PROYECTO 1
BLOCKCHAIN INDEXING

Integrantes:

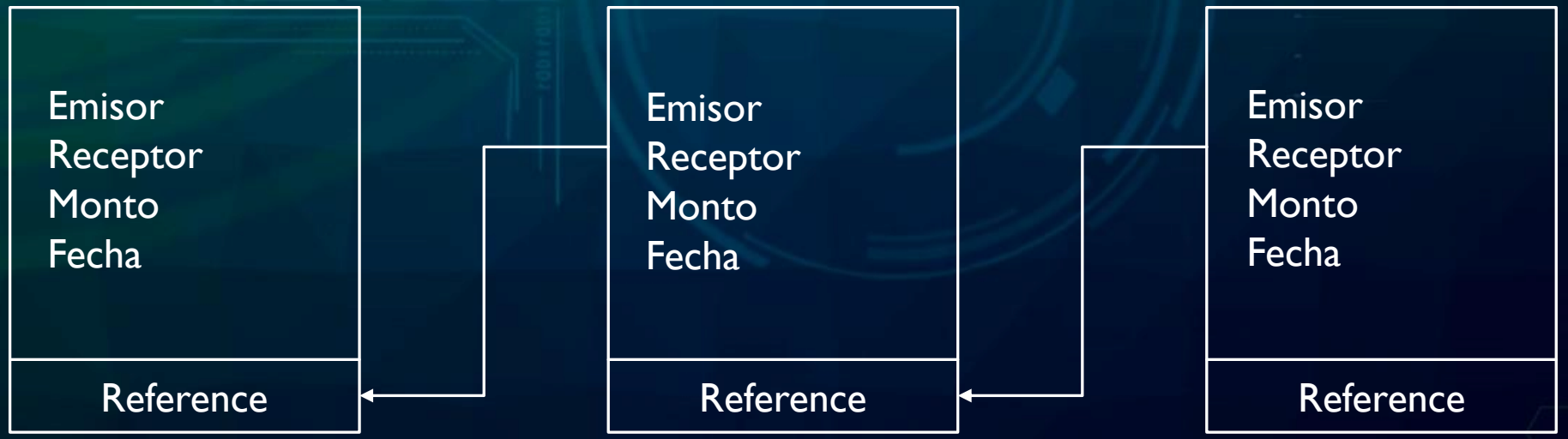
- Jose Garayar
- Roberth Ramos
- Juan Granados
- Carlos Villanueva

Docente:

- Heiner Sanchez

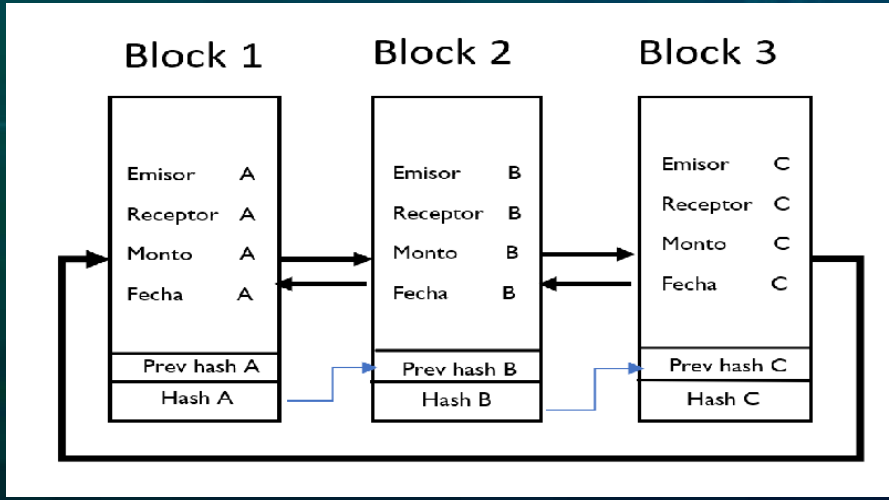
DESCRIPCIÓN DEL CASO DE ESTUDIO PLANTEADO POR EL GRUPO

El caso de estudio es un blockchain con indexación eficiente diseñado específicamente para almacenar y gestionar información de transacciones financieras. El blockchain actuará como un registro seguro de datos que contendrá detalles clave de las transacciones, como el emisor, el receptor, el monto y la fecha. Además, se implementarán estructuras de datos avanzadas para permitir una búsqueda rápida y eficiente de la información almacenada en el blockchain..



ESTRUCTURA DEL BLOCK CHAIN

El blockchain utiliza una estructura de datos conocida como cadena de bloques, donde cada bloque contiene un enlace a un bloque anterior, creando así una secuencia lineal de bloques. Para la estructura del block chain se utilizará el circular double list, en donde cada nodo representara un bloque que tiene un enlace tanto al nodo siguiente como al nodo anterior, y el último nodo de la lista está enlazado con el primer nodo, formando un ciclo.



