



# ALGORITHMIE

Par K vin Ansard

# Les principes d'une Blockchain



LES DONNÉES DOIVENT  
ÊTRE PÉRENNES



LES DONNÉES DOIVENT  
ÊTRE INFALSIFIABLES



ARCHITECTURE  
DISTRIBUÉE



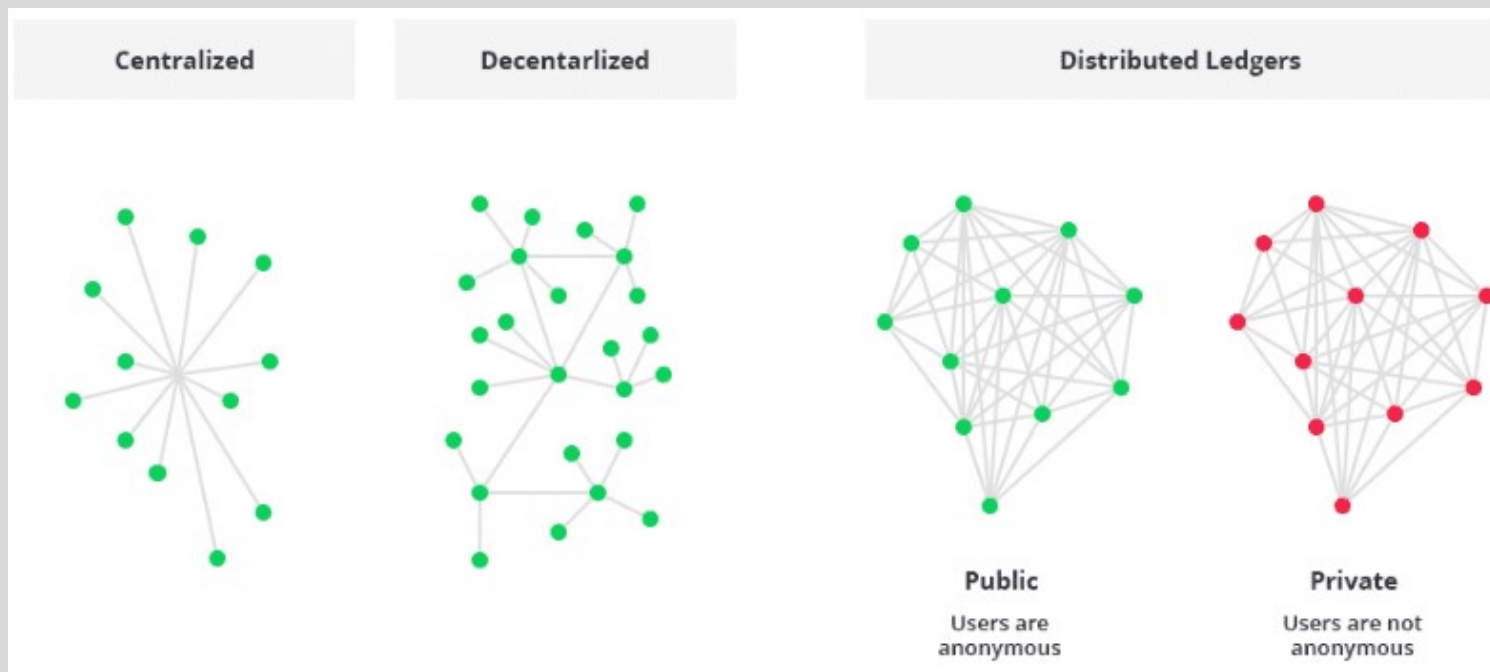
## Qu'est ce que des données pérennes ?

Une donnée pérenne est une donnée toujours accessible, peu importe le lieu ou le temps passé après sa création

# Les données doivent être infalsifiables

- Autrement dit les données doivent toujours rester les mêmes peu importe le temps qui passe. Cela permet donc d'assurer la sécurité de la Blockchain

# L'Architecture doit être distribuée





# Pourquoi les Blockchains sont-elles si populaires ?



# Comment fonctionnent les Blockchains ?

1. Les Blockchains fonctionnent toutes sans exception avec une fonction de Hashage
  - C'est quoi une fonction de Hashage ?
  - A quoi cela sert ?
  - Comment cela fonctionne ?
  - Pourquoi c'est utilisé ?
2. De quoi est constitué une Blockchain ?

