PRACTICAL 4

AIM:

Study and Configure Geth over windows or Ubuntu. Perform following tasks:

a. Build Your Own Private Ethereum Blockchain.
b. Create Genesis Block:
c. Initialize the Genesis block:
d. Create Boot node
e. Start Private Network
f. Launch Ethereum Wallet:
g. Create Address:
h. Start Mining

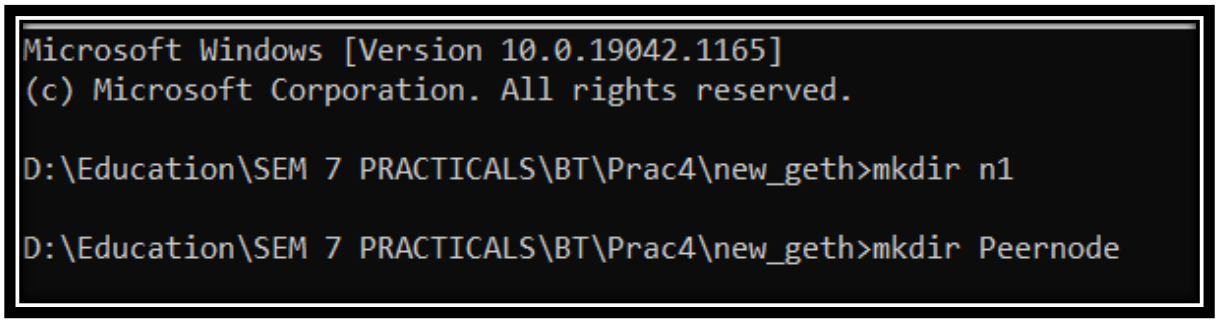
STEPS OF IMPLEMENTATION:

First we will create a main directory. Here we're naming it "new_geth" but any name will work. Then in cmd, we will navigate to it.

```
cd new_geth
```

Then we will make 2 directories inside that folder. We're naming then n1 and peernode.

```
mkdir n1
mkdirPeernode
```



Then we will create new accounts in both the folders with following commands.

```
geth --datadir ./n1 account new
geth --datadir ./Peernode account new
```

```
D:\Education\SEM 7 PRACTICALS\BT\Prac4\new_geth>geth --datadir ./n1 account new
INFO [08-20|11:53:00.959] Maximum peer count                       ETH=50 LES=0 total=50
Your new account is locked with a password. Please give a password. Do not forget this password.
Password:
Repeat password:

Your new key was generated

Public address of the key:   0x24768647A631B3e40d5D3EE2600E474f5ED8E8A2
Path of the secret key file: n1\keystore\UTC--2021-08-20T06-23-04.895238600Z--24768647a631b3e40d5d3ee2600e474f5ed8e8a2

- You can share your public address with anyone. Others need it to interact with you.
- You must NEVER share the secret key with anyone! The key controls access to your funds!
- You must BACKUP your key file! Without the key, it's impossible to access account funds!
- You must REMEMBER your password! Without the password, it's impossible to decrypt the key!
```

That would generate public address of both accounts.
We will create genesis.json file in both directories with following code.

```
{
"config": {
"chainId": 1234,
"homesteadBlock": 0,
"eip150Block": 0,
"eip150Hash": "0x0000000000000000000000000000000000000000000000000000000000000000",
"eip155Block": 0,
"eip158Block": 0,
"byzantiumBlock": 0
},
"difficulty": "400",
"gasLimit": "2000000",
"alloc": {
"<public-address-of-the-key>": {
"balance": "500000000000000000000"
}
}
}
```

Then we will initiate both the genesis blocks.

```
geth --datadir ./n1 init n1/genesis.json
geth --datadir ./PeernodeinitPeernode/genesis.json
```

