## INDEX

|  |  |  |
| --- | --- | --- |
| **Sr.**  **No.** | **PracticalName** | **Page No.** |
| 1 | Write the following programs for Blockchain in Python:   1. A simple client class that generates the private and public keys by using the built-in Python RS Algorithm and test it. 2. A transaction class to send and receive money and test it. 3. Create multiple transactions and display them. 4. Create a blockchain, a genesis block and execute it. 5. Create a mining function and test it. 6. Add blocks to the miner and dump the blockchain. | 1 |
| 2 | Install and configure Go Ethereumand the Mistbrowser. Developand testa sample application. (Meta Mask & Remix) | 8 |
| 3 | Implement and demonstrate the use of the following in Solidity:   1. Variable,Operators,Loops,Decision Making,Strings,Arrays, Enums, Structs, Mappings, Conversions, Ether Units, Special Variables. 2. Functions, Function Modifiers, View functions, Pure Functions, Fall back Function, Function Overloading, Mathematical functions, Cryptographic functions**.** | 20 |
| 4 | Implement and demonstrate the use of the following in Solidity:   1. Withdrawal Pattern, Restricted Access. 2. Contracts, Inheritance, Constructors, Abstract Contracts, Interfaces. 3. Libraries,Assembly,Events,Error handling. | 45 |
| 5 | Write a program to demonstrate mining of Ether. | 64 |
| 6 | Demonstrate the running of the blockchain node. | 66 |
| 7 | Create your own blockchain and demonstrate its use. | 69 |

### PRACTICAL:1

1. A simple client class that generates the private and public keys by using the built-in Python RS Algorithm and test it.

importbinascii

importCrypto fromCrypto.PublicKeyimportRSA fromCrypto.SignatureimportPKCS1\_v1\_5

classClient:

def init (self): random=Crypto.Random.new().readself.\_private\_key= RSA.generate(1024, random) self.\_public\_key = self.\_private\_key.publickey()self.\_signer= PKCS1\_v1\_5.new(self.\_private\_key)

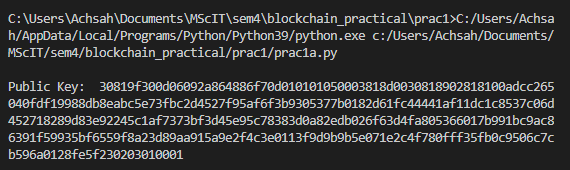
@property defidentity(self):return

binascii.hexlify(self.\_public\_key.exportKey(format="DER")).decode("ascii"

)

Dinesh=Client() print("\nPublicKey:",Dinesh.identity)

### Output:



1. **Atransactionclassto sendand receivemoneyandtestit.**

importbinasciiimport collectionsimportdatetim e

fromclientimportClientfrom Crypto.Hash import SHA

fromCrypto.SignatureimportPKCS1\_v1\_5

classTransaction:

def init (self,sender,recipient,value):self.sender= sender

self.recipient=recipientself.value=value self.time = datetime.datetime.now()

defto\_dict(self):

identity="Genesis"ifself.sender="Genesis"elseself.sender.identity returncollections.OrderedDict(

{

"sender": identity, "recipient": self.recipient,"value": self.value, "time": self.time,

}

)

defsign\_transaction(self): private\_key=self.sender.\_private\_keysigner= PKCS1\_v1\_5.new(private\_key) h=SHA.new(str(self.to\_dict()).encode("utf8")) returnbinascii.hexlify(signer.sign(h)).decode("ascii")

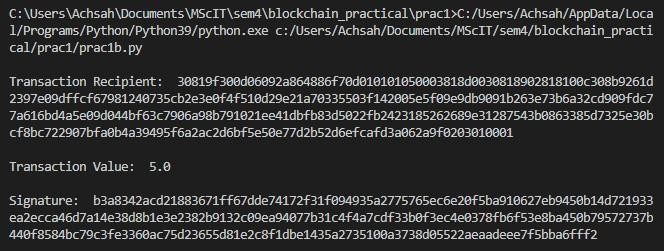
Dinesh = Client() Ramesh=Client()

t = Transaction(Dinesh, Ramesh.identity, 5.0) print("\nTransaction Recipient:\n",t.recipient)#print("\nTransactionSender:\n",t.sender)

print("\nTransactionValue:\n",t.value)

signature=t.sign\_transaction() print("\nSignature:\n",signature)

