

Cryptographie

Léa MENERET, Fathima SAHADATTALY, Ulrike KULZER

22 décembre 2017

Table des matières

1 Guide utilisateur	2
1.1 Introduction	2
1.1.1 En général	2
1.1.2 Méthodes de cryptage	2
1.2 Fonctionnement du programme	4
1.2.1 Accueil et choix de la langue	4
1.2.2 Crypter/Décrypter	5
1.2.3 Choix du principe	6
2 Guide de maintenance	10
2.1 Fonctionnalité	10
2.1.1 En général	10
2.1.2 Différents états du programme	10
2.1.3 En détail et coupé en modules	11
2.2 Structure du programme	12
2.2.1 Explications des modules, constantes et fonctions	13
3 Rapport sur les écarts entre objectif et réalisation	22
4 Annexe : Code du programme	22

1 Guide utilisateur

1.1 Introduction

1.1.1 En général

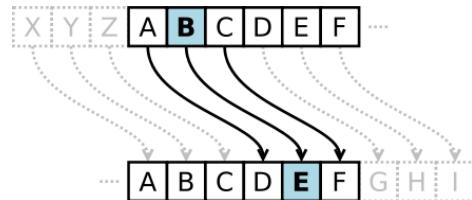
La cryptographie est une technique utilisée pour rendre incompréhensible à autrui un message entre un expéditeur et un destinataire. Ce procédé a notamment été utilisé en période de guerre pour permettre des attaques surprises. Le principe est le suivant : L'expéditeur à partir d'une clé crypte son message et l'envoie au destinataire. Celui-ci possède aussi la clé qui va lui permettre ainsi de décrypter le message.

1.1.2 Méthodes de cryptage

La cryptographie est utilisée depuis l'Antiquité mais certaines de ces méthodes les plus abouies datent du 20e siècle. Il existe différents principes de cryptage plus ou moins compliqués tels que

— le chiffre de César :

Ce procédé a été inventé lors de l'époque romaine par Jules César pour ses communications secrètes. En décalant l'alphabet par un entier donné chaque lettre est associée à une nouvelle lettre, ainsi on peut crypter le message initiale en remplaçant chaque lettre par la nouvelle lettre attribuée.



— le chiffre de Vigenère :

Il a été inventé au 16e siècle par Blaise de Vigenère et est basé sur le tableau à droite. Une clé (un mot) est répétée et mis sous le message et de cette manière on peut trouver les lettres correspondantes à partir du tableau.

Exemple :

		Lettre en clair																									
		26 lettres chiffrées																									
		A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z		
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z			
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z				
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z					
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z						
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z							
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z								
I	I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z								
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z										
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z											
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z												
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z													
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z														
O	O	P	Q	R	S	T	U	V	W	X	Y	Z															
P	P	Q	R	S	T	U	V	W	X	Y	Z																
Q	Q	R	S	T	U	V	W	X	Y	Z																	
R	R	S	T	U	V	W	X	Y	Z																		
S	S	T	U	V	W	X	Y	Z																			
T	T	U	V	W	X	Y	Z																				
U	U	V	W	X	Y	Z																					
V	V	W	X	Y	Z																						
W	W	X	Y	Z																							
X	X	Y	Z																								
Y	Y	Z																									
Z	Z																										

clé : musique
Texte : J'adore écouter la radio toute la journée.
Texte en claire et en dessous la clé répétée:

j'adore écouter la radio toute la journée.
M USIQU EMUSIQU EM USIQU EMUSI QU EMUSIQU
^ ^ ^ ^

Colonne 0, ligne I : on obtient la lettre w.
Colonne D, ligne S : on obtient la lettre v.
Colonne A, ligne U : on obtient la lettre u.
Colonne J, ligne M : on obtient la lettre v.

— *celui de la machine Enigma :*

L'Enigma est une machine de cryptographie inventée par Arthur Scherbius en 1919. Elle a été utilisée durant la Seconde Guerre mondiale pour la communication secrète entre les différentes unités de l'armée allemande. La machine est constituée de cinq rotors dont un réflecteur, d'un clavier, d'un tableau de permutation et de lampes pour chaque lettre. Pour l'allumer il faut une batterie de 4,5 Volt. Le principe est simple : Lorsqu'on appuie sur une lettre du clavier, un courant électrique va être envoyé au tableau de permutation dans lequel la lettre entrée est échangée avec une autre lettre si elles sont connectées. Puis il passera la première fois par les quatre rotors : Dans chacun des trois rotors au milieu il y a un décalage des lettres qui s'opère. À la fin les lettres sont permutees encore une fois dans le réflecteur qui les renvoie par les rotors au tableau de permutation ce qui permettra à une lampe correspondant à une lettre de s'allumer. Ainsi pour chaque lettre on relève la lettre codée, on obtient alors notre message crypté.



