**Paso 1:** Escribimos en Google Visual Studio Code y seleccionamos donde dice “Download”.



**Paso 2:** Seleccionamos el sistema operativo que tenemos y lo descargamos.



**Paso 3:** Al darle clic nos descargará un .exe, al cual le daremos clic encima.



**Paso 4:** Lee y acepta el acuerdo de licencia. Haz clic en Next para continuar.



**Paso 5:** Puedes cambiar la ubicación de la carpeta de instalación o mantener la configuración predeterminada. Haz clic en Next para continuar.



**Paso 6:** Elige si deseas cambiar el nombre de la carpeta de accesos directos en el menú Inicio o si no deseas instalar accesos directos en absoluto. Haz clic en Next.



**Paso 7:** Selecciona las tareas adicionales, por ej. crear un icono en el escritorio o añadir opciones al menú contextual de Windows Explorer. Haz clic en Next.Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

**Paso 8:** Haz clic en Install para iniciar la instalación.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

**Paso 9:** El programa está instalado y listo para usar. Haz clic en Finish para finalizar la instalación y lanzar el programa.



**Paso 10**: ahora vamos a descargar node js, entramos a Google y buscamos node js download.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

**Paso 11**: escogemos la primera opción

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

**Paso 12**: ahora elegimos nuestro sistema operativo y los bits

**Imagen que contiene Escala de tiempo

Descripción generada automáticamente**

**Paso 13**: abrimos el instalador y se nos mostrara una pestaña, apretamos “next”

Texto

Descripción generada automáticamente

**Paso 15**: tendremos que aceptar las licencias de node js marcamos el cuadrado de abajo a la derecha y next

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

**Paso 15**: elegimos el lugar del archivo y next

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

**Paso 16**: en esta ventana podremos customisar el node js con diferentes paquetes apretamos next

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

**Paso 17**: aquí nos piden permiso para instalar diferentes herramientas del node js apretamos next

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

**Paso 18**: finalmente apretamos install para comenzar la instalación

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

**Paso 19**: esperar la instalación

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

**Paso 20:**Creamos una carpeta raíz. En nuestro caso la llamaremos “blockchain”.



**Paso 21**:En el buscador de archivos escribimos “cmd” y se nos abrirá la consola.

Tabla

Descripción generada automáticamente con confianza media

**Paso 22:**Escribimos en la consola “code .”.

Texto

Descripción generada automáticamente

**Paso 23**: nos abre esta pestaña

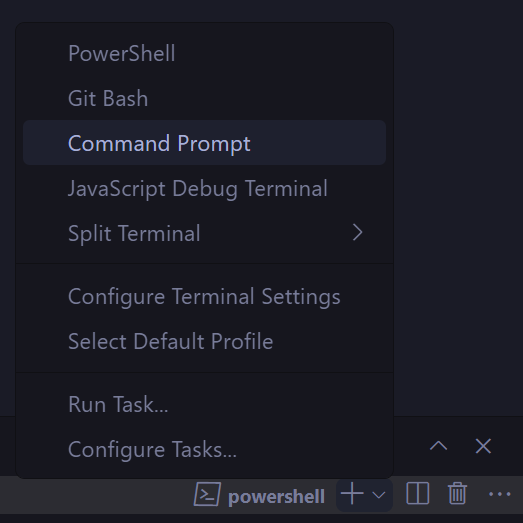
Captura de pantalla de un celular

Descripción generada automáticamente

**Paso 24**: en la barra superior del visual a la derecha encontraremos la opción para abrir la consola el visual.

****

**Paso 25**: una vez abierta la consola abrimos el menú desplegable del powershell y seleccionamos “Command Prompt”



**Paso 26**: eliminamos el powershell y nos quedamos con el cmd

Interfaz de usuario gráfica, Aplicación, Teams

Descripción generada automáticamente

**Paso 27**: ahora en la consola escribimos “npm init --yes” con esto iniciamos el node en nuestro proyecto

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

**Paso 28**: instalamos las dependencias del proyecto “npm i js-sha256”



**Paso 29**: creamos una carpeta llamada “app” en la carpeta raíz

Interfaz de usuario gráfica, Aplicación

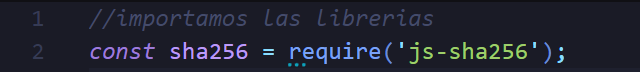
Descripción generada automáticamente

**Paso 30**: dentro de la carpeta app creamos el archivo “blockchain.js”

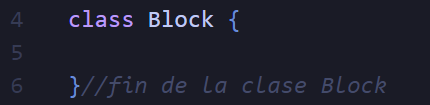
Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