Estructura del blockchain

```
int main() {
    Blockchain blockchain;

    std::vector<Transaction> transactions1;
    transactions1.push_back({1, "John", "Alice", 100.0, "2023-05-22"});
    transactions1.push_back({2, "Jose", "Carol", 50.0, "2023-05-23"});
    blockchain.addBlock(transactions1);

    std::vector<Transaction> transactions2;
    transactions2.push_back({4, "Jose", "Alice", 120.0, "2023-05-24"});
    transactions2.push_back({5, "Alice", "Bob", 100.0, "2023-05-24"});
    blockchain.addBlock(transactions2);

    std::vector<Transaction> transactions3;
    transactions3.push_back({4, "Karl", "Bob", 250.0, "2023-05-26"});
    transactions3.push_back({5, "John", "Karl", 350.0, "2023-05-27"});
    transactions3.push_back({7, "Juanjo", "Robert", 450.0, "2023-05-30"});
    blockchain.addBlock(transactions3);
```

ID EMISOR RECEPETOR IMPORTE FECHA

Bloque 1

Bloque 2

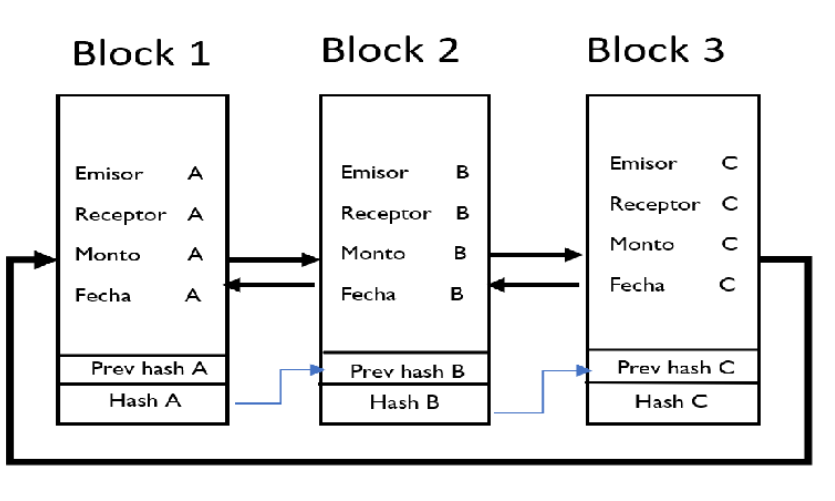
Bloque 3

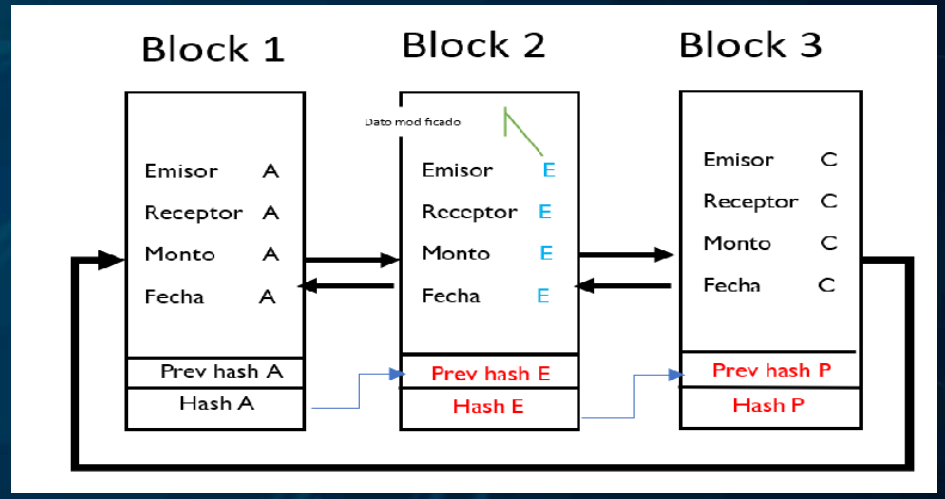
Estructura de la transacción

EFECTO CASCADA

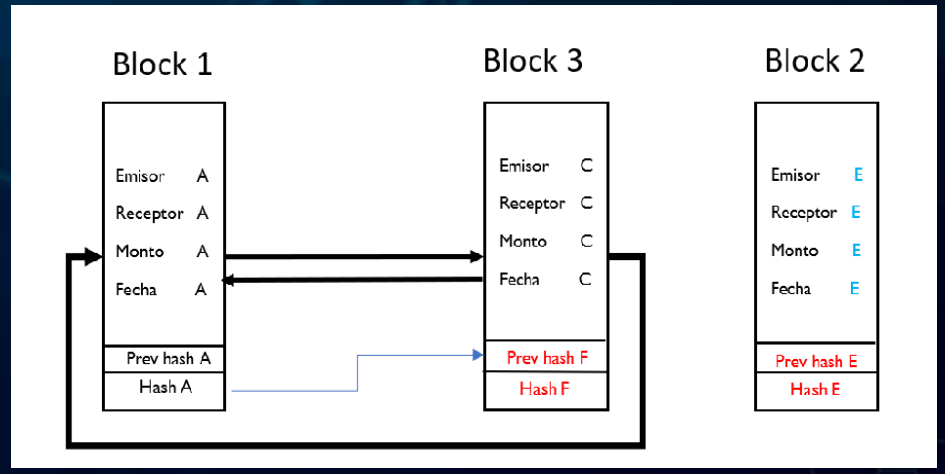
La identidad propia del bloque hijo cambia si la identidad del padre cambia. Para este escenario se implemento el metodo "cascadeEffect()" en la clase "Blockchain", que genera otros actualizados hashcode y previous hash del bloque a partir del padre actualizado (o borrado)

Blockchain inicial





Modificación de una Transacción X



Eliminación de un bloque

ESTRATEGIA PARA ASEGURAR LA INTEGRIDAD DE SU CONTENIDO E IMPLEMENTACIÓN DEL PROOF OF WORK.

