PRACTICAL 9

AIM:

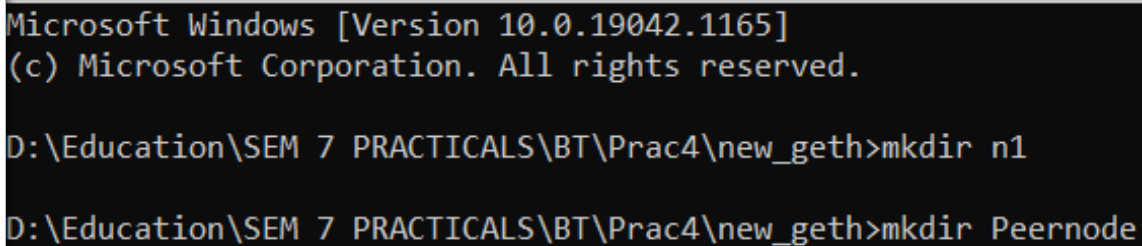Build the private blockchain network using Geth.

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.
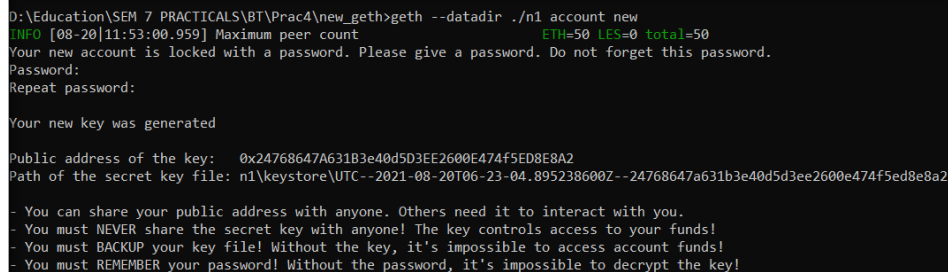mkdir n1
mkdir Peernode

```
Microsoft Windows [Version 10.0.19042.1165]
(c) Microsoft Corporation. All rights reserved.

D:\Education\SEM 7 PRACTICALS\BT\Prac4\new_geth>mkdir n1

D:\Education\SEM 7 PRACTICALS\BT\Prac4\new_geth>mkdir Peernode
```

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": "50000000000000000000000000"
}
}
}
```

Then we will initiate both the genesis blocks.

geth --datadir ./n1 init n1/genesis.json
geth --datadir ./Peernode init Peernode/genesis.json

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])

```
> miner.setEtherbase(webINFO [08-20|11:59.1
> miner.setEtherbase(web3.eth.accounts[0])
true
>
```

Then finally, we are ready to mine some Ethereum now.
miner.start()

```
> miner.start()
INFO [08-20|11:59:34.664] Updated mining threads              threads=8
INFO [08-20|11:59:34.879] Transaction pool price threshold updated price=1,000,000,000
null
> INFO [08-20|11:59:35.107] Commit new mining work              number=1 sealhash=f9eced..a6556a uncles=0 txs=0 gas=0 fees=0 elapsed=218.443ms
INFO [08-20|11:59:36.933] Generating DAG in progress          epoch=0 percentage=0 elapsed=903.890ms
INFO [08-20|11:59:37.800] Generating DAG in progress          epoch=0 percentage=1 elapsed=1.771s
INFO [08-20|11:59:38.681] Generating DAG in progress          epoch=0 percentage=2 elapsed=2.652s
INFO [08-20|11:59:39.507] Generating DAG in progress          epoch=0 percentage=3 elapsed=3.478s
INFO [08-20|11:59:40.000] Looking for peers                   peercount=0 tried=119 static=1
INFO [08-20|11:59:40.375] Generating DAG in progress          epoch=0 percentage=4 elapsed=4.345s
INFO [08-20|11:59:41.219] Generating DAG in progress          epoch=0 percentage=5 elapsed=5.189s
INFO [08-20|11:59:42.036] Generating DAG in progress          epoch=0 percentage=6 elapsed=6.007s
INFO [08-20|11:59:42.837] Generating DAG in progress          epoch=0 percentage=7 elapsed=6.807s
INFO [08-20|11:59:43.633] Generating DAG in progress          epoch=0 percentage=8 elapsed=7.603s
INFO [08-20|11:59:44.497] Generating DAG in progress          epoch=0 percentage=9 elapsed=8.468s
INFO [08-20|11:59:45.303] Generating DAG in progress          epoch=0 percentage=10 elapsed=9.274s
INFO [08-20|11:59:46.112] Generating DAG in progress          epoch=0 percentage=11 elapsed=10.082s
INFO [08-20|11:59:46.920] Generating DAG in progress          epoch=0 percentage=12 elapsed=10.891s
INFO [08-20|11:59:47.739] Generating DAG in progress          epoch=0 percentage=13 elapsed=11.709s
INFO [08-20|11:59:48.537] Generating DAG in progress          epoch=0 percentage=14 elapsed=12.507s
INFO [08-20|11:59:49.371] Generating DAG in progress          epoch=0 percentage=15 elapsed=13.341s
INFO [08-20|11:59:50.212] Generating DAG in progress          epoch=0 percentage=16 elapsed=14.182s
INFO [08-20|11:59:50.518] Looking for peers                   peercount=0 tried=27 static=1
INFO [08-20|11:59:51.025] Generating DAG in progress          epoch=0 percentage=17 elapsed=14.995s
INFO [08-20|11:59:51.889] Generating DAG in progress          epoch=0 percentage=18 elapsed=15.859s
INFO [08-20|11:59:52.737] Generating DAG in progress          epoch=0 percentage=19 elapsed=16.708s
INFO [08-20|11:59:53.663] Generating DAG in progress          epoch=0 percentage=20 elapsed=17.634s
```

To stop that process, we can use…
miner.stop()

```
INFO [08-20|12:02:45.363] Generating DAG in progress          epoch=1 percentage=98 elapsed=1m30.767s
INFO [08-20|12:02:46.443] Generating DAG in progress          epoch=1 percentage=99 elapsed=1m31.847s
INFO [08-20|12:02:46.450] Generated ethash verification cache epoch=1 elapsed=1m31.854s
> miner.stop()
null
> eth INFO [08-20|12:02:56.380] Looking for peers              peercount=0 tried=15 static=1
```

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)

```
> eth.getBalance(eth.coinbase)
5e+23
>
```

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