliser cette valeur de UserAssist :



Dans ce cas-là "abgrcnq.rkr" est convertible, cela donne après conversion "notepad.exe".

```

=== ROT13 : Codage / Décodage ===
1. Coder un message
2. Décoder un message
Votre choix (1 ou 2) : 2
Entrez votre message : abgrcnq.rkr

Message transformé :
notepad.exe

```