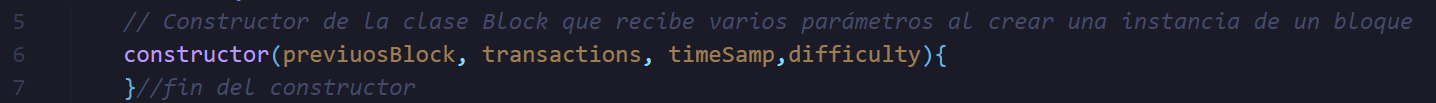
**Paso 31:** importamos las librerias

****

**Paso 32**: creamos la clase llamada “Block”



**Paso 33**: creamos el constructor de la clase Block con cuatro parámetros “previuosBlock” que es el bloque anterior, “transaction” que son transacciones, “timeSamp” que es el momento en que se crea el bloque y “dificulty” que es el poder computacional que se requiere para crear el bloque



**Paso 34**: El constructor acepta información sobre el bloque, como el bloque anterior, las transacciones y la dificultad de minería. El método createNonce() se utiliza para calcular un valor nonce que, cuando se combina con otros datos del bloque y se calcula el hash, cumple con la dificultad de minería requerida para agregar el bloque a la cadena

Texto

Descripción generada automáticamente

**Paso 35:** ahora creamos la función “createNonce()”

Texto

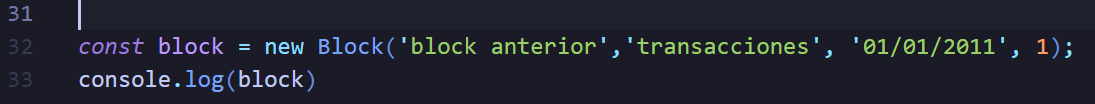
Descripción generada automáticamente

**Paso 36**: se inicializa el nonce a 0 y se crea una cadena llamada targetZeros que consiste en una serie de ceros igual a la dificultad establecida (línea 17 y 18). se inicia un bucle while (true) que continuará hasta que se encuentre un hash válido, se calcula un hash concatenando el nonce actual, el hash del bloque anterior y las transacciones, Se verifica si el hash calculado comienza con la cadena de ceros targetZeros. Si es así, significa que se ha encontrado un hash válido que cumple con la dificultad Si el hash no cumple con la dificultad, se incrementa el nonce y se continúa con el siguiente intento. Una vez que se encuentra un hash válido devuelve el valor del nonce calculado.

Texto

Descripción generada automáticamente

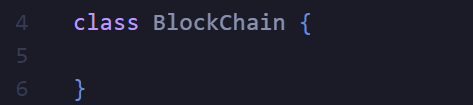
**Paso 37**: ahora hacemos una prueba, creamos un objeto y lo mostramos por consola

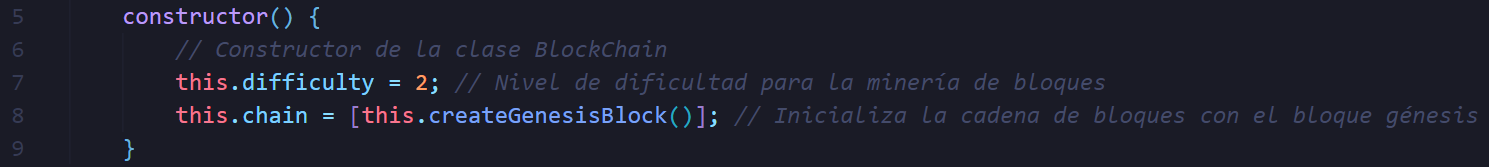
****

**Paso 38**: nos vamos a la consola y escribimos “node app/blockchain.js”, en este caso el loop tuvo que iterar 11 veces par obtener el hash

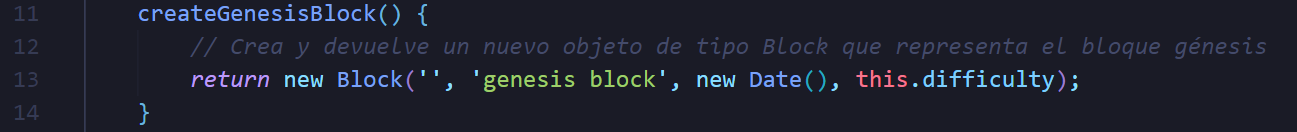


**Paso 39**: borramos la línea 32 y 33, ahora en la línea 4 creamos una nueva clase “BlockChain”

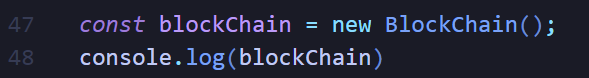


**Paso 40**: creamos el constructor de la clase BlockChain

**Paso 41**: dentro de la clase BlockChain creamos la función “createGenesisBlock” la cual crea un objeto tipo block que será el bloque genesis