### Output:



###### Createmultipletransactionsanddisplaythem.

fromclientimportClient fromtransaction\_classimportTransaction

Dinesh = Client() Ramesh=Client()

t = Transaction(Dinesh, Ramesh.identity, 5.0) print("\nTransaction Recipient:\n",t.recipient)#print("\nTransactionSender:\n",t.sender) print("\nTransaction Value:\n", t.value)

signature=t.sign\_transaction() print("\nSignature:\n",signature)

Dinesh = Client() Ramesh = Client() Seema=Client()Vijay

=Client()

t1=Transaction(Dinesh,Ramesh.identity,15.0)t1.sign\_transaction() transactions = [t1]

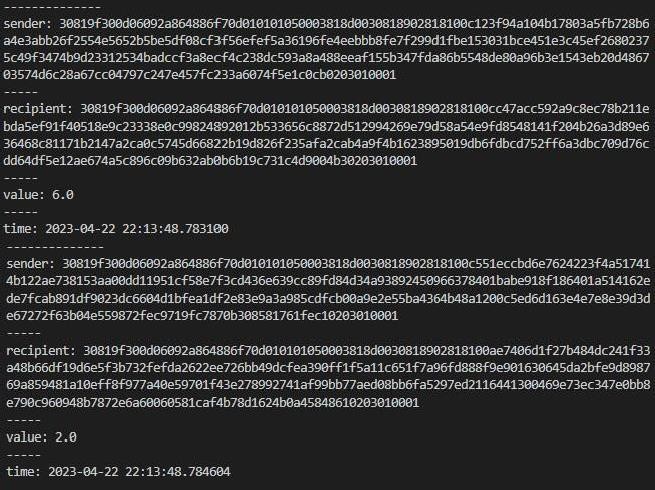
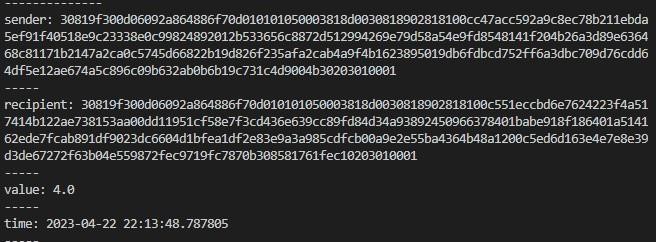
t2=Transaction(Dinesh,Seema.identity,6.0)

t2.sign\_transaction() transactions.append(t2) t3 = Transaction(Ramesh, Vijay.identity, 2.0) t3.sign\_transaction() transactions.append(t3) t4=Transaction(Seema,Ramesh.identity,4.0) t4.sign\_transaction() transactions.append(t4)

fortransactionintransactions:Transaction.display\_transaction(transaction)print("–

————————————–")

### Output:



###### Createablockchain,agenesisblockandexecuteit.

fromclientimportClient fromtransaction\_classimportTransaction

classBlock:

def init (self, client): self.verified\_transactions = [] self.previous\_block\_hash=""self.Nonce= "" self.client=client

defdump\_blockchain(blocks): print(f"\nNumberofblocksinthechain:{len(blocks)}")

fori,blockinenumerate(blocks):print(f"block#

{i}")

fortransactionin block.verified\_transactions:Transaction.display\_transaction(transaction)print("–

————————————–") print("=====================================")

Dinesh=Client() t0=Transaction("Genesis",Dinesh.identity(),500.0)

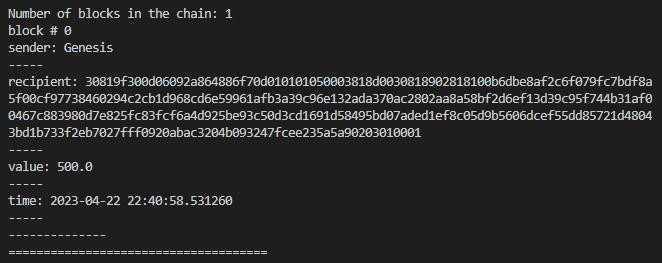
block0=Block(Dinesh)block0.previous\_block\_hash="" NONCE