# C'est quoi une fonction de Hashage ?

Binaire	hexadécimal
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	A
1011	B
1100	C
1101	D
1110	E
1111	F

Une fonction de hashage est une fonction mathématique qui a pour but d'obtenir une empreinte pour une donnée. La donnée peut correspondre à des fichiers, du texte, des objets etc...

Les fonctions de hashage sont pour la plus part du temps représentées en format 256bits, cette empreinte est en réalité constituée de 64 caractères hexadécimaux qui vont de 0 à f.

Un caractère hexadécimal contient donc 4bits.



Voici un exemple !!!

test =>

9F86D081884C7D659A  
2FEAA0C55AD015A3BF  
4F1B2B0B822CD15D6C  
15B0F00A08

test! =>

1882B91B7F49D479CF1  
EC2F1ECEE30D0E5392E  
963A2109015B7149BF7  
12AD1B6



Permet de faire des signatures



Permet de stocker des mots de passe

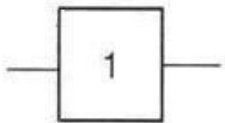
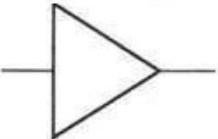
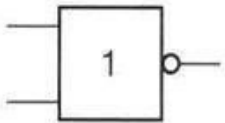
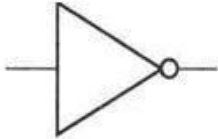
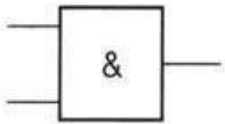
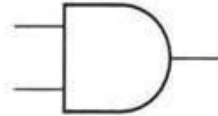
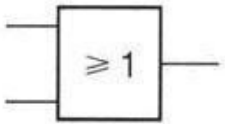
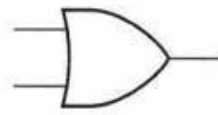
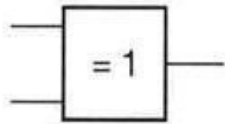
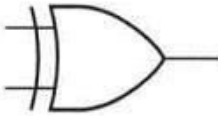
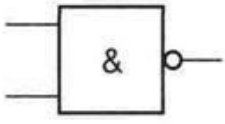
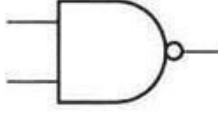