Para el proyecto se escogió la técnica de Hashcash donde el número de ceros iniciales es de solo 4 ceros, ya que se busca un equilibrio entre seguridad y eficiencia. Cada bloque es identificable por un código hash, generado mediante un algoritmo de hash criptográfico a partir de toda la información que contiene el bloque. En nuestra clase "Block" se generara a traves del metodo "mineBlock()"

1

```
public:
    Block(int index, const vector<Transaction>& data, const string& previousHash) {
        this->index = index;
        this->data = data;
        this->previousHash = previousHash;
        this->nonce = 0;
        this->hash = mineBlock();

    }
```

2

```
    string mineBlock() {
        string targetPrefix = "0000";
        while (true) {
            hash = calculateHash(index, data, previousHash, nonce);
            if (hash.substr(0,4) == targetPrefix) {
                return hash;
            }
            nonce++;
        }
    }
```

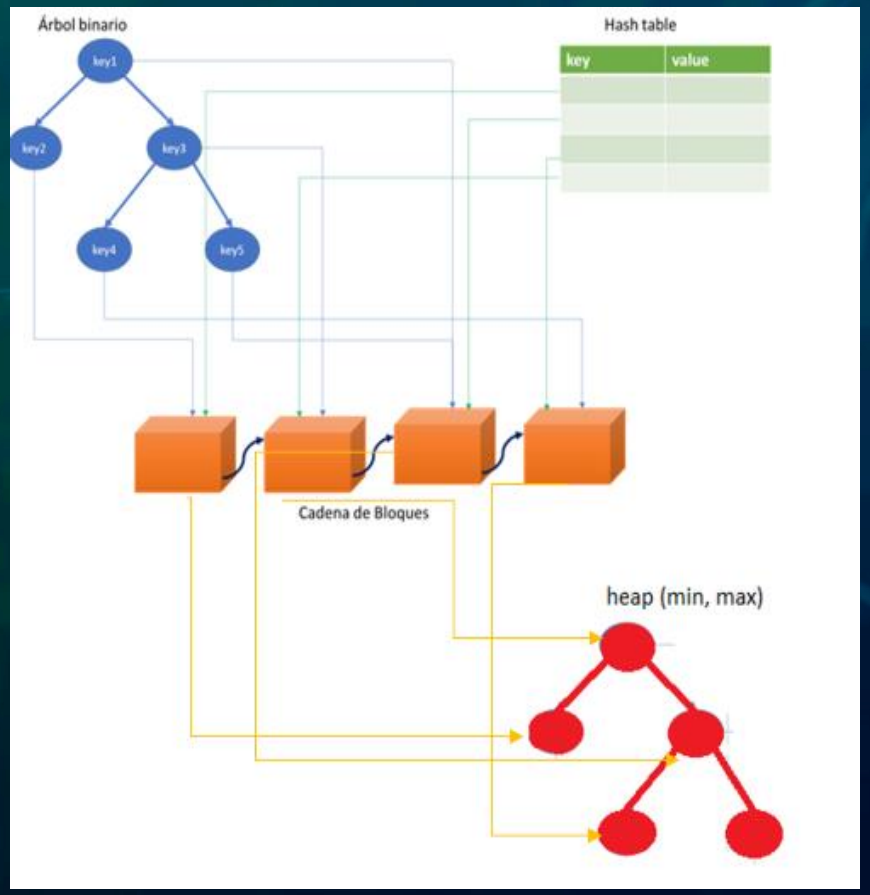
3

```
string calculateHash(int index, const vector<Transaction>& data, const string& previousHash, int nonce) const {
    stringstream ss;
    ss << index;
    for (const Transaction& transaction : data) {
        ss << transaction.idTransaccion << transaction.nombreOrigen << transaction.nombreDestino << transaction.importe << transactio
    }
    ss << previousHash << nonce << index;

    unsigned char hash[SHA256_DIGEST_LENGTH];
    SHA256_CTX sha256;
    SHA256_Init(&sha256);
    SHA256_Update(&sha256, ss.str().c_str(), ss.str().size());
    SHA256_Final(hash, &sha256);

    stringstream hashStream;
    for (int i = 0; i < SHA256_DIGEST_LENGTH; ++i) {
        hashStream << hex << setw(2) << setfill('0') << (int)hash[i];
    }

    return hashStream.str();
}
```