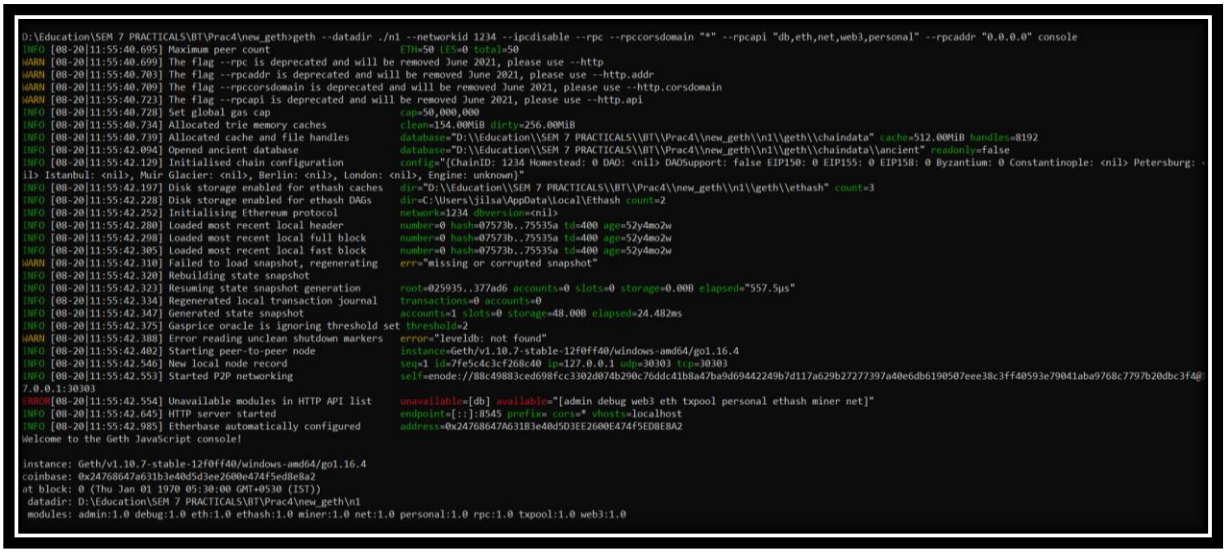
```
D:\Education\SEM 7 PRACTICALS\BT\Prac4\new_geth>geth --datadir ./n1 init n1/genesis.json
INFO [08-20|11:55:25.742] Maximum peer count                       ETH=50 LES=0 total=50
INFO [08-20|11:55:25.782] Set global gas cap                       cap=50,000,000
INFO [08-20|11:55:25.791] Allocated cache and file handles         database="D:\\Education\\SEM 7 PRACTICALS\\BT\\Prac4\\new_geth\\n1\\geth\\chaindata" cache=16.00MiB handles=16
INFO [08-20|11:55:25.978] Writing custom genesis block
INFO [08-20|11:55:25.989] Persisted trie from memory database      nodes=1 size=149.00B time=0s gcnodes=0 gcsize=0.00B gctime=0s livenodes=1 livesize=0.00B
INFO [08-20|11:55:26.042] Successfully wrote genesis state         database=chaindata hash=07573b..75535a
INFO [08-20|11:55:26.065] Allocated cache and file handles         database="D:\\Education\\SEM 7 PRACTICALS\\BT\\Prac4\\new_geth\\n1\\geth\\lightchaindata" cache=16.00MiB handles=16
INFO [08-20|11:55:26.251] Writing custom genesis block
INFO [08-20|11:55:26.266] Persisted trie from memory database      nodes=1 size=149.00B time=1.0246ms gcnodes=0 gcsize=0.00B gctime=0s livenodes=1 livesize=0.00B
INFO [08-20|11:55:26.289] Successfully wrote genesis state         database=lightchaindata hash=07573b..75535a

D:\Education\SEM 7 PRACTICALS\BT\Prac4\new_geth>geth --datadir ./Peernode init Peernode/genesis.json
INFO [08-20|11:55:32.154] Maximum peer count                       ETH=50 LES=0 total=50
INFO [08-20|11:55:32.160] Set global gas cap                       cap=50,000,000
INFO [08-20|11:55:32.163] Allocated cache and file handles         database="D:\\Education\\SEM 7 PRACTICALS\\BT\\Prac4\\new_geth\\Peernode\\geth\\chaindata" cache=16.00MiB handles=16
INFO [08-20|11:55:32.292] Writing custom genesis block
INFO [08-20|11:55:32.305] Persisted trie from memory database      nodes=1 size=149.00B time="87.6µs" gcnodes=0 gcsize=0.00B gctime=0s livenodes=1 livesize=0.00B
INFO [08-20|11:55:32.337] Successfully wrote genesis state         database=chaindata hash=175976..cd48c4
INFO [08-20|11:55:32.363] Allocated cache and file handles         database="D:\\Education\\SEM 7 PRACTICALS\\BT\\Prac4\\new_geth\\Peernode\\geth\\lightchaindata" cache=16.00MiB handles=16
INFO [08-20|11:55:32.514] Writing custom genesis block
INFO [08-20|11:55:32.528] Persisted trie from memory database      nodes=1 size=149.00B time=0s       gcnodes=0 gcsize=0.00B gctime=0s livenodes=1 livesize=0.00B
INFO [08-20|11:55:32.563] Successfully wrote genesis state         database=lightchaindata hash=175976..cd48c4
```

