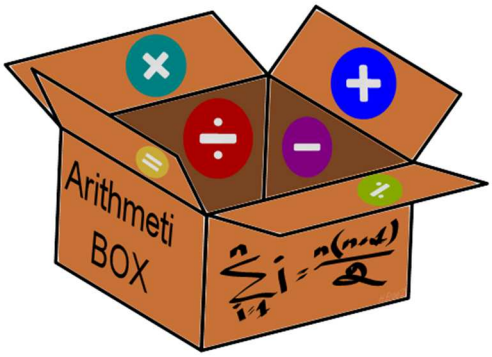

Rapport de projet
Arithmétibox

Quentin RAT
Adrien CAVALIERI
Jack KAING
Senaroith NOR
Fahath MOUGAMMADOU

Ce projet est encadré par David Hébert

Avant-propos et remerciement

Le projet assigné pour notre dernier semestre est « Arithmétibox ». Projet pensé, médité, imaginé et conçu par M. David Hébert. Toutefois, à la suite des changements de groupe pour le S4, deux aventuriers ont dû quitter la péripétie d'Arithmétibox. Il s'agit de Jason WONG et de Jeremy DOS SANTOS. Mais cette aventure ne pouvait pas s'arrêter là et Quentin, Jack et Fahath ne pouvaient pas continuer le projet à trois. C'est alors que le renfort, tant attendu par les membres historiques, est arrivé. Adrien CAVALIERI et Senaroith NOR remplacent Jason et Jeremy pour continuer cette histoire. Cependant, ils auront la lourde tâche de remplacer ces anciens membres et de convaincre Quentin, Jack et Fahath de leur utilité.

Nous remercions M. David Hébert pour l'aide qu'il nous a apporté tout au long du projet, que ce soit en S3 ou en S4.

Nous remercions, également, M. Lucian Finta qui a confirmé la continuité du projet pour ce S4.

Table des matières

Introduction

Le projet Arithmétibox a pour finalité d'être sur le site de M. David Hébert. Ce site aura pour vocation d'aider les futurs étudiants de S3 de DUT Informatique qui étudieront le module de cryptanalyse.
Pour le projet, nous continuerons à utiliser les langages de programmation web tels que le PHP, le HTML, le CSS et le JavaScript. Ainsi que le langage de compilation mathématiques LaTeX. Ce projet sera encadré par notre enseignant-chercheur et client M. David Hébert. De ce fait, nous serons cinq à développer ce site.
Quentin sera le chef de projet et se chargera de rendre les documents à notre client, M. David Hébert.

1. Description du projet

Nous reprenons la base du site qui a été créé en S3. Toutefois, il n'y aura aucune modification des fichiers déjà créés. Sauf, si une de nos missions nous le demande. Dans ce cas, les modifications auront lieu à cet effet. Nous avons donc juste à implémenter de nouvelle fonctionnalité sans toucher à cette base.

Lors de notre premier rendez-vous avec M. Hébert. Nous avons eu trois directives pour réaliser le projet.

- **Le chiffrement par substitution :**

  Continuer le développement de ce dernier et ajouter à la fonction de substitution la possibilité de changer le résultat obtenue à l'aide d'un bouton changer. De plus, il faut afficher l'analyse fréquentielle dans un cadre à côté du message à décoder.

- **Le chiffrement de Hill :**

  Terminer la méthode de chiffrement de Hill et implémenter le cryptage et décryptage de Hill à dimension n.
- **Le chiffrement de Vigenere :**

  Implémenter une fonction de cryptage et décryptage du chiffrement de Vigenere.

-

2. Gestion de projet

2.1 Les users stories

Une User story est une phrase simple, exprimée dans le langage courant, permettant de décrire avec suffisamment de précision le contenu d'une fonctionnalité à développer.

1. En tant qu'utilisateur, je veux pouvoir crypter et décrypter un message avec la méthode de substitution afin de pouvoir lire clairement ce message ou le crypter.

2. En tant qu'utilisateur, je veux crypter et décrypter un message avec une matrice de dimension n afin de coder ou décoder mes messages avec la méthode de chiffrement de Hill.

3. En tant qu'utilisateur, je veux avoir un site design, graphiquement et interactif afin de rendre dynamique la page web.

4. En tant qu'utilisateur, je veux pouvoir avoir un bouton “changer” afin de changer le résultat obtenu pour le chiffrement de substitution en fonction de la fréquence d'apparition des lettres.

5. En tant qu'utilisateur, je veux pouvoir crypter et décrypter un message avec la méthode de Vigenère afin de pouvoir lire clairement ce message ou le crypter.