Sites de référence :

- <https://fr.wikipedia.org/wiki/Cryptographie>
- <http://www.bibmath.net/crypto/>
- https://fr.wikipedia.org/wiki/Chiffrement_par_d%C3%A9calage
- https://fr.wikipedia.org/wiki/Chiffre_de_Vigen%C3%A8re
- [https://fr.wikipedia.org/wiki/Enigma_\(machine\)](https://fr.wikipedia.org/wiki/Enigma_(machine))
- photo d'Enigma : Von William Warby from London, England - Enigma, CC BY 2.0, <https://commons.wikimedia.org/w/index.php?curid=46848023>

1.2 Fonctionnement du programme

1.2.1 Accueil et choix de la langue

Pour exécuter notre programme, il est nécessaire de l'ouvrir avec le logiciel Pycharm, d'aller dans le fichier "main_module" et d'appuyer sur "run" - "run" - "run main_module". Les instructions du programme seront affichées dans la console qui s'ouvre en bas de l'interface comme montrés ci dessous :

```
Hello and welcome to the cryptography program  
Please select your language:  
'f' for Français, 'e' for English.
```

Fig. 1 – Écran d'accueil

Vous venez d'entrer dans le programme Cryptographie, vous devez entrer la lettre ' f ' pour avoir les instructions en français et ' e ' pour les avoir en anglais. Si la lettre entrée est ' e ', les prochaines instructions seront donc en anglais.

Note : Vous retrouverez les mêmes interfaces en français.

Une nouvelle fenêtre va ensuite apparaître :

```
What do you want to do - encrypt or decrypt a message?  
Enter 'c' for encrypting or 'd' for decrypting.  
Enter 's' to change settings(language)  
or 'q' to exit :(
```

Fig. 2 – Menu principal

Le programme demande maintenant ce que vous voulez faire, vous devez entrer ' c ' si vous voulez que le programme crypte votre texte ou ' d ' si vous voulez qu'il le décrypte. Si vous voulez changer de langue entrez ' s ' et si malheureusement vous voulez quitter le programme entrez ' q ' .

Si la lettre entrée est ' s ' , une nouvelle fenêtre permettant de changer la langue va alors apparaître.

```
Please select your language:  
'f' for Français,    'e' for English.
```

Fig. 3 – Paramètres

Vous devez entrer la lettre ' f ' pour avoir les instructions en français et ' e ' pour les avoir en anglais.

Si la lettre entrée est ' f ' les prochaines instructions seront donc en français.
Vous reviendrez sur les instructions précédentes cette fois-ci en français.

1.2.2 Crypter/Décrypter

```
Bonjour et bienvenue sur le programme de cryptographie.  
Que voulez vous faire-crypter ou décrypter un message?  
Insérez 'c' pour crypter ou 'd' pour décrypter.  
Insérez 's' pour changer les paramètres(langue)  
ou 'q' pour quitter le programme :(
```

Fig. 4 – Menu principal

Vous devez entrer ' c ' si vous voulez que le programme crypte votre texte ou ' d ' si vous voulez qu'il le décrypte.

Si vous voulez changer de langue entrez ' s ' et si malheureusement vous voulez quitter le programme entrez ' q ' .

Si la lettre entrée est ' c ' , le programme va donc être servi pour le cryptage d'un texte.
Il existe trois méthodes par lesquelles le texte peut être crypté, nous allons donc détailler les différentes manières.

1.2.3 Choix du principe

Si vous avez choisi de crypter votre texte, il vous faut maintenant choisir quelle méthode vous voulez utiliser, trois choix s'offrent à vous.

De quelle manière voulez-vous crypter votre texte?

Insérez 'c' pour le cryptage par la méthode du chiffre de César, insérez 'v' pour le cryptage par la méthode du chiffre de Vigenère ou insérez 'e' pour le cryptage par la méthode de la machine Enigma.

Insérez 'h' pour ouvrir l'aide dans laquelle il y a plus d'informations sur les différents principes.

Insérez 'm' pour retourner au menu principal.

Fig. 5 – Cryptage

Vous devez entrer ' c ' si vous voulez que le programme crypte votre texte par la méthode du chiffre de César, ' v ' par la méthode du chiffre de Vigenère et enfin ' e ' par la méthode de la machine Enigma.

Si tout autre fois vous voulez retourner au menu principal, entrez ' m '.

De la même manière, si à l'interface précédente vous aviez choisi le mode décryptage, vous allez devoir choisir la méthode par laquelle vous voulez déchiffrer votre texte. Les commandes pour choisir la méthode de déchiffrement sont alors les mêmes que pour un texte à crypter : ' c ' pour César, ' v ' pour Vigenère, ' e ' pour Enigma et ' m ' pour retourner au menu principal.

