Relais Épreuve #2 - CsCoins

Description

Programmation à relais est une épreuve où les 3 coéquipiers sont dispersés dans différentes salles et travaillent sur des projets distincts. Chaque heures, les coéquipiers doivent se relayer, changer d'ordinateur et continuer le projet débuté par les autres membres de l'équipe. Durée = ~3 heures. (1 heure par relai)

Les langages supportés sont Java et Python.

Règlements

- Aucune communication directe n'est permise entre les coéquipiers ou les différentes salles.
- Un fichier readme.txt est obligatoire. Vous devez y expliquer brièvement vos fonctionnalités ainsi que les questions que vous avez répondues Chaque question ou fonctionnalité implémentée qui ne se retrouve pas dans readme.txt, même si évidente, ne sera pas corrigée. Ne vous exposez pas à la colère d'un correcteur manquant de sommeil!
- Dans le même ordre d'idée, soyez certain que votre code soit facile à compiler et exécuter.
- Le seul moyen de communication entre coéquipiers toléré, et fortement suggéré, est le fichier readme.txt et les commentaires dans le code.
- Vous pouvez expliquer pourquoi vous méritez des points pour une fonctionnalité incomplète (dans le fichier readme.txt). Par contre, les points sont généralement attribués pour les fonctionnalités complètes uniquement. Assurez-vous de compléter vos fonctionnalités dans les délais.
- Les trois épreuves ont des totaux différents, mais elles comptent chacune pour un tiers des points. Les résultats seront normalisés.

Project #2 - CSCoins (32 points)

Certainement une des merveilles de l'Internet, *Bitcoin* est une monnaie cryptographique et distribuée. La plupart d'entre vous devraient savoir de quoi il s'agit avec toute la couverture médiatique de l'année dernière.

> *A cryptocurrency is a type of digital currency (which in turn is a type of alternative currency) that relies on cryptography, usually alongside a proof-of-work scheme, in order to create and manage the currency. Cryptocurrencies are peer-to-peer and decentralized*[1]*.*

Bitcoin n'est pas la seule monnaie cryptographique, il y a une multitude de "clones", des "alt-coins". Ceux-ci copient *Bitcoin* avec plus ou moins d'innovation. Pour cette épreuve, vous devez bâtir un portefeuille pour le nouveau "alt-coin" du jour, le *CSCoins*. Ne vous en faites pas, vous n'aurez pas à implémenter des fonctions cryptographiques complexes, la majorité du travail est fait par un API. Votre travail consiste à bâtir une application GUI qui intègre cette API.

L'abréviation à trois lettre pour le *CSCoins* est: **CSG** (**CSC** est déjà un alt connu)

À des fins d'inspiration, il y a des captures d'écran du portefeuille *Bitcoin* original dans le dossier **samples.** Vous n'avez pas à suivre ce design, mais vous devez utiliser un design qui a du sens.

Requis:

1. Listez et stockez les adresses *CSCoins* du propriétaire dans le portefeuille. Le fichier **Data.txt** contient les adresses existantes (incluant la clé publique et la clé privée). Vous n'avez pas besoin de décoder ce fichier, vous pouvez utiliser votre propre stockage et format. **(4 points)**
2. Créez la fonction de carnet d'adresses. Le portefeuille permet de gérer et afficher une liste d'adresse de destination pour un usage futur (vous n'avez pas besoin d'une clé publique, ni d'une clé privée pour celles-ci). **(5 points)**
3. Listez les transactions du propriétaire qui apparaissent dans le registre des transactions (blockchain). **(4 points)**
4. Afficher la balance TOTALE du propriétaire.. **(3 points)**
5. Permettez à l'usager de générer une nouvelle adresse. Vous devez aussi stocker la clé publique, la clé priveé, ainsi que l'adresse. Ce stockage doit être persistant. **(4 points)**

[1] Wikipedia: http://en.wikipedia.org/wiki/Cryptocurrency

6. Implémentez l'envoi de monnaie à partir du solde de l'usager, vers une adresse *CSCoins*. Le solde doit être mis à jour. **(4 points)**

7. Cryptez le portefeuille (adresses, clés publique, clés privée) en utilisant votre propre implémentation du "XOR Cipher" (un article Wikipédia est fourni (en anglais seulement, mais les exemples devrait être suffisant)). Le mot de passe doit être de 8 caractères ou plus. Cette partie ne requiert pas de GUI, mais vous devez démontrer que votre implémentation fonctionne. **(5 points)**

8. Ajouter la fonction de cryptage dans le GUI. Demander le mot de passe de l'usager avant l'envoi de monnaie et avant de générer une nouvelle adresse. Vous ne devez pas demander un mot de passe au démarrage, mais seulement dans ces circonstances. **(3 points)**

The api: (Create an instance of CsCoinsApi<Langage>Impl)

```
The currency core parameters.
def coreParameters(): Core

The balance of an address according to the sum of transaction in the
blockchain
def addressBalance(address: Address): Long

Generate a new random public key / private key pair and the new address you
can receive coins to.
def generateAddress(): (Address, PublicKey, PrivateKey)

List all transactions on an address (sent or received) from the blockchain
def listTransactions(address: Address): List[Transaction]

Send coins to an address.
def sendCoins(amount: Long, publicKey: PublicKey, privateKey: PrivateKey,
destinationAddress: Address): Boolean
```

The data classes:

```
case class Address(address: String)

case class PrivateKey(text: String)

case class PrivateKey(text: String)

case class Transaction(address: Address, amount: Long)

trait Core {
  /* How much unit is a coins, ex: 1 coins = 1000L */
  val CurrencyDivision: Long

  /* The maximum coins that can be created */
  val CurrencyLimit: Long
}
```

Notes:

- For Scala: The jar name is CSCoins.jar. The class to use is **CsCoinsApiScalaImpl**.
- For Java: The jar name is CSCoins.jar. The class to use is **CsCoinsApiJavaImpl**.
- For Python: The api file name is: **CsCoinsApi.py** the class to use is **CsCoinsApiImpl**. The API is still in Scala, so the JVM is required, but glue code is provided to run the Scala API in Python, this should be seamless.
- For Java and Scala, use your favorite IDE for GUI building, also the Swing documentation is provided.
- For Python, the PyQt api and documentation is provided. Qt Designer is available to build GUI.