****

**Paso 42**: ponemos a prueba el fragmento de código que acabamos de codificar creando un objeto BlockChain y mostrándolo en consola



**Paso 43:** escribimos esto en la consola

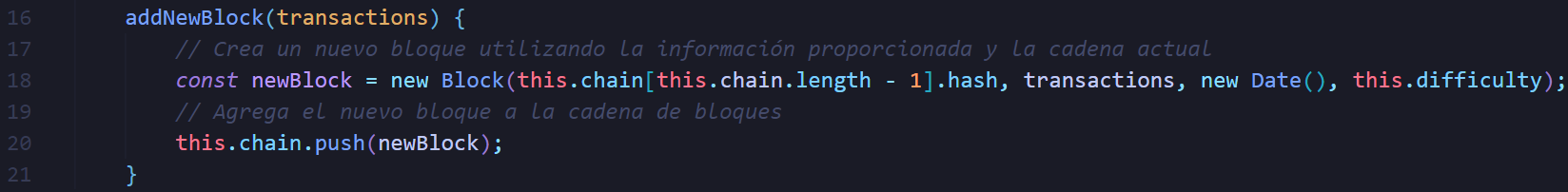


**Paso 44:** aquí tenesmo la dificultad y la cadena de caracteres junto con el bloque creado

Texto

Descripción generada automáticamente

**Paso 45**: creamos un objeto Block y le damos como parámetros el chain actual con el ultimo bloque creado y el hash del bloque creado además le damos las transactions la fecha y la dificultad y por ultimo agregamos el bloque a la cadena

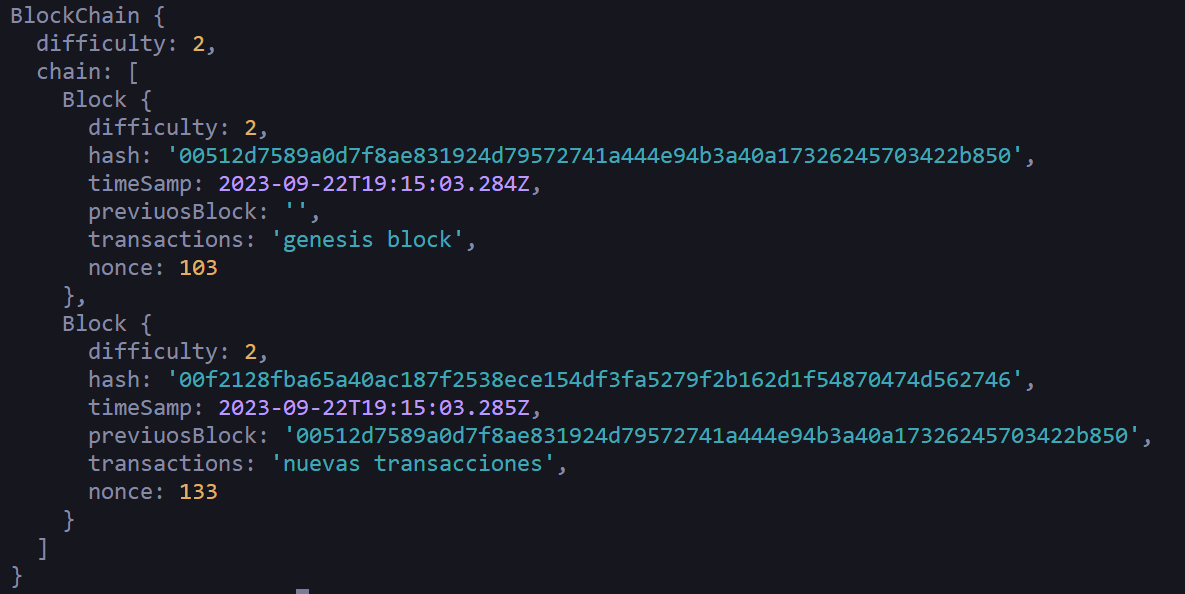
****

**Paso 46**: ponemos a prueba el fragmento de código que acabamos de codificar

Texto

Descripción generada automáticamente

**Paso 47**: ahora tenemos dos bloques en la cadena con diferente información como el hash



**Paso 48**: ahora validaremos la información creamos una constante hash los valores del bloque y validamos si esa información coincide con la del bloque