De quelle manière voulez-vous déchiffrer votre texte?

Insérez 'c' pour le cryptage par la méthode du chiffre de César, insérez 'v' pour le cryptage par la méthode du chiffre de Vigenère ou insérez 'e' pour le cryptage par la méthode de la machine Enigma.

Insérez 'h' pour ouvrir l'aide dans laquelle il y a plus d'informations sur les différents principes.

Insérez 'm' pour retourner au menu principal.

Fig. 6 – Déchiffrement

Dans les deux cas vous pouvez consulter l'aide en entrant ' h ' qui vous rappellera les principes de cryptage utilisé pour les différentes méthodes César, Vigenère ou Enigma :

***** AIDE *****

Voici les explications pour les différents principes de (dé)cryptage.

- Le chiffre de César :

Ce procédé a été inventé lors de l'époque romaine par Jules César pour ses communications secrètes. En décalant l'alphabet par un entier donné chaque lettre est associée à une nouvelle lettre, ainsi on peut crypter le message initiale en remplaçant chaque lettre par la nouvelle lettre attribuée.

- Le chiffre de Vigenère :

Il a été inventé au 16e siècle par Blaise de Vigenère et est basé sur le tableau de Vigenère (tableau avec deux fois l'alphabet). Une clé (un mot) est répétée et mise sous le message et de cette manière on peut trouver les lettres correspondantes à partir du tableau.

- Le principe de la machine Enigma :

L'Enigma est une machine de cryptographie inventée par Arthur Scherbius en 1919. Elle a été utilisée durant la Seconde Guerre mondiale pour la communication secrète entre les différentes unités de l'armée allemande.

La machine est constituée de cinq rotors dont un réflecteur, d'un clavier, d'un tableau de permutation et de lampes pour chaque lettre.

Lorsqu'on appuie sur une lettre du clavier, un courant électrique va être envoyé au tableau de permutation dans lequel la lettre entrée est échangée avec une autre lettre si elles sont connectées. Puis il passera la première fois par les quatre rotors : Dans chacun des trois rotors au milieu il y a un décalage des lettres qui s'opère. À la fin les lettres sont permutees encore une fois dans le réflecteur qui les renvoie par les rotors au tableau de permutation ce qui permettra à une lampe correspondant à une lettre de s'allumer. Ainsi pour chaque lettre on relève la lettre codée, on obtient alors notre message crypté.

Insérez 'm' pour retourner au menu principal.

Fig. 7 – Aide

Le chiffre de César :

Imaginons que vous ayez choisi de crypter ou décrypter votre texte par la méthode du chiffre de César, l'interface suivante apparaîtra :

Vous avez choisi <<César>>:

Insérez votre clé (un nombre compris entre 1 et 25);

Insérez 'm' pour retourner au menu principal.

Fig. 8 – Le chiffre de César

Vous devez maintenant choisir la clé à utiliser pour (dé)crypter votre texte. Cette clé doit être un nombre compris entre 1 et 25 et correspondra au décalage de l'alphabet lors du (dé)cryptage de chaque lettre. Vous avez également la possibilité de retourner au menu principal en entrant 'm'.

Le chiffre de Vigenère :

Si votre choix se porte sur le (dé)cryptage par la méthode du chiffre de Vigenère, vous verrez alors le message suivant :

```
Vous avez choisi <<Vigenère>>:  
Insérez votre clé (un mot).  
  
Insérez 'm' pour retourner au menu principal.
```

Fig. 9 – Le chiffre de Vigenère

Vous devez maintenant entrer la clé que vous souhaitez utiliser. Cette clé doit être un mot composé uniquement de lettres (minuscules ou majuscules) sans espace ni tiret. Vous pouvez aussi entrer ' m ' si vous voulez retourner au menu principal.

La machine Enigma :

Si vous avez sélectionné le (dé)cryptage par la méthode de la machine Enigma, le message suivant apparaîtra :

```
Vous avez choisi <<Enigma>>:  
Insérez votre clé (composée de trois lettres enmajuscules).  
  
Insérez 'm' pour retourner au menu principal.
```

Fig. 10 – Le cryptage par la machine Enigma

Le programme attend de vous que vous entriez la clé de (dé)cryptage. Cette clé doit se présenter sous la forme de trois lettres majuscules.

Vous avez toujours, comme précédemment la possibilité de retourner au menu principal en entrant ' m '.

Quelle que soit la méthode de (dé)cryptage choisie vous serez confronté au message suivant qui vous indique d'entrer le texte à traiter. Notez que, dans le texte, tous les caractères spéciaux seront normalisés (c'est-à-dire les accents seront enlevés) et les chiffres et signes de ponctuation seront enlevés.