= None

block0.verified\_transactions.append(t0)digest = hash(block0)

last\_block\_hash=digest

TPCoins = [block0] dump\_blockchain(TPCoins)



###### Createaminingfunctionandtestit.

importhashlib

defsha256(message): returnhashlib.sha256(message.encode("ascii")).hexdigest()

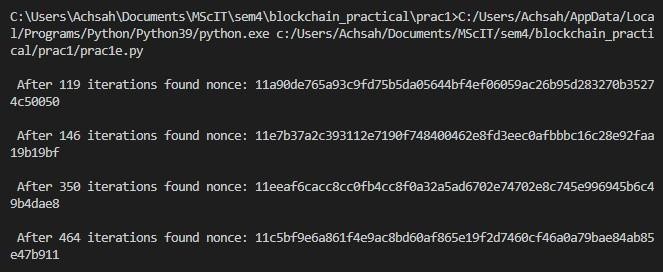
def mine(message, difficulty=1):assert difficulty÷1prefix="1"\*difficulty for i in range(1000):

digest=sha256(str(hash(message))+str(i))if digest.startswith(prefix):

print(f"after{str(i)}iterationsfoundnonce:{digest}")#returnprint(digest) mine("test

message", 2)

###### Output:



1. **Addblockstothemineranddumptheblockchain.**

importdatetime import hashlib

#Createaclasswithtwofunctions

classBlock:

def init (self,data,previous\_hash): self.timestamp=datetime.datetime.now(datetime.timezone.utc)self.data=data self.previous\_hash = previous\_hashself.hash =

self.calc\_hash()

defcalc\_hash(self):

sha=hashlib.sha256()

hash\_str=self.data.encode("utf-8")sha.update(hash\_str) returnsha.hexdigest()

#Instantiatetheclass blockchain=[Block("Firstblock","0")]

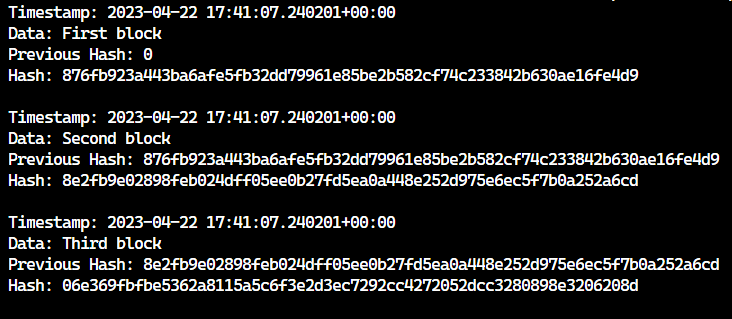
blockchain.append(Block("Second block", blockchain[0].hash))blockchain.append(Block("Third block", blockchain[1].hash))

# Dumping the blockchain forblockinblockchain:print(

f"Timestamp:{block.timestamp}\nData:{block.data}\nPreviousHash:

{block.previous\_hash}\nHash:{block.hash}\n")