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

ESTRUCTURAS DE DATOS UTILIZADA EN SU APLICACIÓN DE ACUERDO A LOS CRITERIOS DE BÚSQUEDA.

Dado que se requiere una aplicación transaccional que permita a un usuario registrar transacciones bancarias de manera segura en el Blockchain, para luego aplicar búsquedas eficientes usando diversas estructuras de datos como mecanismos de indexación para diferentes criterios de búsqueda. Utilizaremos para nuestra aplicacion las estructuras Hash table, Arbol binario BST , y Heap

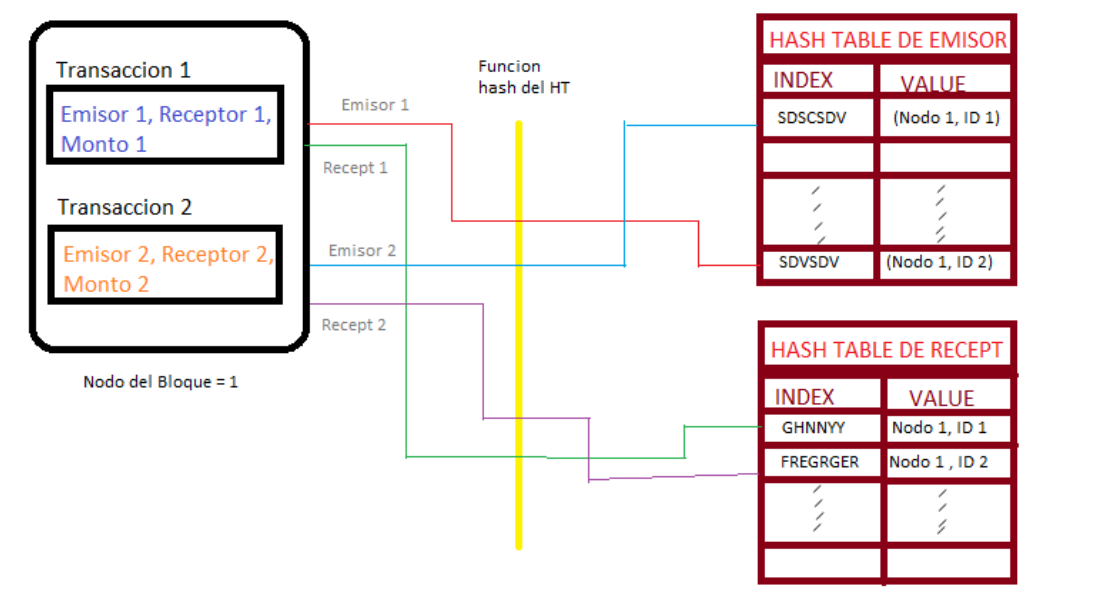


De acuerdo al tipo de filtrado requerido:

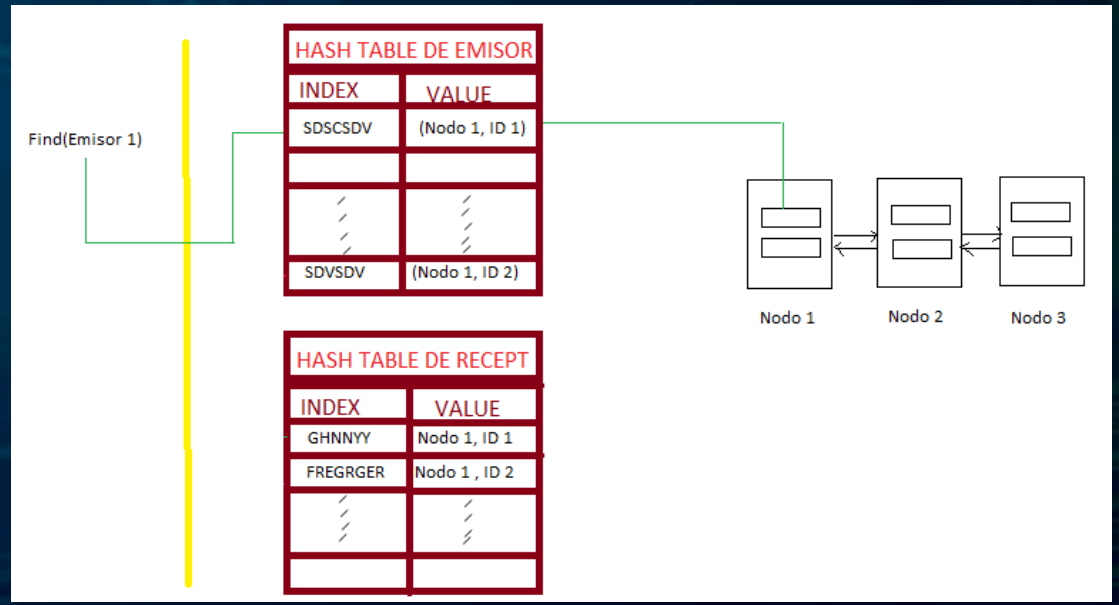
- Igual a X vector search(TK key)
- Entre X y Y vector range_search(TK begin, TK end)
- Inicia con vector start_with(string prefix)
- Máximo valor de Record max_value( )
- Mínimo valor de Record min_value( )