À ce stade vous pouvez aussi entrer ' m ' pour retourner au menu principal.

```
Insérez votre texte.  
  
Insérez 'm' pour retourner au menu principal.
```

Fig. 11 – Entrée du texte

Enfin le programme s'exécute pour effectuer l'opération que vous lui avez demandé, et l'un des message suivant apparaîtra suivi de votre texte crypté ou décrypté.

Voici votre texte crypté:

Fig. 12 – Affichage du texte crypté

Voici votre texte décrypté:

Fig. 13 – Affichage du texte décrypté

Sous le texte (dé)crypté il s'affichera la phrase suivante :

Insérez 'm' pour retourner au menu principal.

Fig. 14 – Affichage du texte crypté

Il faut insérer ' m ' pour retourner au menu principal.

À partir de cela vous pouvez choisir, si vous voulez continuer ou quitter le programme (voir 4).

Si ensuite la lettre entrée est ' q ', le message de fin s'affichera :

Merci d'avoir utilisé notre programme.

Bonne journée, au revoir.

Fig. 15 – Sortie du programme

2 Guide de maintenance

2.1 Fonctionnalité

2.1.1 En général

- Quand l'utilisateur lance le programme on lui demande de choisir la langue, soit français, soit anglais.
 - Notre programme va proposer à l'utilisateur trois façons différentes de (dé)crypter un texte de complexité croissante et demander une clé.
 - Les différentes manières sont le principe du chiffre de César, du chiffre de Vigenère et de la machine Enigma.

2.1.2 Différents états du programme

Les différents états du programme sont représentés dans le diagramme ci-dessous. Dès que le programme est lancé, il est généralement toujours dans un état en attendant des saisies de l'utilisateur. La seule exception est l'état en cryptant ou décryptant. Sur les flèches on peut voir ce qu'il faut faire pour accéder à un autre état, soit le précédent, soit le suivant. Avant que le programme est fermé, il affichera un petit message de remerciement.

