Exercice en Java Card 2.x
Master SEMS (SECSdF2)

L'objectif de ce TP est de chiffrer et de déchiffrer les données échangées entre un terminal et une cardlet en utilisant la technologie Java Card 2.
L'algorithme de chiffrement utilisé est le triple DES car c'est cet algorithme qui est implémenté sur nos cartes. Il est basé sur une clé de 192 bits et des blocs de 64 bits avec un vecteur d'initialisation de même taille que les blocs.

Nous considérons l'environnement habituel de Eclipse en partant des programmes client et applet Echo. Il s'agit de compléter ces deux programmes avec les instructions adéquates.

Question 1 : Compléter le client afin qu'il puisse :

- envoyer une donnée de 8 octets en clair à l'applet. Cette donnée sera chiffrée par l'applet et renvoyée au terminal qui la déchiffre et l'affiche.
- envoyer une donnée chiffrée à l'applet qui la déchiffre et retourne la donnée en clair. Cette donnée sera affichée par le terminal.

Pour cela, il faudra suivre les étapes suivantes :

a. définir les commandes APDU nécessaires,
b. importer les packages suivants :

```
import javax.crypto.*;
import javax.crypto.spec.DESedeKeySpec;
import javax.crypto.spec.IvParameterSpec;
```

c. déclarer un tableau de 8 octets initialisé à une valeur hexadécimal qui représente la donnée que le client doit envoyer à l'applet.
d. déclarer un tableau d'octets qui a la taille de la clé et qui contient la valeur de la clé secrète.
e. déclarer le vecteur d'initialisation initialisé à 0 car l'algorithme du 3DES côté carte repose sur un vecteur d'initialisation qui est toujours égal à 0 à cause de l'espace mémoire limité.
f. Chaque opération de chiffrement ou de déchiffrement commence par les étapes suivantes:
    - créer un objet DESedeKeySpec correspondant à la clé secrète qui a été définie,
    - créer une fabrique de clés secrètes avec la classe SecretKeyFactory pour l'algorithme 3DES nommé ici "DESede" pour DES Encryption-Decryption-Encryption,
    - générer la clé secrète avec la méthode generateSecret(),
    - créer un objet de type IvParameterSpec correspondant au vecteur d'initialisation,
    - créer un objet de type Cipher en précisant la variante d'algorithme 3DES qui sera utilisé ici, à savoir "DESede/CBC/NoPadding", c'est-à-dire le DESede

utilisant un découpage de messages en blocs (Cipher Block Chaining, ici de 64 bits) sans bits de bourrage (No padding).

- initialiser l'objet Cipher en précisant l'opération (chiffrement ou déchiffrement) à effectuer. L'initialisation nécessite 3 paramètres :
  i. nature de l'opération : la constante Cipher.ENCRYPT_MODE si chiffrement ou Cipher.ENCRYPT_MODE si déchiffrement.
  ii. la clé secrète
  iii. et le vecteur d'initialisation.

g. L'opération de chiffrement/déchiffrement proprement dite est réalisée grâce à la méthode doFinal().

Question 2 : Compléter l'applet Java Card afin qu'elle puisse traiter les commandes APDU envoyées par le terminal.

Pour cela, il faudra suivre les étapes suivantes :

h. importer les packages suivants :

```
import javacard.security.DESKey;
import javacard.security.Key;
import javacard.security.KeyBuilder;
import javacardx.crypto.Cipher;
```

i. déclarer une clé identique à celle du client.
j. déclarer un tableau susceptible de stocker les données reçues du client. Notez qu'il existe une méthode qui permet de recopier des données d'un tableau à un autre comme suit : Util.arrayCopyNonAtomic(source, index début dans la source, destination, index début dans destination, taille des données à recopier) ;
k. selon la commande APDU, la donnée reçue sera chiffrée ou déchiffrée avant d'être renvoyée au terminal.
l. Qu'il s'agisse d'une opération de chiffrement ou de déchiffrement, les étapes suivantes doivent être réalisées :
   - créer un objet de type Cipher pour préciser qu'il s'agit d'un chiffrement/déchiffrement avec l'algorithme 3DES de type CBC No padding à l'aide de la constante Cipher.ALG_DES_CBC_NOPAD et d'un paramètre mis à false.
   - ensuite créer un objet DESKey pour fabriquer la clé avec KeyBuilder.buildKey() avec comme paramètres KeyBuilder.TYPE_DES, KeyBuilder.LENGTH_DES3_3KEY et false.
   - la méthode setKey(clé, offset) permet d'initialiser l'objet DESKey avec la clé.
   - La méthode init() de l'objet Cipher permet de fixer l'opération à effectuer : chiffrement si le deuxième paramètre est égal à Cipher.MODE_ENCRYPT et déchiffrement si le deuxième paramètre est égal à Cipher.MODE_ DECRYPT. Le premier paramètre étant l'objet DESKey correspondant à la clé secrète.
   - Le chiffrement/déchiffrement est effectivement réalisée à l'aide de la méthode doFinal() de l'objet Cipher.

Client ====== donnée en clair =====> Applet (exécute la méthode de chiffrement de la donnée)

Client < ========== donnée chiffrée======Applet
déchiffre la donnée
et l'affiche

Client =======donnée chiffrée ======> Applet (exécute la méthode de déchiffrement de la donnée)
Client <======= donnée en clair ====== Applet
affiche la donnée