Vérifie l'intégrité d'une donnée



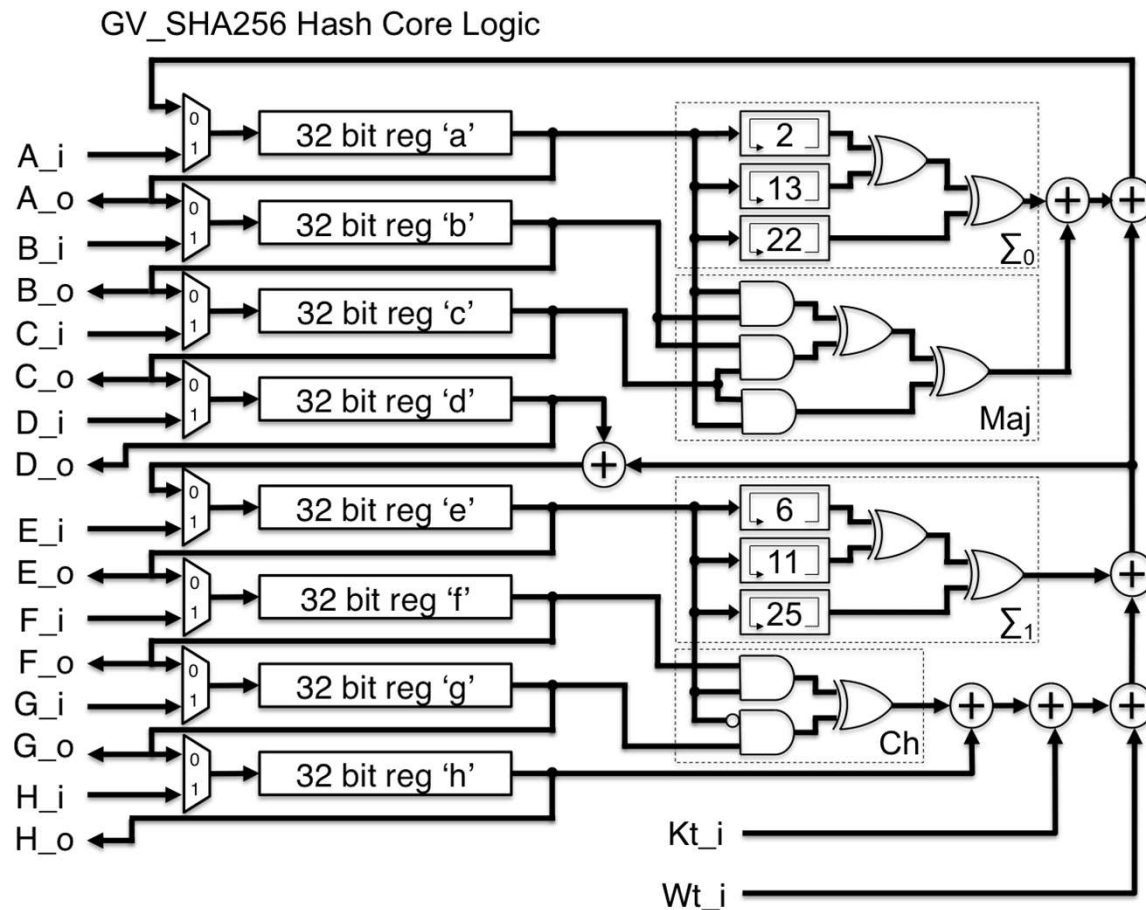
Fait des Blockchains

A quoi cela sert ?

Porte OUI (YES)			entrée 0 1	sortie 0 1
Porte NON (NO)			entrée 0 1	sortie 1 0
Porte ET (AND)			entrées 0 0 0 1 1 0 1 1	sortie 0 0 0 1
Porte OU (OR)			entrées 0 0 0 1 1 0 1 1	sortie 0 1 1 1
Porte OU exclusif (XOR)			entrées 0 0 0 1 1 0 1 1	sortie 0 1 1 0
Porte NON-ET (NAND)			entrées 0 0 0 1 1 0 1 1	sortie 1 1 1 0

Comment cela fonctionne ?

