Rapport de projet

Audric Schiltknecht

Résumé

Implémentation d'une machine Enigma en C

1 Introduction

Pour ce sujet, il faut coder en langage C une machine Enigma capable de coder et décoder des séquences de caractères. Les machines seront représentées dans un fichier texte qui sera chargé par le programme.

2 Codage

Les raffinages des fonctions sont disponibles dans les listings 1, 2, 3 et 4.

Algorithm 1 decale_enigma

1: **Procédure** DECALE_ENIGMA (Enigma e)
2: $i \leftarrow 0$ ◁ C'est le numéro du rotor
3: tourne = Vrai ◁ On fait toujours tourner le premier couple
4: **Tant que** i< $NB\_ROTOR - 1$ && *tourne* **Faire** ◁ Les rotors $j = 0..i - 1$ ont tourné
5: tourne ← tourne & & cran$[i][0]$ ◁ On regarde s'il faut faire tourner le couple suivant
6: `decale_rotor`(rotor$[i]$) ◁ On fait tourner le rotor
7: `decale_cran`(cran$[i]$)◁ On fait tourner le cran suivant
8: $i \leftarrow i + 1$
9: **Fin Tant que**
10: **Fin Procédure**

3 Modification du code

J'ai légèrement modifié le code du fichier `main.c`, qui permet de saisir le mot à encoder sur la ligne de commande suivant la syntaxe suivante :
`enigma fichier_machine [mot à encoder]`.

Si le mot à encoder n'est pas passé en paramètre, le programme utilisera le mot « `BAC` ».

Algorithm 2 decale_rotor

```
1: Procédure DECALE_ROTOR (Tableau caractères t)
2:     tmp ← t[0]
3:     Pour i ← 0 à NB_LETTRES-2 Faire
4:         suivant = ← i+1 ◁ Il faut calculer l'écart entre la lettre suivante et son cryptage
5:         ecart ← entier(t[suivant]) - suivant
6:         t[i] ← caractere((i+ecart+NB_LETTRES)mod [NB_LETTRE]) ◁ La nouvelle lettre s'obtient en ajoutant cet écart au code de la lettre non-chiffrée. Attention : l'écart peut être négatif !
7:         ecart ← entier(tmp)
8:         t[NB_LETTRES - 1] ← caractere((NB_LETTRES-1)mod [NB_LETTRES])
9:     Fin Pour
10: Fin Procédure
```

Algorithm 3 decale_cran

```
1: Procédure DECALE_CRAN (Tableau entiers t)
2:     tmp ← t[0] ◁ On sauvegarde le premier élément
3:     Pour i ← 0 à NB_LETTRES-2 Faire
4:         t[i] ← t[i + 1] ◁ On décale les éléments par permutation circulaire
5:     Fin Pour
6:     t[NB_LETTRES-1] ← tmp
7: Fin Procédure
```

Algorithm 4 chiffre_chaine

```
1: Tableau de caractères Fonction CHIFFRE_CHAINE (Tableau caractère t, Enigma e)
2:     l ← longueur(t)
3:     Pour i ← 0 à l-1 Faire
4:         Si c[i] ∉ lettres codées par Enigma Alors Retourne Erreur
5:         Sinon
6:             cc[i] ← chiffre_car(c[i],e) ◁ On encode le caractère, et on le stocke dans la nouvelle variable cc
7:             decale_enigma(e) ◁ On décale la machine
8:         Fin Si
9:     Fin Pour
       Retourne cc
10: Fin Fonction
```