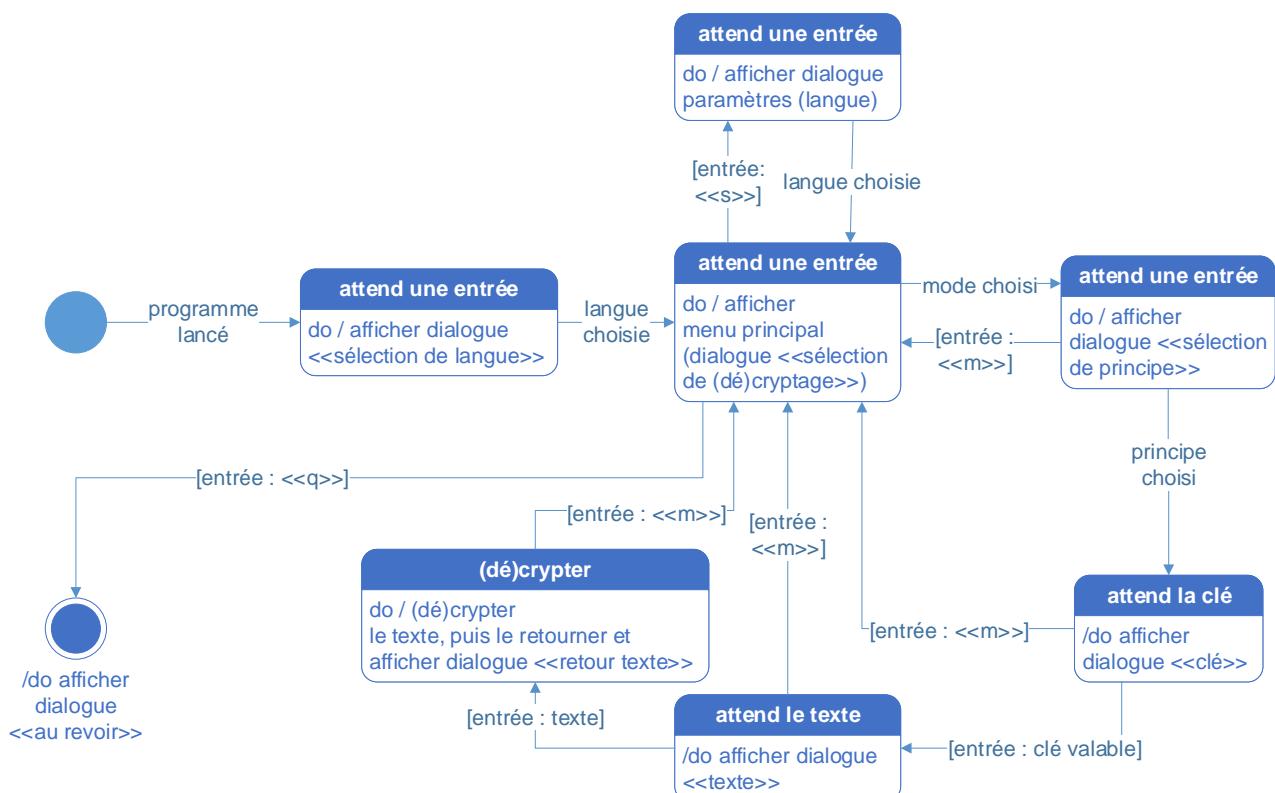


Fig. 16 – Diagramme états-transitions du programme

2.1.3 En détail et coupé en modules

- *Avant de commencer le (de)cryptage :*

On va créer une méthode qui formatera le texte (enlever les accents, les caractères spéciaux, les signes de ponctuation et les espaces, puis mettre tout en majuscule).

- *(Dé)Cryptage César :*

On compte transformer les lettres du texte en nombres par le système unicode, ensuite on additionne la clé (nombre) aux nombres obtenus par le système. On reconvertis alors les lettres en caractères et retourne le texte. Pour le décryptage, il suffit de soustraire au lieu d'additionner.

- *(Dé)Cryptage Vigenère :*

Dans un premier temps on crée une matrice de 26 x 26 lettres contenant l'alphabet représentant le tableau de Vigenère et une matrice à deux lignes pour le texte et la clé. Dans un deuxième temps on prend le texte et on insère tour à tour les caractères individuels dans la première ligne de la matrice et la clé dans la deuxième. Chaque lettre du texte doit être attribuée à un caractère de la clé (mot) ce qui est permis en répétant le mot tant qu'ils restent des lettres du texte. Le cryptage est fait caractère par un caractère. Pour trouver la lettre cryptée on regarde en premier la lettre du texte et on cherche la colonne de la matrice qui appartient à la lettre et on la mémorise. Par la suite on considère le caractère de la clé auquel la lettre du texte est attribuée et on cherche la ligne de la matrice qui appartient au caractère. Dès qu'on a trouvé les deux, la lettre de la matrice qui est enregistré sur cette case est la lettre cryptée laquelle est mémorisée dans une chaîne de caractère. Lorsqu'on a crypté toutes les lettres du texte, on retourne la chaîne de caractère qui est le texte crypté. Pour le décryptage, on possède la matrice à deux lignes avec le texte crypté sur une ligne et la clé répétée sur l'autre. Pour chaque lettre du texte crypté, on parcourt dans le tableau de Vigenère (matrice) la ligne correspondant à la lettre de la clé répétée et lorsqu'on trouve la lettre cryptée cherchée on remonte la ligne pour obtenir le caractère décrypté correspondant à cette lettre. De la même manière que pour le cryptage on mémorise les lettres obtenues dans une chaîne de caractère ce qui permet d'avoir au final le message décrypté.

- *(Dé)Cryptage Enigma :*

Pour commencer nous créons une liste comportant les 26 lettres de l'alphabet. Nous créons ensuite une fonction qui recherche l'indice de chaque lettre du texte dans la liste. Une fois cet indice trouvé une nouvelle fonction lui appliquera un décalage, la lettre de départ sera donc échangée avec une nouvelle lettre.

Pour les connecteurs, il y en aura dix ainsi 20 lettres seront connectées entre elles et six qui ne le seront pas. Nous allons créer une fonction qui, si la lettre est connectée, va retourner la lettre associée et si elle ne l'est pas renvoie la lettre de départ.

On appellera ensuite des fonctions qui exécuteront les décalages prédéfinis (décalages historiques de la vraie machine) au niveau des trois rotors et après on utilisera la fonction correspondant au réflecteur pour permuter les lettres. On repassera finalement à nouveau par les rotors et on obtiendra une certaine lettre. Si la lettre est connectée à une autre, on retournera l'autre lettre, sinon on gardera la même, celle-ci sera alors la lettre (dé)cryptée.

2.2 Structure du programme



Fig. 17 – Structure du programme en modules

2.2.1 Explications des modules, constantes et fonctions

Explications du module *screen_constants* :

Ce module contient toutes les constantes nécessaires pour le screen_module, c'est-à-dire tous les textes de l'interface utilisateur.