Cela fonctionne avec un algorithme qui utilise des portes logiques



19/01/2021

Voici la logique  
derrière le code  
de la fonction  
de hashage  
SHA256

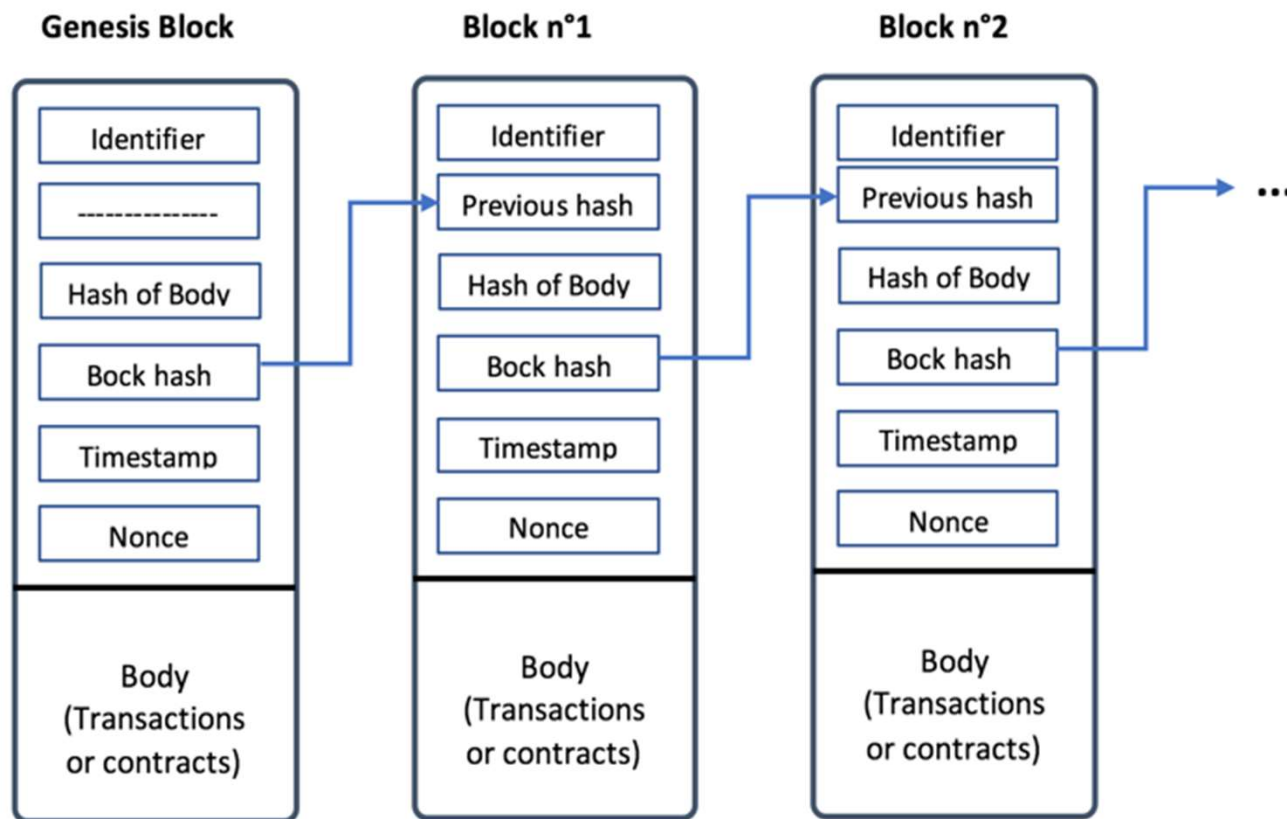
Cette fonction est  
utilisée pour la  
crypto-monnaie  
Blockchain.

