Université Paris 7 - Denis Diderot
Année universitaire 2016-2017

Bastien Masson Vincent Taco
L3 - SI6

PROJET DE CHIFFREMENT PAR SUBSTITUTION

1 Éxécution des applications

1.1 Chiffre.java

1. compiler avec la commande `javac Chiffre.java`
2. exécuter avec la commande `java Chiffre type_chiffrage clé_chiffrage nom_fichier`

`type_chiffrage` correspond à la méthode de chiffrage à utiliser pour chiffrer le texte indiqué.
Entrez la lettre `c` pour le chiffrement de César, la lettre `v` pour le chiffrement de Vigenère et la lettre `p` pour le chiffrement par permutation.

`clé_chiffrage` correspond à la clé utilisée pour chiffrer le texte indiqué.
Entrez la valeur du décalage pour chiffrement de César, le mot clé pour le chiffrement de Vigenère et la permutation pour le chiffrement par permutation.

1.2 Dechiffre.java

1. compiler avec la commande `javac Dechiffre.java`
2. exécuter avec la commande `java Dechiffre type_chiffrage clé_chiffrage nom_fichier`

L'ordre et le type des argument sont les même que pour l'application `Chiffre.java`.

1.3 Decrypt.java

1. compiler avec la commande `javac Decrypt.java`
2. exécuter avec la commande `java Decrypt type_chiffrage nom_fichier [(stratégie ou taille_mot_clé) [mot_texte_clair]]`

Les caractères utilisés pour les types de chiffrages sont le même que que pour les applications `Chiffre.java` et `Dechiffre.java`

`stratégie` correspond aux 3 stratégies (a), (b) et (c) décrites dans l'énnoncé du projet.

`taille_mot_clé` correspond à la taille de la clé utilisée pour le chiffrement de Vigenère.

`mot_text_clair` : correspond au mot connu qui se trouve dans le texte clair.

2 Choix d'algorithmes

Chacune des classes `Decalage, Vigenere, Permutation` comportent 4 méthodes :

```
public char chiffrer(char c, int key)
public char dechiffrer(char c, int key)
public String chiffrer(String s, int key)
public String dechiffrer(String s, int key)
```

2.1 Decalage.java

Dans la méthode `public char chiffrer(char c, int key)` on utilise la clé `key` pour décaler le caractère.
On commence par retrancher 97 (qui correspond au code ASCII de la lettre minuscule a).
Ensuiste on ajoute la valeur du décalage, puis on calcule le modulo 26 du nombre obtenu.
On finit par ajouter 97 pour obtenir le caractère chiffré.

Dans la méthode `public String chiffrer(String s, int key)`, on applique la méthode précédente à tous les caractères de la chaîne passée en argument.
On renvoie la chaîne chiffrée obtenue.

Le principe des méthodes de déchiffrement est similaire.
On la différence est qu'on utilise la soustraction modulo 26 pour retrouver le caractère d'origine.

2.2 Vigenere.java

Les méthodes `chiffrer(char c, char key)` et `dechiffer(char c, char key)` se comportent de la même manière que les méthodes du même nom de la classe Decalage.
La différence est que la clé utilisée pour le chiffrement/déchiffrement est un caractère.

La méthode `public String chiffrer(String s, String key)` applique la méthode `char chiffrer(char c, char key)` à chacun des caractères de la chaînes passée en argument.
Les décalages induits par les caractères du mot-clé sont utilisés cycliquement sur tous les caractères de la chaînes.

Le principe utilisé est le même pour le déchiffrage.

2.3 Permutation.java

Dans la méthode `char chiffrer(char c, String key)` on renvoie le caractère qui se trouve à l'indice (code ASCII du caractère passé en argumen - 97) de la chaîne passé en argument.
Cette chaînes de caractère représente la permutation utilisée pour chiffrer les textes clairs.

Dans la méthode `char dechiffrer(char c, String key)` on renvoie le caractère obtenu en ajoutant 97 à l'indice auquel se trouve le caractère `c` dans la chaîne `key`.

Les méthodes `String chiffrer/dechiffrer(String s, String key)` applique les 2 méthodes précédentes à chaque caractères de la chaîne `s` passée en argument.

2.4 Chiffre.java, Dechiffre.java

Ces applications ouvrent en lecture le fichier passé en argument. Puis elles appellent les fonctions décrites ci-dessus à chaque lignes de ce fichier afin de chiffer/déchiffrer entièrement.

2.5 Decrypt.java

2.5.1 Chiffrement de César

2.5.2 Chiffrement de Vigenère

Dans cette partie, on suppose la longueur n du mot-clé utilisé pour le chiffrement connue.
Les étapes de l'algorithme de décryptage du chiffrement de Vigenère sont :

1. Subdivision du texte chiffré en n sous-chaînes.
   La première sous-chaîne est composé des caractères d'indice $0, 0+n, 0+2n, ...$ du texte chiffré, la deuxième sous-chaîne est composé des caractères d'indice $1, 1+n, 1+2n, ...$ du texte chiffré, ...
   Toutes les sous-chaînes sont stockées dans un tableau.
2. Pour chaque sous-chaîne k on effectue une analyse statistique de ses caractères.
3. À partir de cette analyse statistique on déduit le décalage utilisé pour chiffrer les caractères de cette sous-chaîne.
4. De ce décalage on déduit le caractère d'indice $k-1$ du mot-clé
5. À partir de ces caractères on reconstitue le mot- clé utilisé pour chiffrer le texte clair

2.5.3 Chiffrement par permutation

On effectue une analyse statistique des caractères du texte chiffré.
On compare les fréquences de chaque caractères du texte chiffré, aux fréquences des lettres de la langue française.
On associe les fréquences des les plus élevées des caractères du texte chiffré, aux fréquences les plus élevées des lettres de la langue Française. Idem pour les fréquences les plus basses.
De ces association on déduit la permutation utilisée pour chiffrer le texte clair.
Cette méthode ne permet qu'un decryptage partiel du texte chiffré.