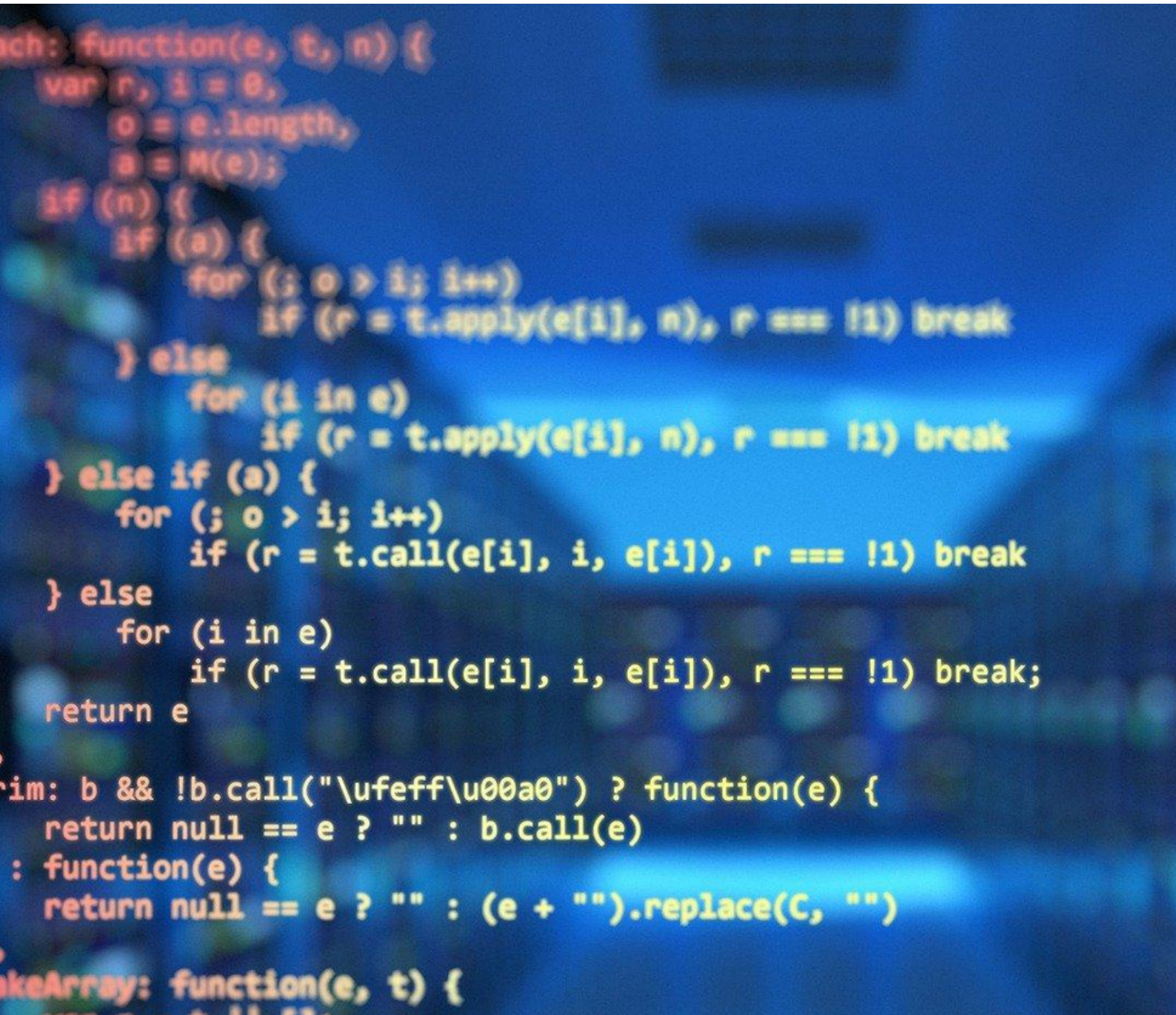
KENZA KOUDDANE ET SAYFE-DIN EREHAIHI

PROJET :

BLOCKCHAIN APPLIQUÉE À UN PROCESSUS ÉLÉCTORAL

2021/2022 - LU2IN006 - STRUCTURES DE DONNÉES



2

OBJECTIFS DU PROJET :

Grâce aux structures de données, il est possible d'organiser et de stocker des données pour rendre leur manipulation plus concrète. En effet, en informatique, il est nécessaire que les données puissent être utilisées convenablement et soient accessibles. Il existe plusieurs façons de stocker des données, nous les exploitons dans nos divers algorithmes.

Dans ce projet nous avons essayé de représenter un processus électoral par scrutin uninominal majoritaire à deux tours, nous avons réfléchi aux structures de données nécessaire à mettre en place pour un scrutin anonyme, fiable et garantissant l'efficacité des algorithmes, tout en s'inspirant du protocole RSA. Notre programme est divisé en 5 parties, chacune se consacrant sur une catégorie essentielle du processus électoral.