# Pourquoi c'est utilisé ?

## **Pour éviter les collisions !!!**

Une collisions est le fait de trouver le même hash pour deux fichiers différents



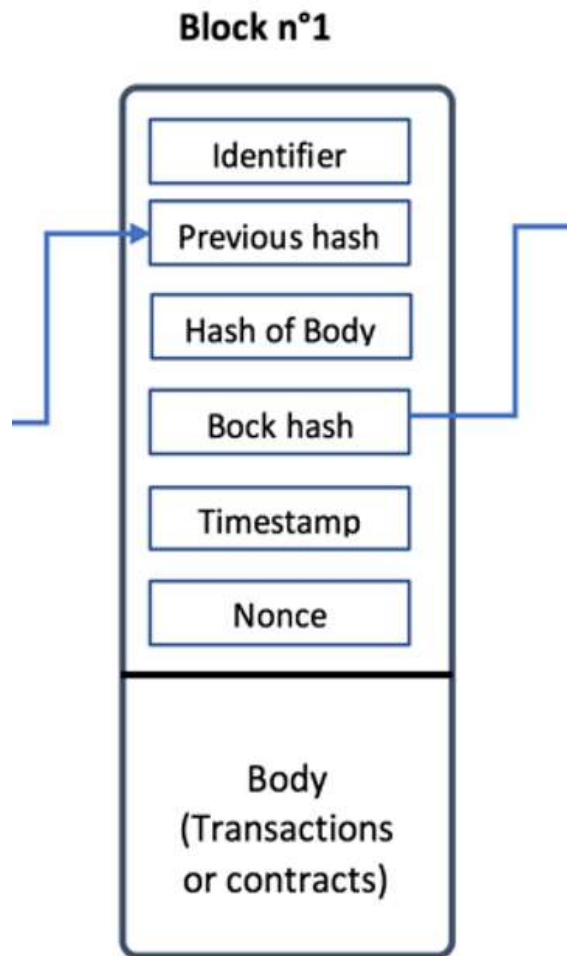


# Une Blockchain c'est quoi ?

Une chaîne de Block



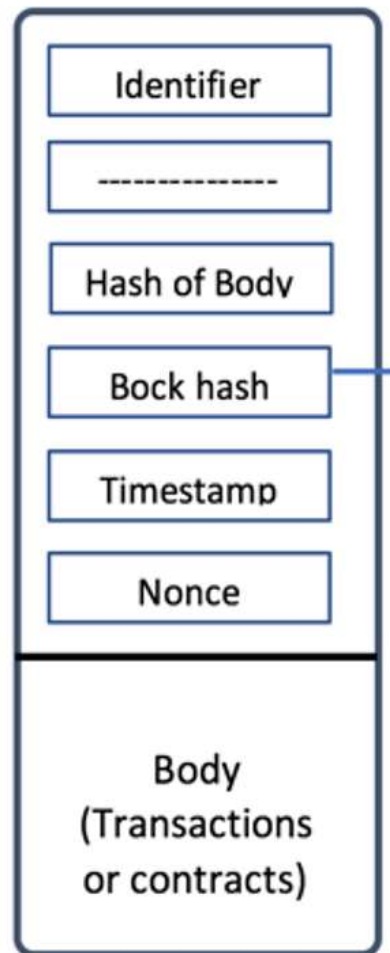




De quoi est constitué  
un block d'une  
Blockchain ?

1. Hash du block précédent
2. Donnée ou transaction
3. Signature
4. Preuve de travail
5. Sa date de création
6. Hash du block lui-même

## Genesis Block



## Le block g n sis

Ce block est le tout  
premier block  
d'une blockchain il  
ne contient donc  
pas de Previous  
Hash

# Live Coding Python les bases avancées

Les points abordés :

- Les Classes
- Les Constructeurs
- Les Méthodes
- Mise en place d'une classe d'exemple

# Passons à la pratique !

---

Les prérequis

Python

---

Un IDE (VS code / Pycharm)

---

Groupe de 2/3 imposés

---

Exercices

# A VOUS DE RECODER VOTRE BLOCKCHAIN

Un sujet vous sera donné en PDF vous expliquant l'algorithme de chaque exercice.

# Exercice 1 - La classe Block

Concevoir une classe qui représentera les blocks de votre blockchain

1. Hash du block précédent
2. Donnée ou transaction
3. Signature
4. Preuve de travail
5. Sa date de création
6. Hash du block lui-même



# Exercice 2 - La classe Block

Faites une classe qui représentera les blocks de votre blockchain

# Exercice 3 - La classe Blockchain

Concevoir une classe qui représentera votre blockchain

# Exercice 4 - La classe Blockchain

Faite une classe qui représentera votre blockchain

# Exercice 3 - Finir votre Blockchain

```
Blockchain validity: True
Block #0 [
  index: 0
  previous hash: None
  timestamp: 2019-12-22 12:05:47.902654
  data: Genesis block
  hash: 000024654ea02bfefcb159df0701373b92997a680d3a0358a3a9e7d065a54795
  nonce: 74766
]

Block #1 [
  index: 1
  previous hash: 000024654ea02bfefcb159df0701373b92997a680d3a0358a3a9e7d065a54795
  timestamp: 2019-12-22 12:05:48.074296
  data: Second Block
  hash: 000082fd8ea8de4fc0e289107e31bb4849cd13b0832ba34df50660652afde877
  nonce: 92187
]
```



19/01/2021

Exercice 4 - Bonus  
recoder la  
fonction SHA256

Fuyez pauvres fous  
!!!