HASH TABLE

En el blockchain, utilizamos un hash table como estructura de indexación para mejorar la eficiencia de la búsqueda de información en la cadena de bloques. Un hash table es una estructura de datos que nos permitirá el acceso rápido a las transacciones especificas del blockchain



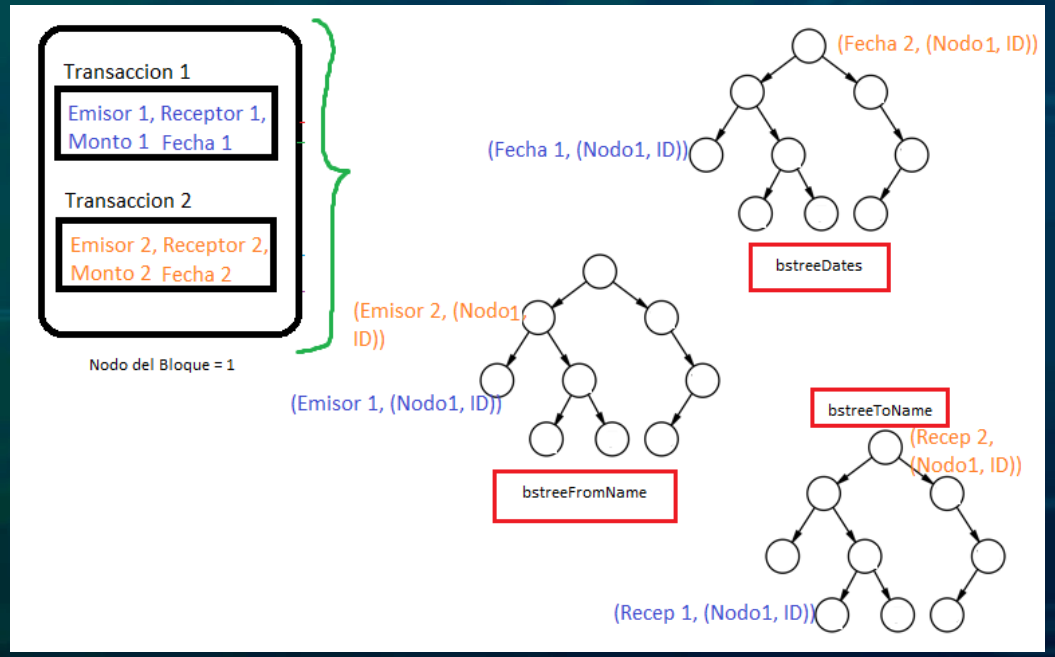
Hash table en la inserción de un bloque



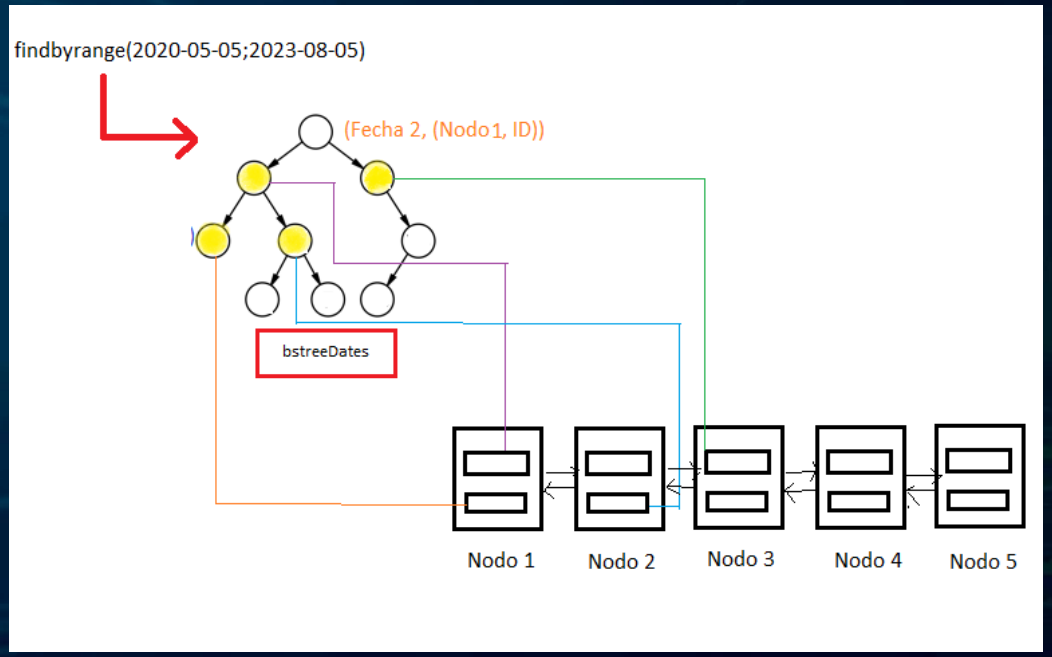
Búsqueda del Emisor o Receptor de una transacción