- **START_ASK_LANGUAGE** :
dialogue de démarrage (voir fig. 1)
- **LANGUAGE_SETTINGS** :
dialogue des paramètres de langue (voir fig. 3)
- **ENGLISH_MAIN_MENU** et **FRENCH_MAIN_MENU** :
menu principal (voir fig. 2 et 4)
- **ENGLISH_PRINCIPLES_ENCRYPTING** et
FRENCH_PRINCIPLES_ENCRYPTING :
dialogue de sélection du principe de cryptage (voir fig. 5)
- **ENGLISH_PRINCIPLES_DECRYPTING** et
FRENCH_PRINCIPLES_DECRYPTING :
dialogue de sélection du principe de décryptage (voir fig. 6)
- **ENGLISH_ASK_KEY_CAESAR** et **FRENCH_ASK_KEY_CAESAR** :
dialogue de demande de la clé pour (dé)crypter par César (voir fig. 8)
- **ENGLISH_ASK_KEY_VIGENERE** et **FRENCH_ASK_KEY_VIGENERE** :
dialogue de demande de la clé pour (dé)crypter par Vigenère (voir fig. 9)
- **ENGLISH_ASK_KEY_ENIGMA** et **FRENCH_ASK_KEY_ENIGMA** :
dialogue de demande de la clé pour (dé)crypter par Enigma (voir fig. 10)
- **ENGLISH_ASK_TEXT** et **FRENCH_ASK_TEXT** :
dialogue de demande du text à (dé)crypter (voir fig. 11)
- **ENGLISH_ENCRYPTED_TEXT** et **FRENCH_ENCRYPTED_TEXT** :
dialogue du texte crypté (voir fig. 12)
- **ENGLISH_DECRYPTED_TEXT** et **FRENCH_DECRYPTED_TEXT** :
dialogue du texte décrypté (voir fig. 13)
- **ENGLISH_CONTINUE** et **FRENCH_CONTINUE** :
dialogue du pour continuer après avoir reçu le text (dé)crypté (voir fig. 14)
- **ENGLISH_HELP_PRINCIPLES** et **FRENCH_HELP_PRINCIPLES** :
dialogue d'aide (voir fig. 7)
- **ENGLISH_QUIT_MESSAGE** et **FRENCH_QUIT_MESSAGE** :
message de sortie (voir fig. 15)

Explications du module *screen_module* :

Ce module est responsable de l'interface utilisateur, c'est-à-dire il montre le bon texte dans la langue choisie.

show_start_ask_language (ligne 8)

Déclaration	show_start_ask_language()
Rôle	demande à l'utilisateur la langue qu'il veut utiliser au début du programme
Paramètre	aucun
Retour	la chaîne de caractère choisie par l'utilisateur : ‘ f ’ pour le français ‘ e ’ pour l'anglais
Appelé par	run()
Appelle	aucun

show_language_settings (ligne 18)

Déclaration	show_language_settings()
Rôle	demande à l'utilisateur la langue qu'il veut utiliser au cours du programme
Paramètre	aucun
Retour	la chaîne de caractère choisie par l'utilisateur : ‘ f ’ pour le français ‘ e ’ pour l'anglais
Appelé par	run()
Appelle	aucun

show_main_menu (ligne 28)

Déclaration	show_main_menu/english)
Rôle	demande à l'utilisateur ce qu'il veut utiliser la fonction cryptage ou décryptage du programme
Paramètre	english : booléen True si la langue choisie est l'anglais False si la langue choisie est le français
Retour	la chaîne de caractère choisie par l'utilisateur : ‘c’ pour cryptage ‘d’ pour décryptage ‘s’ pour changer de langue ‘q’ pour quitter le programme
Appelé par	run() (attention à vérifier)
Appelle	aucun

show_principles (ligne 50)
 Déclaration show_principles/english, encrypting
 Rôle demande à l'utilisateur la méthode qu'il veut utiliser pour (dé)crypter
 Paramètre english : booléen
 True si la langue choisie est l'anglais
 False si la langue choisie est le français
 encryption : booléen
 True si l'utilisateur veut crypter
 False s'il veut décrypter
 Retour la chaîne de caractère choisie par l'utilisateur :
 'c' pour la méthode César
 'v' pour la méthode Vigenère
 'e' pour la méthode de la machine Enigma
 'm' pour retourner au menu principal
 Appelé par run() (attention à vérifier)
 Appelle aucun

show_ask_key (ligne 81)
 Déclaration show_ask_key/english, principle
 Rôle demande à l'utilisateur la clé pour la méthode qu'il veut utiliser pour (dé)crypter
 Paramètre english : booléen
 True si la langue choisie est l'anglais
 False si la langue choisie est le français
 principle : char
 ' c ', ' v ' ou ' e ' selon la méthode de (dé)cryptage qu'il a choisi la clé ou
 ' m ' pour retourner au menu principal
 Retour run() (attention à vérifier)
 Appelé par aucun
 Appelle

show_ask_text (ligne 114)
 Déclaration show_ask_text/english
 Rôle demande à l'utilisateur le texte qu'il veut (dé)crypter
 Paramètre english : booléen
 True si la langue choisie est l'anglais
 False si la langue choisie est le français
 Retour la clé ou
 'm' pour retourner au menu principal
 Appelé par run() (attention à vérifier)
 Appelle aucun