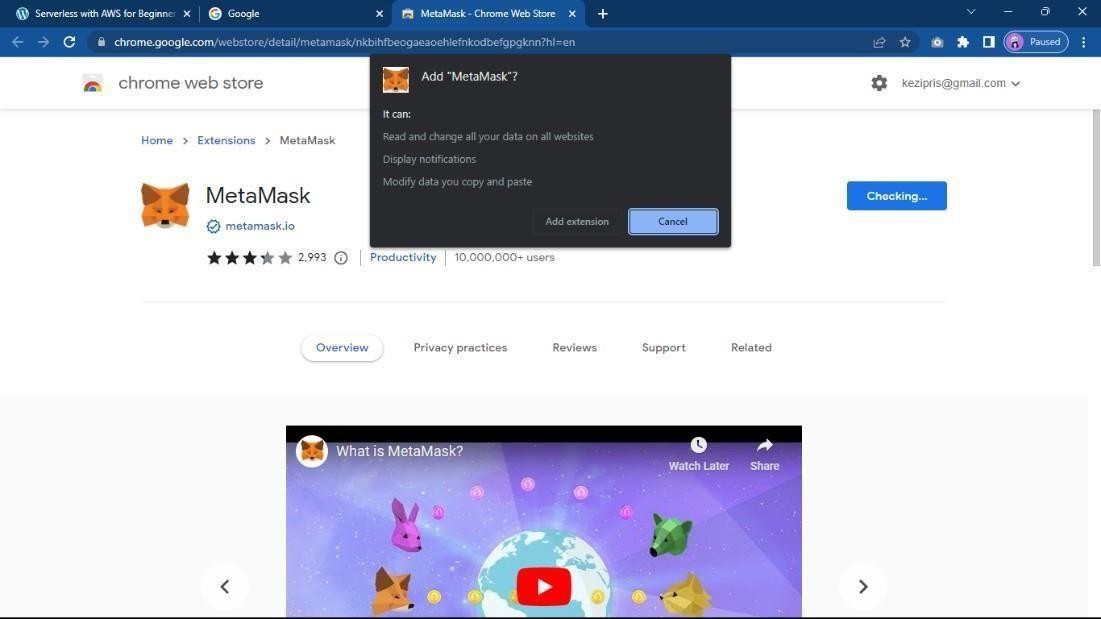
**Output:**

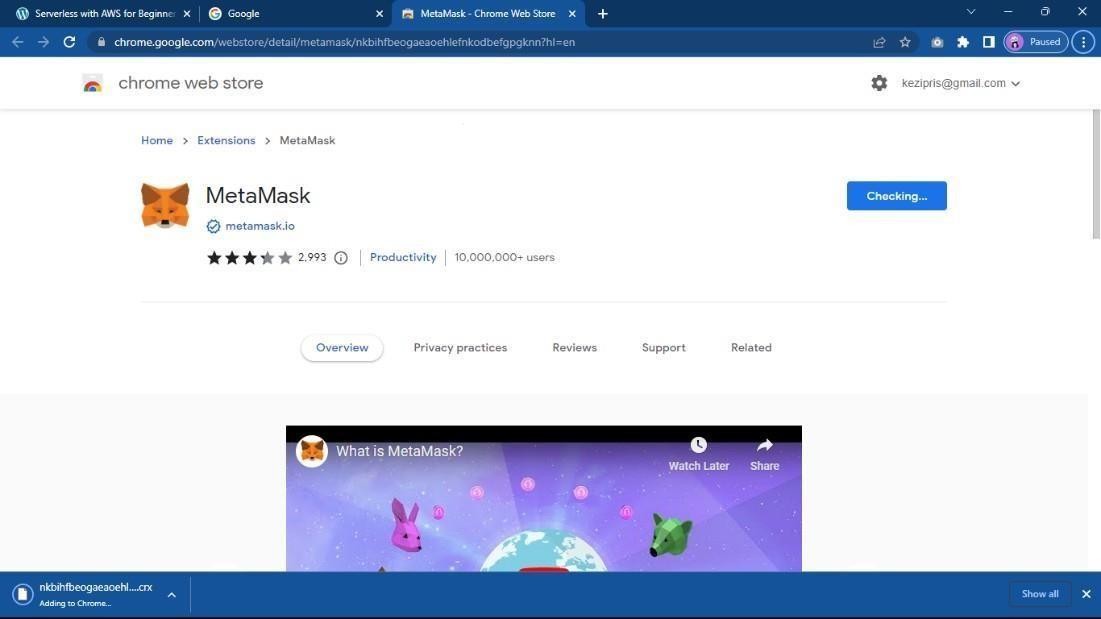


### PRACTICAL-2

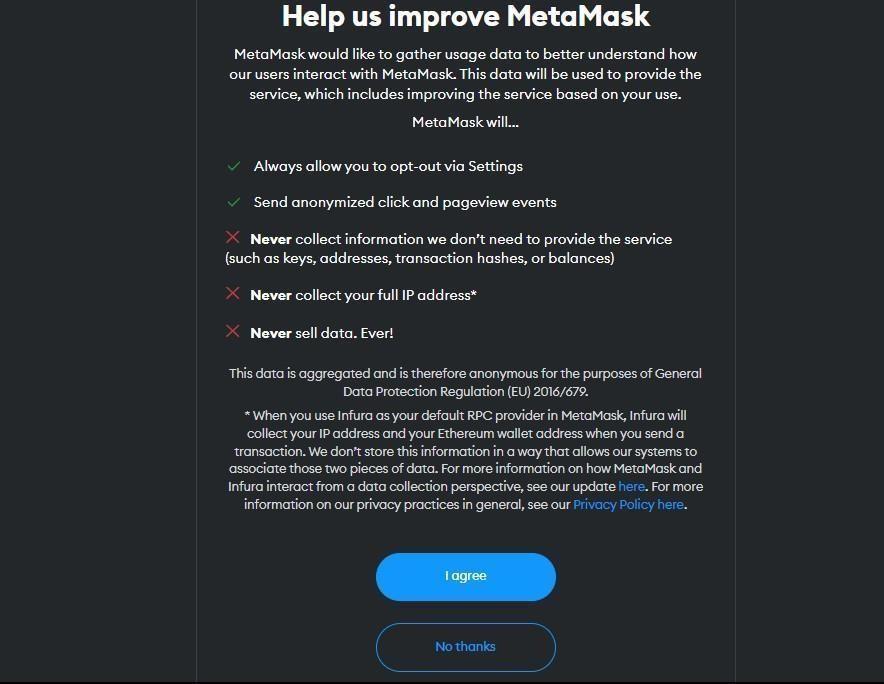