Then we will start interactive javascript environment to interact with our newly created

blockchain.

```
geth --datadir ./n1 --networkid 1234 --ipcdisable --rpc --rpccorsdomain "*" --rpcapi "db,eth,net,web3,personal" --rpcaddr "0.0.0.0" console
```



To check balance in our account, we will simply type the following command while the background process is still running.

```
eth.getBalance(web3.eth.accounts[0])
```

Admin object exposes methods to interact with the RPC APIs. To get information about enode, we will write following command.

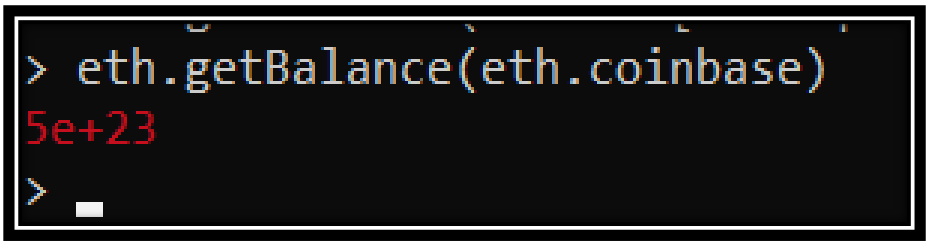
```
admin.nodeInfo.enode
```

We will copy this enode address and Request adding a new remote node to the list of tracked static nodes.

```
admin.addPeer("paste the enode address")
```

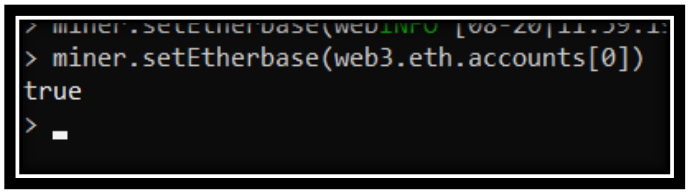
To check your the etherbase account balance, we will use…

```
eth.getBalance(eth.coinbase)
```