Nous prenons en compte un système électoral basé sur des clés (une clé publique et une clé privé par personne) ainsi un électeur exprime son vote, une déclaration, anonymement à l'aide des clés. De la même manière, un candidat peut exprimer une déclaration de candidature.

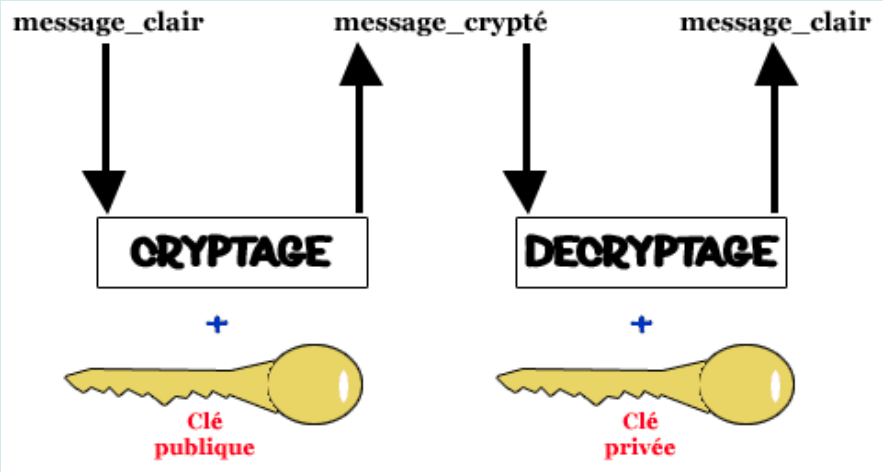


Schéma
Protocole RSA

SOMMAIRE

P.4 Schéma de la structure du projet

P.5 Descriptions des structures

P.6 Description globale des parties

P.7 Réponses aux questions

REPERTOIRE

PROJET

PARTIE 5

partie5.c partie5.h
main5.c
commande :
make partie5 && ./partie5

PARTIE 4

partie4.c partie4.h
main4.c
commande :
make partie4 && ./partie4

PARTIE 3

partie3.c partie3.h
main3.c
commande :
make partie3 && ./partie3

PARTIE 2

partie2.c partie2.h
main2.c
commande :
make partie2 && ./partie2

PARTIE 1

exo1.c exo2.c main1.c
exo1.h exo2.h
commande :
make partie1 && ./partie1

Remarque : w.sh permet d'obtenir la cle public de chaque candidat ainsi que le nombre de vote obtenue, **commande** : ./w.sh

PARTIE2.H

Key : structure représentant une clé, qu'elle soit public ou privée, contenant 2 variables de type long;

type de structure -> tableau

Signature : structure permettant de produire des déclarations de vote signée, contenant un tableau de long ainsi que la taille de celui-ci;

Protected : structure contenant les déclarations signées c'est-à-dire la clé publique de l'émetteur, son message, qui est la déclaration de vote, ainsi que la signature associée qui sont des données protégées;

type de structure -> listes (simplement) châinées

CellKey : liste chaînée de clés;

type de structure -> listes (simplement) châinées

CellProtected : liste chaînée de déclarations signées;

PARTIE3.H

HashCell : structure utilisée de deux manières différente, 1-stocker la clé public d'un électeur ainsi qu'une valeur pour savoir si cette personne a déjà voté, 2-stocker la clé public d'un candidat ainsi que son nombre de voix;

type de structure -> table de hachage, les collisions sont gérée par probing linéaire

HashTable : table de hachage contenant des HashCell de votants ou de candidats représenté donc par un tableau de pointeur de HashCell ainsi que la taille de ce tableau;

PARTIE4.H

Block : un bloc du blockchain contenant une clé, une liste de déclaration, une valeur haché et la valeur haché du bloc précédent;

PARTIE5.H

type de structure -> arbre

CellTree : Noeud d'une structure de type arbre contenant le noeud père, le premier fils, le noeud suivant et la hauteur

6

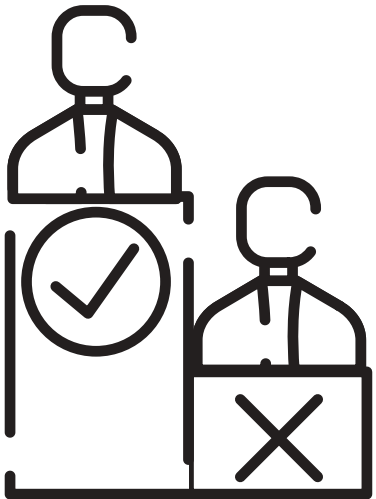
L'anonymat d'un électeur et la confidentialité de son vote permettent la sécurité et l'intégrité de l'élection. Pour cela, nous avons implémenté un algorithme de cryptographie inspiré du chiffrement RSA, notamment utilisé pour l'échange de données confidentielles sur Internet. La Partie 1 se consacre donc aux fonctions de développement d'outils.