show_treated_text	(ligne 133) show_treated_text/english, encrypting retourne le message d'affichage du texte (dé)crypté english : booléen True si la langue choisie est l'anglais False si la langue choisie est le français encryption : booléen True si l'utilisateur veut crypter False s'il veut décrypter le message d'affichage du texte (dé)crypté run() (attention à vérifier)
Retour	aucun
Appelé par	
Appelle	
show_help_principles	(ligne 166) show_help_principles/english Affiche une aide dans laquelle le principe de chaque méthode est expliquée demande la saisie de ' m ' pour retourner au menu principal english : booléen True si la langue choisie est l'anglais False si la langue choisie est le français ' m ' run() (attention à vérifier)
Retour	aucun
Appelé par	
Appelle	
show_quit_message	(ligne 184) show_quit_message/english Affiche un message de fin d'utilisation du programme english : booléen True si la langue choisie est l'anglais False si la langue choisie est le français aucun run() (attention à vérifier)
Retour	aucun
Appelé par	
Appelle	

Explications du module *text_module* :

Ce module contient toutes les fonctions avec lesquelles le texte de l'utilisateur peut-être traité.

César

encrypt_caesar	(ligne 7)
Déclaration	encrypt_caesar(text, key)
Rôle	crypte le texte selon le principe de César
Paramètre	text : le texte de l'utilisateur à crypter key : la clé de l'utilisateur
Retour	char : le texte crypté
Appelé par	run() (attention à vérifier)
Appelle	aucun

decrypt_caesar	(ligne 25)
Déclaration	decrypt_caesar(text, key)
Rôle	crypte le texte selon le principe de César
Paramètre	text : le texte de l'utilisateur à décrypter key : la clé de l'utilisateur
Retour	char : le texte crypté
Appelé par	run() (attention à vérifier)
Appelle	aucun

Vigenère

create_vigenere_table	(ligne 47)
Déclaration	create_vigenere_table()
Rôle	crée un tableau modélisant le tableau de Vigenère
Paramètre	aucun
Retour	le tableau
Appelé par	encrypt_vigenere(text, key) decrypt_vigenere(text, key)
Appelle	aucun

create_table_text_key	(ligne 60)
Déclaration	create_table_text_key(text, key)
Rôle	crée un tableau avec deux lignes, la première contenant le texte formaté et la deuxième la clé répétée (même longueur que le texte)
Paramètre	text : le texte de l'utilisateur à (dé)crypter key : la clé de l'utilisateur
Retour	le tableau
Appelé par	encrypt_vigenere(text, key) decrypt_vigenere(text, key)
Appelle	aucun

encrypt_vigenere

Déclaration (ligne 82)
Rôle encrypt_vigenere(text, key)
Paramètre crypte le texte selon le principe de Vigenère
text : le texte de l'utilisateur à crypter
key : la clé de l'utilisateur
Retour char : le texte crypté
Appelé par run() (attention à vérifier)
Appelle create_table_text_key

decrypt_vigenere

Déclaration (ligne 103)
Rôle decrypt_vigenere(text, key)
Paramètre décrypte le texte selon le principe de Vigenère
text : le texte de l'utilisateur à décrypter
key : la clé de l'utilisateur
Retour char : le texte crypté
Appelé par run() (attention à vérifier)
Appelle create_table_text_key

Enigma**create_initial_list**

Déclaration (ligne 130)
Rôle create_initial_list()
Paramètre crée une liste avec les lettres de l'alphabet
aucun
Retour initial_list : la liste remplie avec les lettres de l'alphabet
Appelé par enigma(text, key)
Appelle aucun

search_index

Déclaration (ligne 139)
Rôle search_index(initial_list, letter)
Paramètre cherche l'indice d'une lettre donnée dans une liste de lettres donnée
initial_list : liste dans laquelle l'indice est cherché
letter : lettre donnée de laquelle on cherche l'indice
Retour int : indice de la lettre trouvé
Appelé par enigma(text, key)
Appelle aucun

plugboard	(ligne 167)
Déclaration	plugboard(letter)
Rôle	modélise le tableau de permutation : dix lettres sont associées à dix autres lettres. Si la lettre donnée est associée à une autre lettre, celle-ci va être retournée.
Paramètre	Il y a six lettres qui ne sont associées à aucune autre lettre.
Retour	letter : la lettre donnée qui va être échangée si elle est liée à une autre lettre
Appelé par	char : lettre correspondante à la lettre entrée ou la lettre elle-même
Appelle	enigma(text, key)
	aucun
permutation_reflector	(ligne 225)
Déclaration	permutation_reflector(letter)
Rôle	modélise le réflecteur :
Paramètre	même principe que plugboard(letter), mais cette fois-ci toutes les lettres sont associées à une autre lettre
Retour	letter : la lettre donnée qui va être échangée
Appelé par	char : lettre correspondante à la lettre entrée
Appelle	enigma(text, key)
	aucun
shift_first_rotor	(ligne 292)
Déclaration	shift_first_rotor(return_path, index, offset)
Rôle	modélise le premier rotor, exécute le décalage des lettres par indices et retourne la lettre correspondante
Paramètre	return_path : booléen pour indiquer si c'est le chemin de retour après le réflecteur
Retour	index : indice de la lettre donnée
Appelé par	offset : indique le décalage dans le rotor
Appelle	char : lettre échangée par le décalage du rotor
	enigma(text, key)
	aucun

