

**Ingeniería Matemática e Inteligencia Artificial**

**Memoria Blockchain**

Práctica Final

Fundamentos de los Sistemas Operativos

Sergio Herreros Pérez

Daniel Sánchez Sánchez

**BlockChain.py**

* Clase Bloque:

Forma los cimientos de la cadena. Cada bloque tiene unas ciertas transacciones, las cuáles se formalizan una vez que el bloque se une a la cadena. Para “enganchar” los bloques entre sí, se le da un hash generado según su propia estructura, además de tener un valor de prueba para que cumpla con la dificultad elegida (que empiece con cierto número de ceros), y tiene el hash del bloque anterior en la cadena. Por último, tiene un índice y un registro de tiempo correspondiente a su creación.

Además, tiene tres métodos: calcular hash, el cual calcula el hash del bloque; toDict, el cual simplemente pasa el bloque a diccionario, y fromDict, que hace lo contrario.

* Clase Blockchain:

La cadena. Sirve para conectar los bloques y contiene la gran mayoría de métodos. Contiene una lista con los bloques añadidos, una lista con las transacciones aun no añadidas, su nivel de dificultad (explicado en la clase bloque) y el número de bloques añadidos. Además, para empezar, se añade un nuevo bloque.

Sus métodos son:

1. Primer bloque: Crea un nuevo bloque con hash previo igual al string 1 y lo integra, inicializándose con el hash que le corresponde
2. Nuevo bloque: Crea un nuevo bloque dado un hash previo cuyas transacciones son el pool de blockchain y lo devuelve
3. Nueva transacción: Crea una nueva transacción con un origen, una cantidad y un destino y la añade a pool.
4. Prueba trabajo: Calculará el hash del bloque añadiendo uno a la prueba hasta que cumpla la dificultad.
5. Prueba válida: Comprueba que un bloque tenga un hash correcto.
6. Integra bloque: Añade un bloque dado a la blockchain si el hash previo y su hash son correctos.
7. Check chain: Recorre la cadena desde el inicio y si algún bloque tiene un hash previo incorrecto, devuelve False. En caso contrario devolverá True.
8. toDict: Pasa la clase a diccionario (al igual que cada bloque de la blockchain).
9. fromDict: Crea la clase a partir de un diccionario creado por toDict.

El funcionamiento será el siguiente: se creará la blockchain cuando se inicialice el programa (con ello generándose el primer bloque). Se irán haciendo transacciones a medida que se hagan requests, y una vez que se mine un bloque (y no haya conflictos), se añadirán esas transacciones al nuevo bloque.

**Blockchain\_app.py**

En este fichero se incluye la funcionalidad de la aplicación de Blockchain. Utilizando Flask, creamos una app que se podrá ejecutar por cada nodo y permitirá la conexión con otros nodos, de manera que se cree una red peer-to-peer.

* Nueva transacción:
  + Asociamos la ruta “/transacciones/nueva” y el método POST a la funcion nueva\_transaccion()
  + Recibe un input en json y