BST (BINARY SEARCH TREE)

Cuando se necesita buscar un rango de valores en una estructura de datos, como en el caso de buscar las transacciones realizadas en un intervalo específico, el BST se destaca debido a que mantienen sus elementos ordenados.



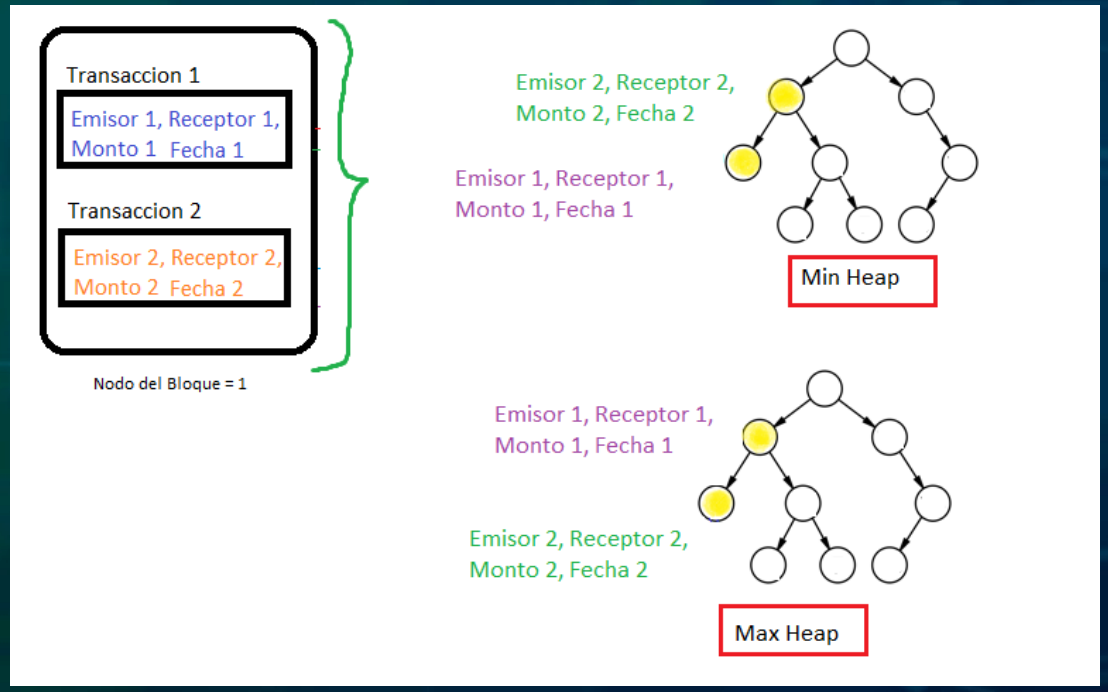
BST en la inserción de un bloque



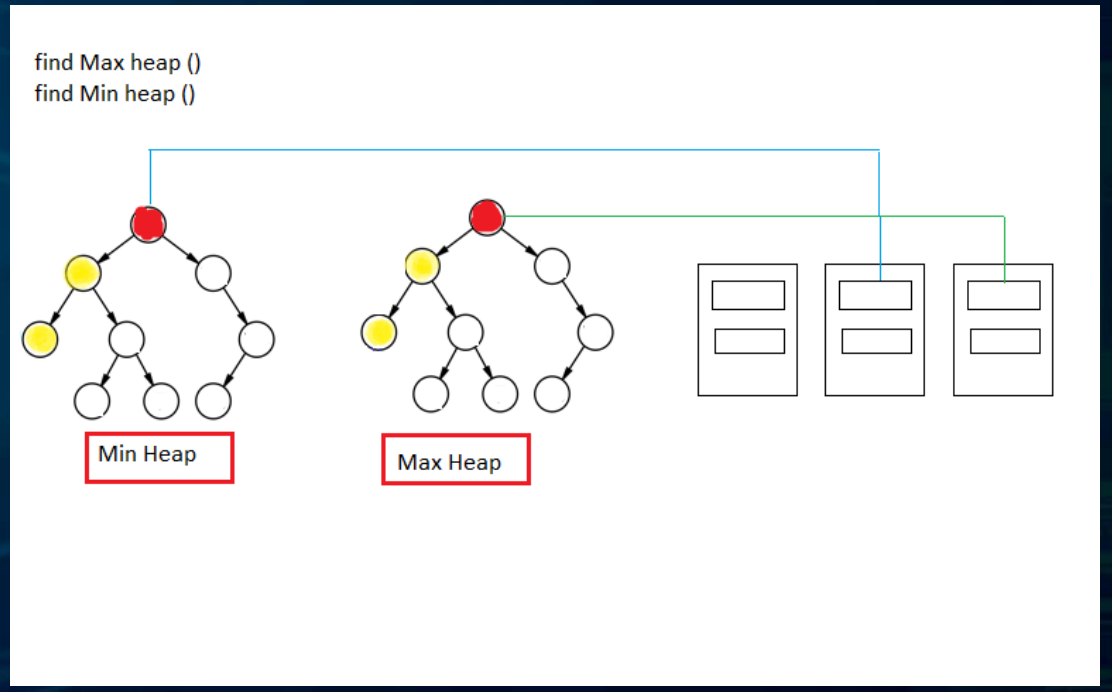
Búsqueda de transacciones por rango de fecha

HEAP

Para realizar búsqueda del máximo y mínimo importe transferido se utilizó la estructura de datos Heap, MaxHeap y MinHeap para obtener el máximo y mínimo importe respectivamente



HEAP en la inserción de un bloque



Búsqueda de máximo y mínimo importe

ANÁLISIS DE COMPLEJIDAD

Se muestra una tabla comparativa de analisis de complejidad de los metodos del Blockchain con indices vs sin indices

Método del Blockchain	Sin indice O( )	Con indice O( )
addBlock	O(1)	O(lgn)
deleteBlock	O(n)	O(n)
updateDataBlock	O(n)	O(n) -
cascadeEffect	O(n)	O(n)

En la siguiente tabla se muestra la complejidad O ( ) para los siguientes metodos de busqueda

Método de busqueda del Blockchain	O( )
findTransactionsByFromName	O(k)
findTransactionsByToName	O(k)
findTransactionsByFromNameBeginWith	O(lgn)
findTransactionsByRangeof	O(lgn)
findMaxTransaction	O(1)
findMinTransaction	O(1)

