

The following steps describe how to create a blockchain entirely in Python.

LIBRARY IMPORT

```
In [1]: # Importing the necessary library
import hashlib
```

CREATING A BLOCK CLASS

```
In [2]: # Creating a block class
class Block:
    # Creating a constructor for the block class
    def __init__(self, data, previous_hash):
        self.data=data
        self.previous_hash=previous_hash
        self.hash=self.calculate_hash()
    # Creating a method to calculate hash using SHA-256 encryption system
    def calculate_hash(self):
        sha=hashlib.sha256() # SHA-256 = Secure Hash Algorithm 256 bit
        sha.update(self.data.encode("utf-8")) # Transform the data in a Unicode format in 8 bit
        return sha.hexdigest() # Returns the hash of a given input as a hexadecimal string
```

CREATING THE BLOCKCHAIN CLASS

```
In [3]: # Creating a blockchain class
class Blockchain:
    # Creating a constructor for the blockchain class
    def __init__(self):
        self.chain=[] # Initialize the blockchain block list
        self.chain.append(self.genesis_block()) # Insert the genesis block as the first one
    # Creating a method to creates the "Genesis block", meaning what is the first block of a blockchain
    def genesis_block(self):
```

```

    return Block("The genesis block", "0")
# Creating a method to create a new block and adds it to for the blockchain (the List)
def add_new_block(self, data):
    previous_block=self.chain[-1]
    new_block=Block(data, previous_block.hash)
    self.chain.append(new_block)

```

TESTING THE BLOCKCHAIN

```

In [4]: # Testing the blockchain
        blockchain=Blockchain()

```

ADDING DATA TO THE BLOCKCHAIN

```

In [5]: # Adding data to the blockchain
def num_data_blocks_blockchain():
    # Adding a specific number of blocks to the blockchain
    num_input_blocks=int(input("How many blocks do you want to add to the blockchain: "))
    print()
    # Entering individual data for each block
    for i in range(num_input_blocks):
        data=input(f"Enter the data for the block number {i+1}: ")
        blockchain.add_new_block(data)
    print()

```

COUNT OF BLOCKS PRESENT IN THE BLOCKCHAIN

```

In [6]: # Counting the number of blocks present in the blockchain
def count_blocks_blockchain():
    num_tot_blocks=0
    for block in blockchain.chain:
        num_tot_blocks=num_tot_blocks+1
    print(f"In this blockchain there is/are in total {num_tot_blocks} block(s): (1 genesis block + {num_tot_blocks-1} normal b

```

PRINTING THE FINAL BLOCKCHAIN

```
In [7]: # Printing all the elements of each block of the blockchain
def block_element_info_blockchain():
    i=-1
    print(f"In this blockchain there is/are the block(s):")
    print()
    for block in blockchain.chain:
        i+=1
        print(f"The block number {i} is composed of:")
        print(f"The data is: {block.data}")
        print(f"The previous hash is: {block.previous_hash}")
        print(f"The hash of the block is: {block.hash}")
        print()
```

FINAL BLOCKCHAIN

```
In [8]: # Execution of all created functions
def final_blockchain():
    num_data_blocks_blockchain()
    block_element_info_blockchain()
    count_blocks_blockchain()
    print()
    input("Press Enter to exit the program... ")
```

```
In [9]: # Executing the final summary function of the other functions
final_blockchain()
```

In this blockchain there is/are the block(s):

The block number 0 is composed of:

The data is: The genesis block

The previous hash is: 0

The hash of the block is: a60558ffeb06932db4cb691d8304baf6457336e641be41ac6c6a21dc332d4496

The block number 1 is composed of:

The data is: a

The previous hash is: a60558ffeb06932db4cb691d8304baf6457336e641be41ac6c6a21dc332d4496

The hash of the block is: ca978112ca1bbdcafac231b39a23dc4da786eff8147c4e72b9807785afee48bb

The block number 2 is composed of:

The data is: b

The previous hash is: ca978112ca1bbdcafac231b39a23dc4da786eff8147c4e72b9807785afee48bb

The hash of the block is: 3e23e8160039594a33894f6564e1b1348bbd7a0088d42c4acb73eeaed59c009d

The block number 3 is composed of:

The data is: c

The previous hash is: 3e23e8160039594a33894f6564e1b1348bbd7a0088d42c4acb73eeaed59c009d

The hash of the block is: 2e7d2c03a9507ae265ecf5b5356885a53393a2029d241394997265a1a25aefc6

The block number 4 is composed of:

The data is: d

The previous hash is: 2e7d2c03a9507ae265ecf5b5356885a53393a2029d241394997265a1a25aefc6

The hash of the block is: 18ac3e7343f016890c510e93f935261169d9e3f565436429830faf0934f4f8e4

The block number 5 is composed of:

The data is: e

The previous hash is: 18ac3e7343f016890c510e93f935261169d9e3f565436429830faf0934f4f8e4

The hash of the block is: 3f79bb7b435b05321651daefd374cdc681dc06faa65e374e38337b88ca046dea

In this blockchain there is/are in total 6 block(s): (1 genesis block + 5 normal block(s))