CRY 2020

Laboratoire #1

27-10-2020

1 Préliminaires

Ce laboratoire utilisera le langage de programmation Python3. Vous trouverez un template du code à remplir sur cyberlearn. Vous devez rendre **votre code** le **09.11.2020** à minuit sur cyberlearn. Il vaudra 2.5 pts de la note. Les 2.5 pts restants seront obtenus lors d'un petit test de 15 minutes **à livre fermé** qui aura lieu le **10.11.2020** pendant le cours.

2 Chiffre de César Généralisé

Le but de cette partie consiste à écrire deux routines en Python, nommées respectivement `cesar_encrypt` et `cesar_decrypt`, qui prennent en entrée une chaîne de caractères ainsi qu'un nombre secret de décalages, et qui retournent la version chiffrée et déchiffrée de cette chaîne de caractères, respectivement.

1. Implémenter les routines `cesar_encrypt` et `cesar_decrypt` en Python. Pour simplifier, ignorez les caractères spéciaux, les espaces et les signes de ponctuation. Convertissez aussi toutes les lettres minuscules en majuscules et les lettres accentuées en leur version sans accent.

2.1 Analyse de Fréquence

Le but de cette partie consiste à calculer les statistiques d'apparition des lettres dans un texte français. Pour ce faire, on s'aidera d'un ou de plusieurs textes de référence, que l'on pourra trouver sur le site du projet Gutenberg [1], par exemple.

2. Ecrire une routine `freq_analysis` en Python qui prend en entrée un texte et qui retourne une liste contenant les probabilités d'apparition de chacune des lettres de l'alphabet.
3. Quelles statistiques avez-vous obtenu ?
4. Vos statistiques correspondent-elles à celles que l'on peut trouver sur Internet ?

1. http://www.gutenberg.org

2.2 Cryptanalyse du Chiffre de César Généralisé

Au moyen des statistiques obtenues lors de l'étape précédente, écrire une routine en Python `cesar_break` prenant en entrée un texte chiffré au moyen du chiffre de César généralisé, qui teste tous les décalages possibles, et qui retourne le décalage pour lequel les fréquences des lettres correspond le mieux aux statistiques du français. Pour ce faire, on utilisera le test statistique du χ^2 :

$$\chi^2 = \sum_{i=1}^{26} \frac{(O_i - E_i)^2}{E_i}$$

où O_i représente le nombre observé d'apparitions pour la lettre i, et E_i représente le nombre attendu d'apparitions. Le décalage recherché minimisera donc probablement la valeur de χ^2.

5. Ecrire une routine en Python `cesar_break` prenant en entrée un texte chiffré au moyen du chiffre de César généralisé, qui teste tous les décalages possibles, et qui retourne le décalage pour lequel les fréquences des lettres correspond le mieux aux statistiques du français.

3 Chiffre de Vigenère

De la même manière, le but de cette partie consiste à écrire deux routines en Python, nommées respectivement `vigenere_encrypt` et `vigenere_decrypt`, qui prennent en entrée une chaîne de caractères ainsi qu'un mot-clef secret, et qui retournent la version chiffrée et déchiffrée de cette chaîne de caractères, respectivement.

6. Ecrire les routines `vigenere_encrypt` et `vigenere_decrypt` en Python.

3.1 Indice de Coïncidence

Le but de cette partie est de retrouver la longueur du mot-clef utilisé dans le cadre d'un chiffrement de Vigenère au moyen du calcul de l'indice de coïncidence. L'indice de coïncidence d'un texte de longueur N se calcule de la manière suivante :

$$\mathrm{IC} = \frac{26 \sum_{i=1}^{26} n_i(n_i - 1)}{N(N-1)}.$$

où n_i est le nombre d'occurences de la lettre i dans le texte chiffré (on a donc $\sum_i n_i = N$).

7. Ecrire une routine en Python `coincidence_index` prenant en entrée un texte et permettant de calculer son indice de coïncidence.
8. Expliquer en termes simples quel phénomène mesure l'indice de coïncidence.
9. Quel indice de coïncidence obtenez-vous lorsque calculé sur un texte en français ?
10. Quel indice de coïncidence obtenez-vous lorsque calculé sur un texte aléatoire ?

3.2 Cryptanalyse du Chiffre de Vigenère

Il est finalement temps d'utiliser tous les outils développés jusqu'à ce moment pour obtenir un outil de cryptanalyse du chiffrement de Vigenère entièrement automatisé. Voici une esquisse du processus complet, qui fonctionne selon une approche de type "divide-and-conquer" :

— Trouver la taille du mot-clef au moyen de l'indice de coïncidence.
— Récupérer chaque caractère du mot-clef au moyen d'un test du χ^2.

La première étape fonctionne de la manière suivante : pour chaque taille ℓ de mot-clef possible, en partant de 1, on calcule l'index de coïncidence en prenant comme texte les caractères en positions $1, \ell+1, 2\ell+1$, etc.

11. Calculer l'indice de coïncidence sur le texte chiffré `vigenere.txt` pour $\ell = 1, \ldots, 20$. Quelle est la longueur de mot-clef la plus vraisemblable ?
12. Décrypter complètement le texte chiffré `vigenere.txt` en vous aidant du test du χ^2 et des routines que vous avez programmées au préalable. Pour ceci, complétez la routine `vigenere_break`. Quel était le mot-clef utilisé ?

4 Cryptanalyse d'une Version Améliorée du Chiffre de Vigenère

Un apprenti en cryptographie décide d'améliorer le chiffre de Vigenère. Son raisonnement est le suivant : "le problème avec le chiffre de Vigenère est la réutilisation de la clef". Il décide donc, après chaque utilisation de la clef de la changer en la chiffrant avec le chiffre de César généralisé. Par exemple, si la clef initiale est la clef "MAISON" et la clef du chiffre de César est 2, les six premières lettres du texte clair sont chiffrées avec "MAISON", les suivantes avec "OCKUQP", puis "QEMWSR".

13. Ecrire les routines `vigenere_caesar_encrypt` et `vigenere_caesar_decrypt` en Python qui prennent en entrée une chaine de caractères, un mot-clef initial pour Vigenère ainsi qu'un décalage pour césar et qui chiffre/déchiffre selon cette nouvelle construction.
14. En vous inspirant de l'exercice précédent, cassez le chiffre de Vigenère amélioré en complétant la routine `vigenere_caesar_break`. Récupérez le mot-clef et le décalage utilisés pour chiffrer le texte `vigenereAmeliore.txt` et décryptez-le.