2.2 Les stories techniques

6. En tant que développeur, je dois installer le logiciel Wamp/Mamp afin de pouvoir éxécuter du code php.

7. En tant que développeur, je dois disposer d'un gitHub afin de pouvoir partager mon code avec le reste de l'équipe.

8. En tant que développeur, je veux pouvoir disposer d'un groupe de messagerie afin de pouvoir discuter avec le reste de l'équipe sur, par exemple, un problème rencontré.

2.3 Stories

1. Ajouter dans un cadre à côté du message à coder l'analyse fréquentielle de chaque caractère

2. Crypter et décrypter un message avec une matrice de dimension n

3. Modifier le design de la page web

4. Ajouter en JS un bouton qui va changer le résultat du décryptage par substitution

5. Créer une fonction Chiffrement de Vigenere

2.4 Estimation du temps et de la difficulté

N° des stories	Facile	Moyen	Difficile
1		10h	
2			15h
3	8h		
4	4h		
5		15h	

Les tâches sont classées suivant leur niveau de difficulté (facile, moyen ou compliqué) cela permet d'estimer approximativement la durée de la tâche dans les cases. Le numéro des stories correspond au numéro des stories listé plus haut (dans la partie “c – Stories”).

Total= 52 points
Nous avons une équipe de 5 développeurs
1 développeur peut faire 15 points 5*15=75 points par mois
52/75= 0,7 mois

2.5 Répartition des tâches

-Fonction chiffrement de Vigenère =>Adrien, Senaroith
-Fonction Hill =>Fahath
-Fonction Substitution => Quentin, Jack

3. Explication des algorithmes

3.2 Chiffrement de Vigenere

Pour crypter/décrypter/attaquer avec la méthode de chiffrement de Hill nous avons besoin d'une matrice (qui sera ici de dimension NxN). On récupère chaque élément de la matrice et on l'insère dans une case du tableau mais comme pour toute méthode de cryptage, nous avons besoin d'une clé et d'un modulo qu'on calculera en fonction de l'alphabet du message ($alphabet). Avec les éléments récupérés dans le tableau on peut calculer le déterminant de la matrice.
Une fois le calcul fait on récupère l'équivalent modulaire en fonction du modulo calculé précédemment ($mod), ces étapes finies on affiche ces résultats grâce au LateX à l'utilisateur pour lui permettre de comprendre les étapes et ainsi avoir une meilleure compréhension de ce qu'il se passe (affichage de la matrice et le détail du calcul euclidien).

Nous devons maintenant vérifier que la clé est valide pour cela il faut calculer le PGCD entre le déterminant et le modulo et avoir un PGCD qui vaut 1, si le PGCD est différent de 1 on affichera un message disant que la clé rentrée n'est pas valide. Avant d'afficher la clé valide si l'utilisateur voulait décrypter le message on devrait récupérer l'inverse modulaire de la matrice grâce à la fonction créer par mon collègue, tout cela possible si bien sûr le PGCD vaut 1, on pourra donc commencer le décryptage une fois la matrice récupérer et le message affiché. La clé, prête, on peut passer au processus de conversion du message en chiffre qu'on utilisera pour le produit de matrice.
En fonction du choix cryptage/décryptage on utilisera la matrice qu'il faudra et par simple produit de la matrice et de l'alphabet codé on obtient le nouveau message chiffré, que l'on convertira en lettre pour pouvoir récupérer le message crypter/décrypter. Mais le plus dur, ici, est de pouvoir calculer le déterminant d'une matrice qui est de dimension N, pour cela nous avons demandé l'aide de notre professeur de cryptanalyse M. Hébert qui a pu nous expliquer les étapes du processus du calcul. Prenons par exemple cette matrice de taille n :

$$A = \begin{pmatrix} a_{1;1} & \cdots & a_{1;n} \\ \vdots & \ddots & \vdots \\ a_{n;1} & \cdots & a_{n;n} \end{pmatrix}$$

Pour calculer le déterminant de cette matrice on devra le calculer à l'aide de N déterminant de matrice de taille N-1 qu'on obtiendra à la matrice de départ une ligne et une colonne nous allons donc utiliser une fonction récursive qui nous facilitera la tâche, la nouvelle matrice d'A avec la première ligne et colonne est :

$$A_{i,j} = \begin{pmatrix} a_{1,1} & \cdots & a_{1,j-1} & a_{1,j+1} & \cdots & a_{1,n} \\ \vdots & & \vdots & \vdots & & \vdots \\ a_{i-1,1} & \cdots & a_{i-1,j-1} & a_{i-1,j+1} & \cdots & a_{i-1,n} \\ a_{i+1,1} & \cdots & a_{i+1,j-1} & a_{i+1,j+1} & \cdots & a_{i+1,n} \\ \vdots & & \vdots & \vdots & & \vdots \\ a_{n,1} & \cdots & a_{n,j-1} & a_{n,j+1} & \cdots & a_{n,n} \end{pmatrix}$$