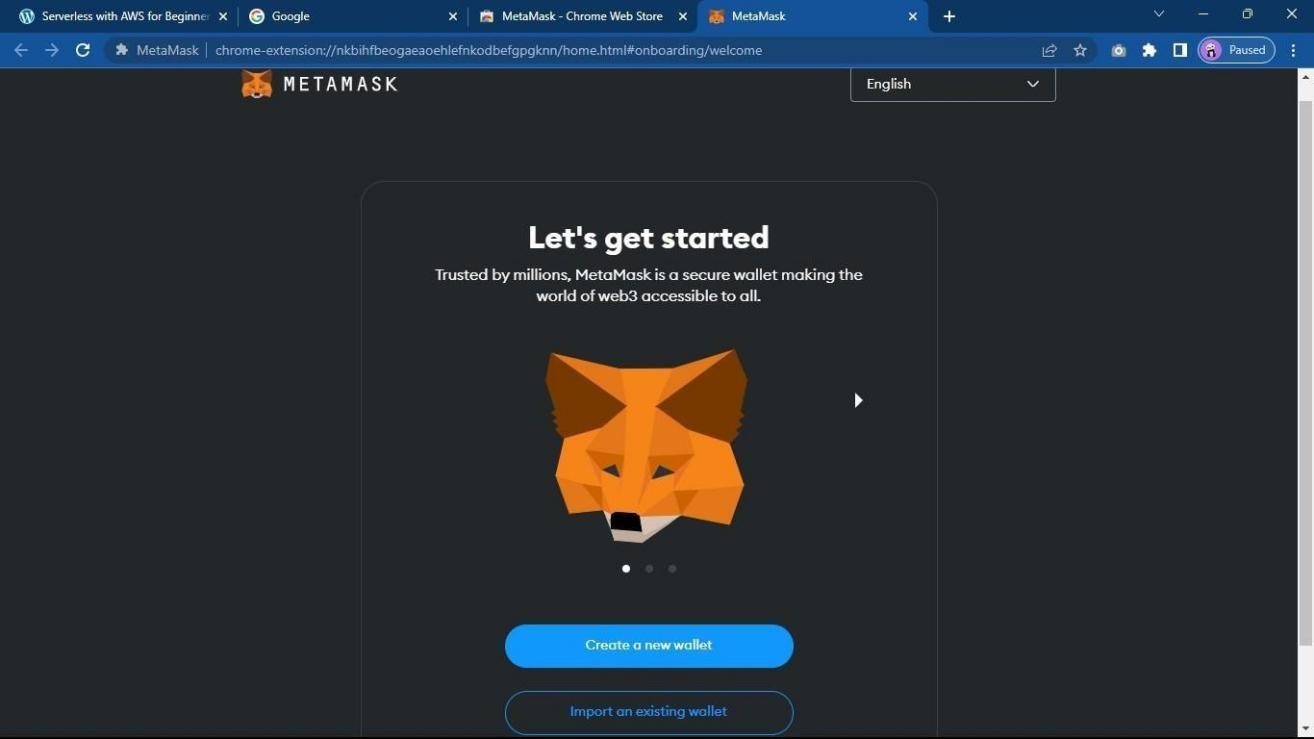
**Aim:Install and configure go Ethereum and themist browser.develop and test asample application (MetaMask& remix)**