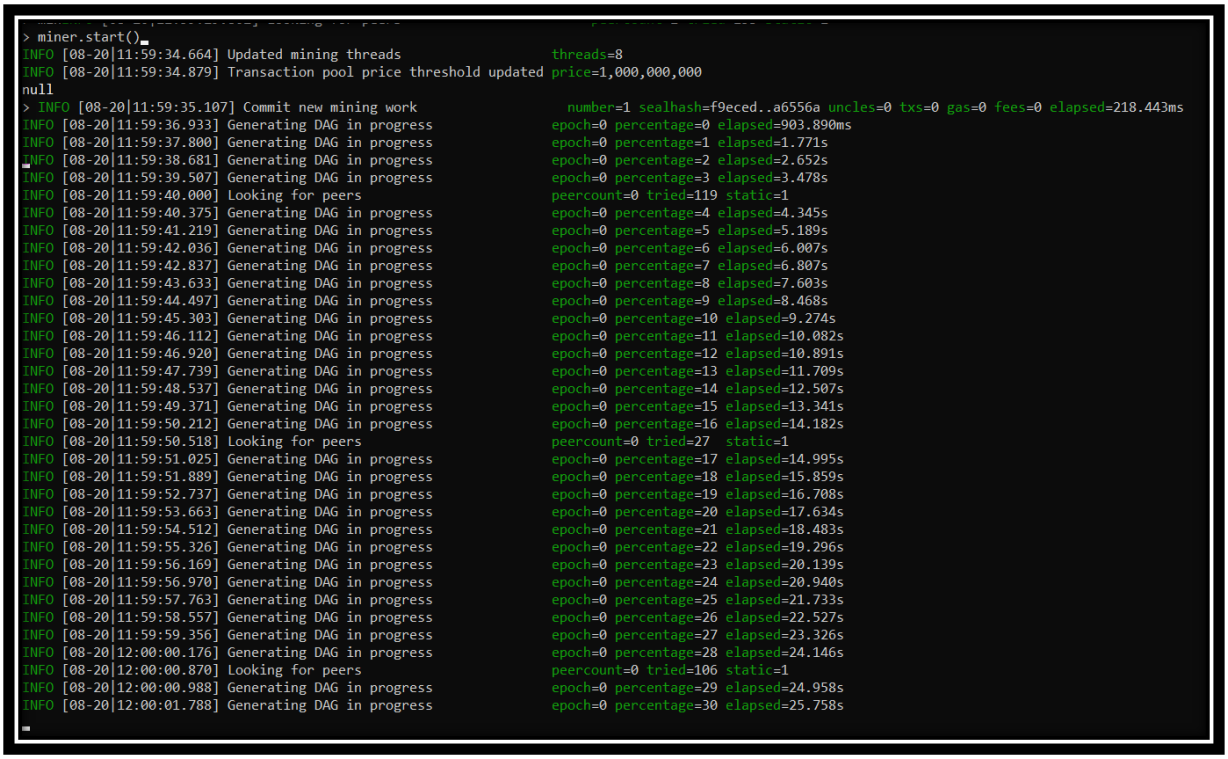
Now we will set the etherbase, where mining rewards will go.

```
miner.setEtherbase(web3.eth.accounts[0])
```



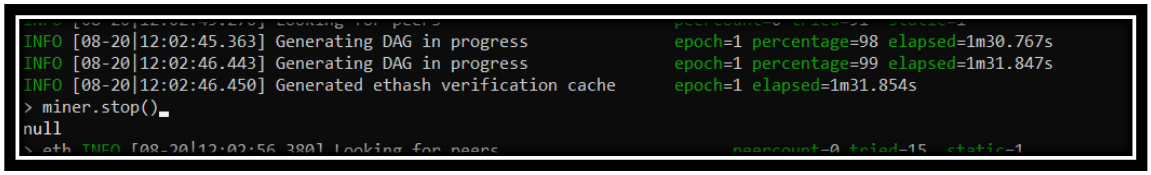
Then finally, we are ready to mine some Ethereum now.

```
miner.start()
```



To stop that process, we can use…

```
miner.stop()
```



Now we will again check balance with following command. Ideally balance will be less than previous balance i.e. the transaction fees deducted.

```
eth.getBalance(eth.coinbase)
```

CONCLUSION:

In this practical, we used command prompt and geth to create our private Ethereum blockchain and we mined some Ethereum using it.