Puis avec la formule donnée on pourra alors le calculer :

$$\det(A) = \sum_{j=1}^{n} a_{i;j}(-1)^{i+j} \det(A_{i,j})$$

On a donc utilisé la formule de Laplace. $(-1)^{i+j} \det(A_{i,j})$ est appelé cofacteur du terme et $a_{i,j}$ est $\det(A_{i,j})$ appelé le mineur du terme $a_{i,j}$

Voici le code en php de la formule de Laplace :

```php
$d = $d + (pow(-1, $k) * $matNN[0][$k] * det($taillem - 1, $smat));
// $k=i+j $smat= Matrice N-1
```

3.3 Chiffrement de Vigenere

Depuis le début du projet, tous les codages et chiffrements donnés par M. Hébert ont été vus en cours. Toutefois, le codage de Vigenere nous a été proposé par M. Hébert comme une ouverture sur la cryptanalyse vue en S3.
Ce codage nous été inconnu et nous avons donc fais des recherches pour comprendre le principe de Vigenere et user de notre imagination pour implémenter ce codage.
Le principe du codage de Vigenere se base sur le codage de César (on utilise un alphabet allant de a à z et on utilise une clé pour codé le message). Sauf qu'avec Vigenere, la clé n'est pas un alphabet décalé mais un mot. Autrement dit, on utilise un mot clé et on effectue des translations de lettres par rapport au mot clé.
Prenons un exemple :

- On veut coder le mot « toilette »
- avec le mot clé : « code »

On répète le mot clé tout le long du message, le texte codé est obtenu en additionnant les deux nombres.
Codage du mot « toilette » :

Mot clé	c	o	d	e	c	o	d	e
Position dans l'alphabet	2	14	3	4	2	14	3	4
Mot à coder	t	o	i	l	e	t	t	e
Position dans l'alphabet	19	14	8	11	4	19	19	4
Addition	21	2	11	15	6	7	22	8
Mot codé	v	c	l	p	g	h	w	i

Le message codé de « toilette » avec la clé « code » est « vclpghwi ».
Le décodage suit le même principe.

Décodage du mot « vclpghwi » :

Mot à décoder	v	c	l	p	g	h	w	i
Position dans l'alphabet	21	2	11	15	6	7	22	8
Mot clé	c	o	d	e	c	o	d	e
Position dans l'alphabet	2	14	3	4	2	14	3	4
Soustraction	19	14	8	11	4	19	19	4
Mot décodé	t	o	i	l	e	t	t	e

Le message décodé de « vclpghwi » avec la clé « code » est « toilette ».

À noter que Vigenere ne peut être attaqué par force brute ou par analyse fréquentielle. Cependant, pour casser ce code, on peut utiliser le test de Kasiski qui consiste à repérer des répétitions de lettres dans un message codé.

Algorithme : Algorithme du codage de Vigenere

Entrées : Message à coder (*msgAC*), clé (*cle*) et alphabet (*Alphabet*)
Sorties : Message codé (*msgC)*
fonction : *indexAlpha*, retourne la position d'une lettre dans l'alphabet
début
Pour chaque lettre de *msgAC* et de *cle* **faire**
motClair <- *indexAlpha* de *msgAC*
cleCode <- *indexAlpha* de *cle*
code <- *motClair* + *cleCode*
code modulo taille *Alphabet*
msgC <- *code*

retourner message codé (*msgC)*
fin

Algorithme : Algorithme du décodage de Vigenere

Entrées : Message à décoder (*msgDC*), clé (*cle*) et alphabet (*Alphabet*)
Sorties : Message clair (*msgC)*
fonction : *indexAlpha*, retourne la position d'une lettre dans l'alphabet
début
Pour chaque lettre de *msgAC* et de *cle* **faire**
motClair <- *indexAlpha* de *msgDC*
cleDeCode <- *indexAlpha* de *cle*
de*code* <- *motClair* - *cleCode*
de*code* modulo taille *Alphabet*
msgC <- *decode*

retourner message clair (*msgC)*

fin

Conclusion