**Step1->** InstallMetaMaskextensionforchromefromChromeWebStore

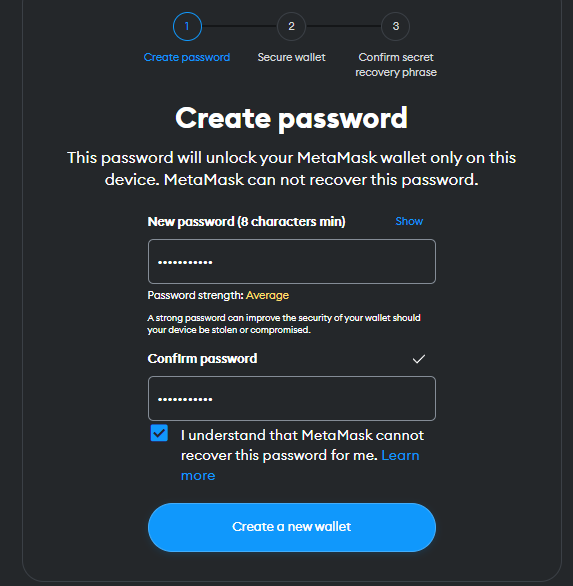


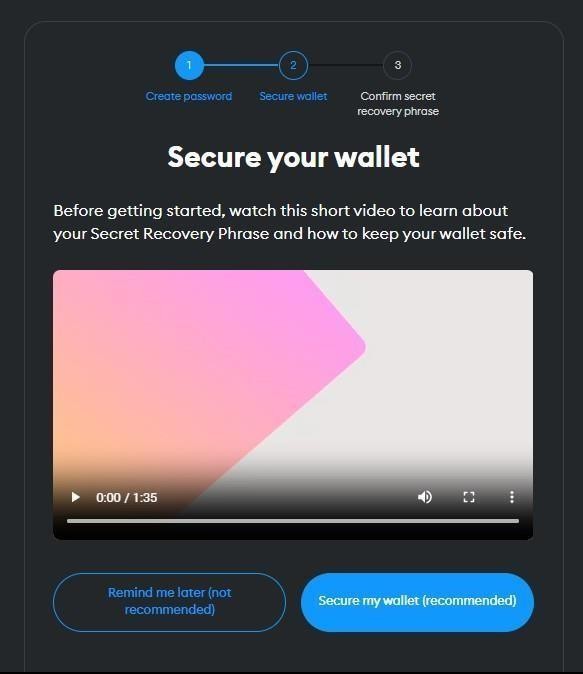


**Step 2->** Click on MetamaskExtension in Extensions. Below page will open in anewtab.Clickon Create a New Wallet. Click on I agree.

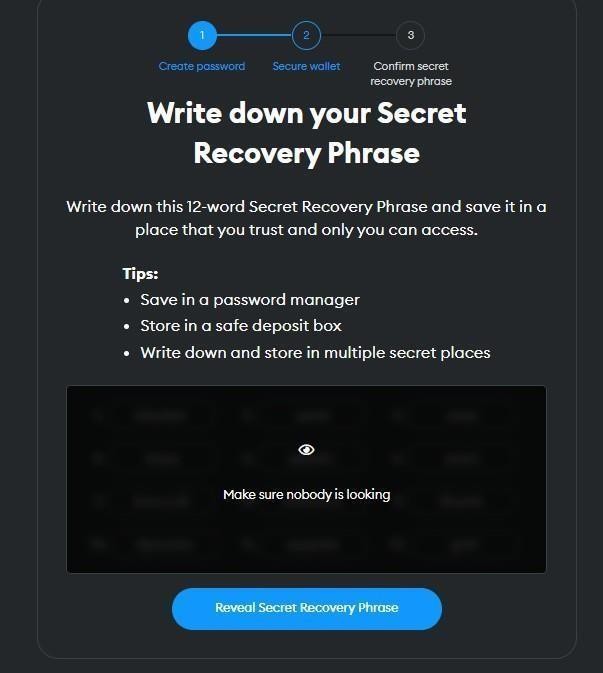


**Step 3->**Createapassword.Thispasswordcanbeusedonlyonthedeviceitwas created on. Create a Strong password and click on Create a new Wallet button