shift_secondRotor	(ligne 327) shift_secondRotor(return_path, index, offset) modélise le deuxième rotor, exécute le décalage des lettres par indices et retourne la lettre correspondante return_path : booléen pour indiquer si c'est le chemin de retour après le réflecteur index : indice de la lettre donnée offset : indique le décalage dans le rotor char : lettre échangée par le décalage du rotor enigma(text, key)
Retour	
Appelé par	
Appelle	aucun
 shift_thirdRotor	
Déclaration	
Rôle	
Paramètre	
Retour	
Appelé par	
Appelle	
 enigma	
Déclaration	(ligne 398) enigma(text, key)
Rôle	(dé)crypte le texte formaté par le principe de la machine Enigma
Paramètre	text : le texte de l'utilisateur à traiter key : la clé donnée par l'utilisateur pour le traitement
Retour	string : texte traité ((dé)crypté)
Appelé par	run()
Appelle	toutes les fonctions d'aide d'Enigma (voir code du text module)

Explications du module *main_module* :

Ce module est responsable du déroulement du programme. De plus il s'occupe du formatage du texte de l'utilisateur pour qu'il puisse être (dé)crypté par la suite.

Pour l'utilisation de ces fonctions il est nécessaire d'importer préalablement les fichiers "screen_module" et "text_module".

normalise_letter (ligne 10)

Déclaration normalise_letter(x)

Rôle normalise la lettre donné si c'est un caractère spécial
(ex : lettre avec accent) ou retourne la lettre donnée
si celle-ci est déjà normalisé

Paramètre x : représente la lettre donnée en argument

Retour char : une lettre

Appelé par format_texte (texte)

Appelle aucun

format_text (ligne 43)

Déclaration format_text(text)

Rôle formate le texte, c'est-à-dire le texte donné est mis en majuscules ;
les virgules, espaces, chiffres, symboles, etc.
sont enlevés et les caractères spéciaux sont normalisés

Paramètre text : texte donné en entrée par l'utilisateur et qui doit être formaté
pour ensuite pouvoir être crypté ou décrypté

Retour string : le texte formaté

Appelé par run()

Appelle normalise_letter(x)

run (ligne 69)

Déclaration run()

Rôle démarre le programme et contrôle son déroulement

Paramètre aucun

Retour none

Appelé par aucun

Appelle toutes les fonctions de screen_module

format_text(text)

les fonction de traitement de texte ((dé)cryptage)

dans text_module

3 Rapport sur les écarts entre objectif et réalisation

Nous comptions réaliser une interface graphique, ce que nous n'avons pas concrétisé faute de temps et de connaissance dans ce domaine. Nous voulions également pouvoir retourner au menu précédent à chaque interface mais cela s'est avéré impossible.

Concernant Enigma nos positions ont souvent changé lors de la programmation. En effet, au départ on voulait demander à l'utilisateur de donner ses propres décalages pour les rotors mais cela ne fonctionnait pas : une lettre entrée pouvait ne pas être modifiée. De ce fait, nous avons dû prendre les vrais décalages historiques de la machine. Au début, nous avions également commencé à programmer en créant 5 listes que nous voulions modifier après chaque rotors ce qui s'avérait inutile et compliqué puisqu'une seule suffisait, le programme final comporte donc désormais qu'une seule liste contenant toutes les listes de l'alphabet. De plus, nous voulions créer des fonctions différentes pour le cryptage et le décryptage pour la méthode Enigma or un texte entré pour être crypté donnait un texte décrypté mais à partir de ce texte décrypté nous ne retrouvions pas le texte à crypter de départ. Nous nous sommes donc orientées vers une unique fonction capable de crypter ou décrypter un texte.

Nous ne sommes pas parvenues à réaliser tous nos projets puisque nous voulions aussi programmer une nouvelle méthode de cryptage inspirée du livre Bitterblue¹, malheureusement nous n'avons pas eu le temps de concrétiser ce projet.

4 Annexe : Code du programme

L'ordre des modules :

1. main_module
2. screen_module
3. screen_constants
4. text_module
5. tests

1. voir https://fr.wikipedia.org/wiki/Kristin_Cashore