Dans un vote, un citoyen effectue des déclarations que ce soit des déclarations de vote ou de candidature, ces déclarations doivent être sécurisées. La Partie 2 contient les algorithmes de manipulations de structures sécurisées, comme les clés, les signatures et les déclarations signées.

En considérant une base de déclarations centralisée, nous allons collecter tous les votes et annoncer le vainqueur de l' élection à tous les citoyens. Après avoir généré pour chaque citoyen une carte électorale unique, nous collectons les déclarations de vote et gérons leur validité, dans la Partie 3 nous nous intéressons à la lecture et au stockage des données dans des listes chaînées, ainsi qu'à l'élection du gagnant.

Par la suite, nous utilisons une base de déclaration de vote décentralisé. Nous implémentons la blockchain qui nous permet de mieux sécuriser notre scrutin de vote. Cette blockchain rend possible la vérification du résultat du vote pour chaque citoyen car tous citoyens possèdent une copie de la base (dans notre cas les données sont les déclarations de vote signées). La Partie 4 se concentre sur la structure d'un blocs ainsi que sa gestion.

Dans la Partie 5, nous manipulons un arbre de block en considérant le consensus comme étant la chaîne de blocs la plus longue. Le consensus dans une blockchain signifie que tous les blocs du réseau doivent se mettre d'accord sur une version identique de la blockchain.



Partie 1

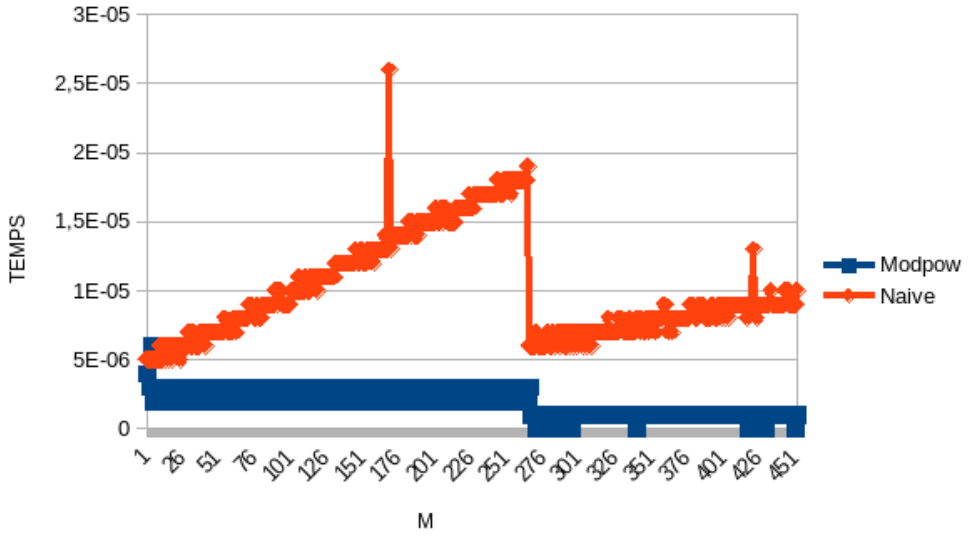
EXERCICE 1

Q1 :
La complexité de is_prime_naive est O(p-3) car nous avons une boucle qui enumère les entiers de 3 à p-1, donc la compléxité est de O(p).

Q2 :
Le plus grand nombre obtenu est 241692.

Q3 : La complexité de modpow_naive est de O(m) car nous effectuons une boucle avec une operation elementaire m fois.

Q4 :
Nous remarquons que le temps d'éxecution de la fonction non naive est plus performant. En effet, les temps varient très peu comparer à la fonction naive. Plus l'entier m est est grand, plus le temps d'éxecution de la fonction naive s'allonge alors que la fonction non naive garde un temps presque constant quelque soit le nombre. (voir fichier modpow_temps.txt)



Partie 5

EXERCICE 8

Q8 :
signature de la fonction de fusion :
CellProtected*merge_declarations(CellProtected* p1,CellProtected* p2);

8

La complexité est en O(h) avec h la longueur de la liste p1.
Pour avoir une fusion en O(1), une solution serai d'avoir des listes fifo nous permettant d'avoir le dernier element dans une autre structure et d'apporter les modifications en temps constant.

EXERCICE 9

Q7 :
CONCLUSION

Dans le cadre d’un processus de vote, l’utilisation d’une blockchain permet transparence et sécurité, en effet il n’y a pas d’organe central de contrôle, les données sont partagées entre tous les citoyens. Ainsi, l’information est immédiate. Cependant, lorsque le vote est terminé, il y a possibilité de fraude simplement en ajoutant à la fin de la BlockChain n'importe quel vote donc la plus longue chaîne finale pourrait finalement n'être composée que de fausses déclarations.