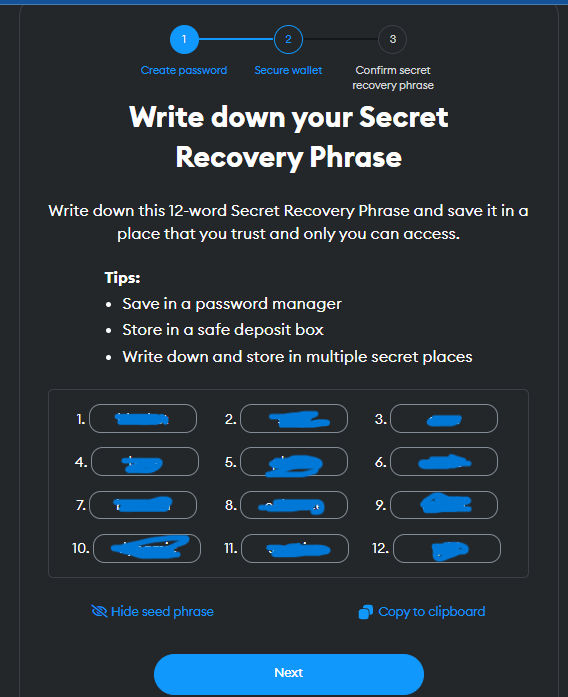




**Step4->** ClickonSecuremywalletbutton,followingwindowwillappear



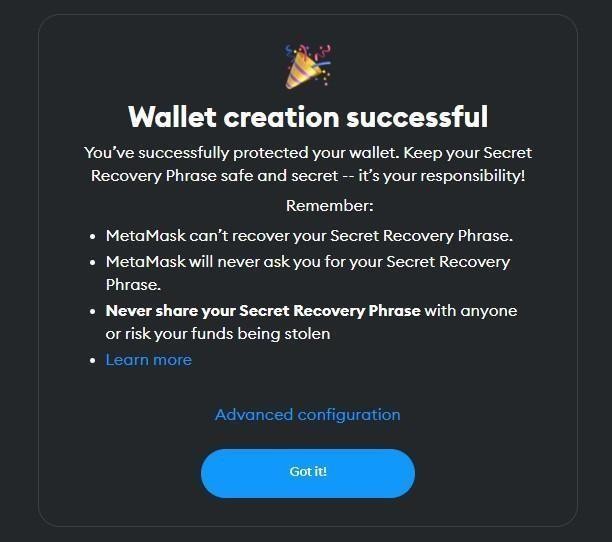
**Step5->** ClickonRevealSecretRecoveryPhrasebuttonandsavethewordsinthesamesequence



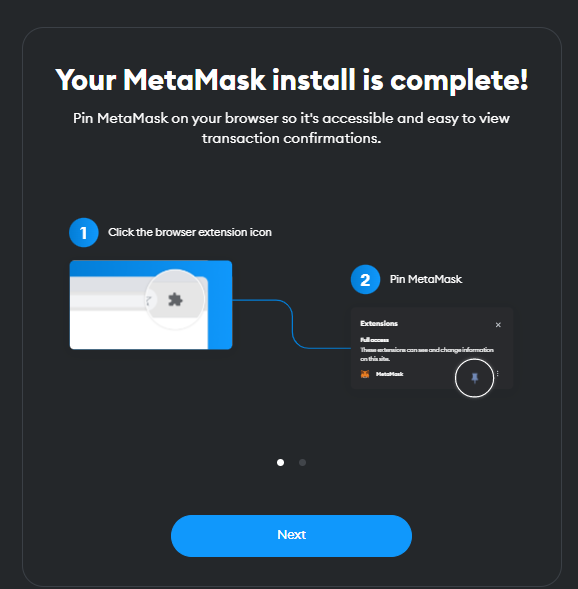
**Step6->** EntertherespectivewordsintheemptypositionsandclickConfirm.