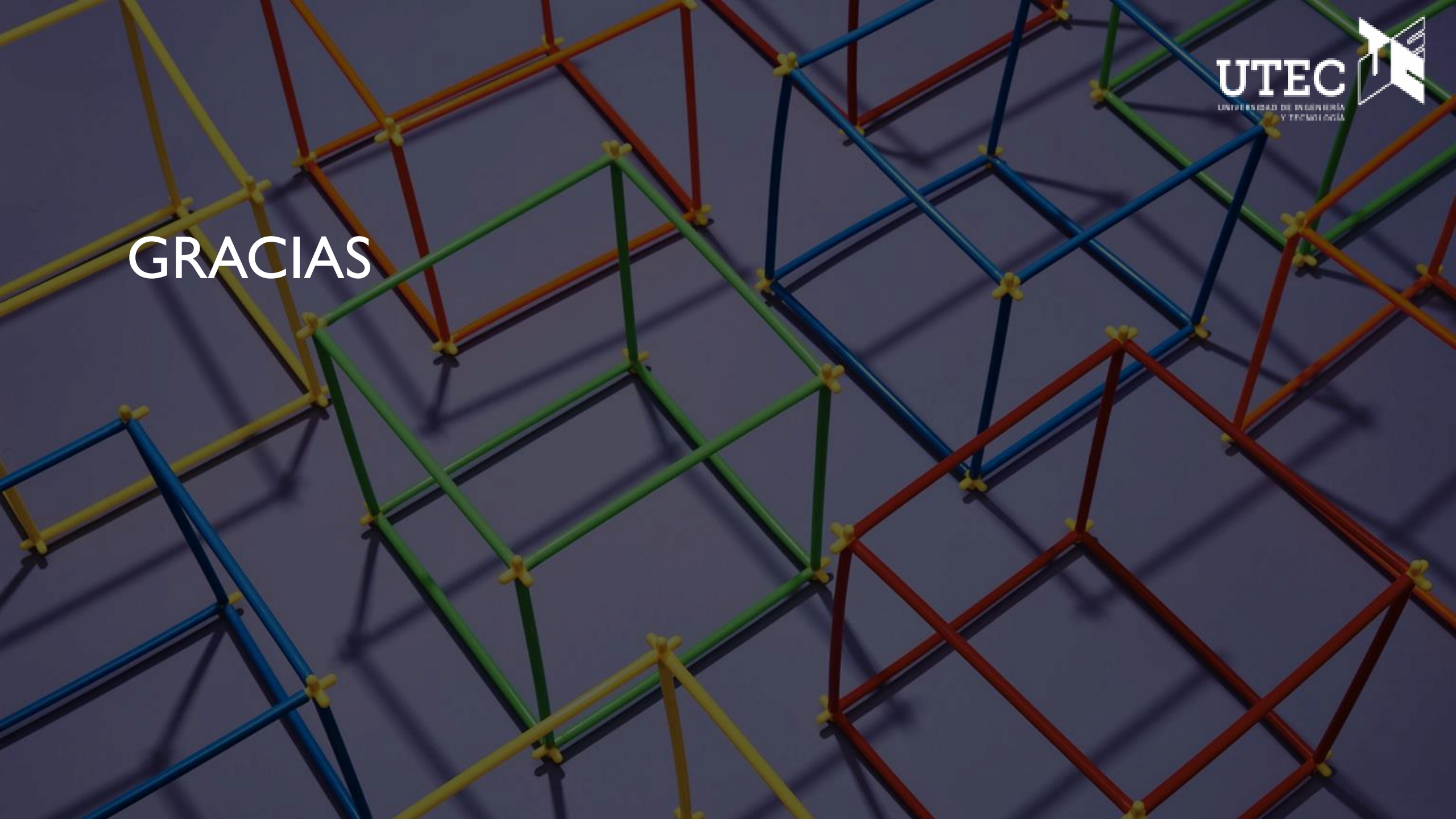
PRODUCTO FINAL

```
MENU PRINCIPAL

1. Insertar, borrar o actualizar Bloque
2. Generar Blockchain desde archivo
3. Mostrar Blockchain
4. Buscar por Emisor o Receptor
5. Buscar por Rango
6. Buscar por Monto Máximo
7. Buscar por Monto Mínimo
8. Recalculo en cascada
9. Salir
```

```
      Mostrar Blockchain


  Index: 0
  Nonce: 58844
  Transactions:
       ID Transaccion: 1
       Nombre Emisor:Xander Welch          Nombre Receptor:Jael Thomas
       Importe: 62887.960000               Fecha:2022-11-16
       ID Transaccion: 2
       Nombre Emisor:Travis Pittman        Nombre Receptor:Zane Dillon
       Importe: 88648.310000               Fecha:2024-01-30
       ID Transaccion: 3
       Nombre Emisor:Tanisha Roman         Nombre Receptor:Wayne Singleton
       Importe: 22266.880000               Fecha:2023-03-09
       ID Transaccion: 4
       Nombre Emisor:Arsenio Anderson      Nombre Receptor:Kareem Chavez
       Importe: 26942.580000               Fecha:2022-07-16
       ID Transaccion: 5
       Nombre Emisor:Neve Knox             Nombre Receptor:Colby Hammond
       Importe: 514.020000                 Fecha:2022-10-09
  Previous Hash: 0000000000000000000000000000000000000000000000000000000000000000
  Hash: 00000cd96d035d9599d187362f972fdd5c6199828cd5410f75c6de49d358aa87


  Presione una tecla [<] para retroceder [>] para avanzar o [Q] para regresar a menu
```

```
      Buscar Monto Máximo

  La transacción con importe máximo registrado es:

  ID Transaccion: 8
  Nombre Emisor: Kristen Cochran
  Nombre Receptor: Melanie Vincent
  Importe: 97460.010000
  Fecha: 2023-04-11


  Presione cualquier tecla para regresar al menu
```

```
     BUSCAR POR RANGO DE FECHAS

Fecha Inicial: 2022-01-01
Fecha Final: 2023-05-05

Se han encontrado los siguientes resultados
     ID Transaccion: 17
     Nombre Emisor:Quintessa Barton      Nombre Receptor:Buckminster Hays
     Importe: 16284.510000               Fecha:2022-06-22

     ID Transaccion: 4
     Nombre Emisor:Arsenio Anderson      Nombre Receptor:Kareem Chavez
     Importe: 26942.580000               Fecha:2022-07-16

     ID Transaccion: 15
     Nombre Emisor:Melinda Hale          Nombre Receptor:Isabella Parker
     Importe: 31088.250000               Fecha:2022-07-24

     ID Transaccion: 11
     Nombre Emisor:Ezra Horne            Nombre Receptor:Abel Deleon
     Importe: 96254.700000               Fecha:2022-10-08

     ID Transaccion: 5
     Nombre Emisor:Neve Knox             Nombre Receptor:Colby Hammond
     Importe: 514.020000                 Fecha:2022-10-09


Presione una tecla para continuar o [Q] para regresar a menu
```



GRACIAS