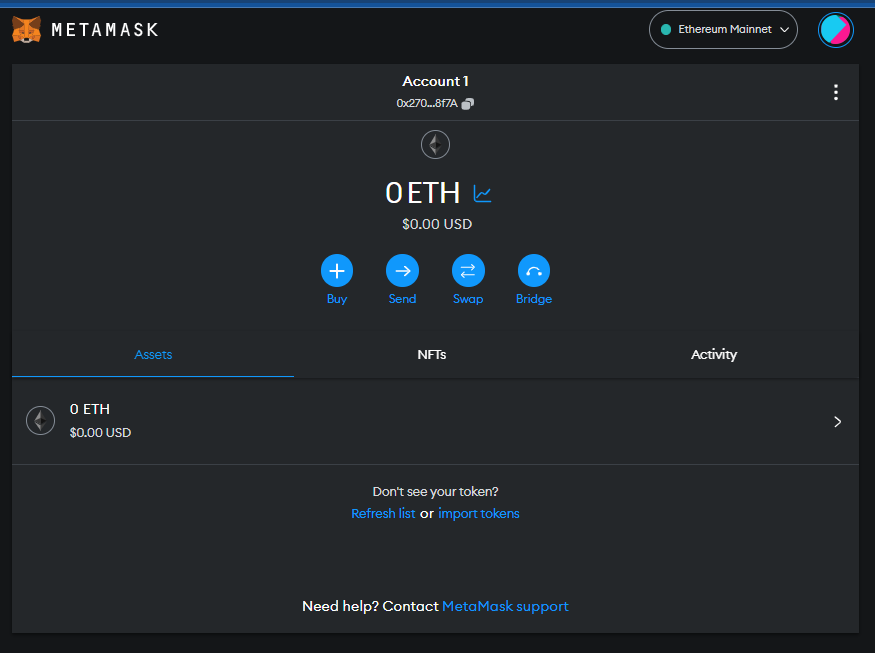
**Step7->** ClickGotit!



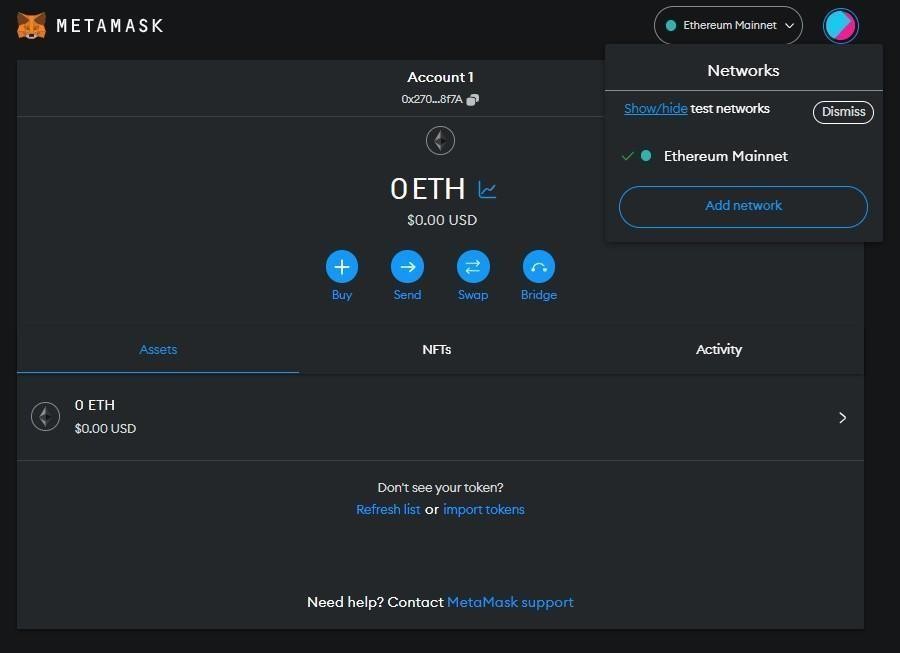
**Step8->** ClickonNext

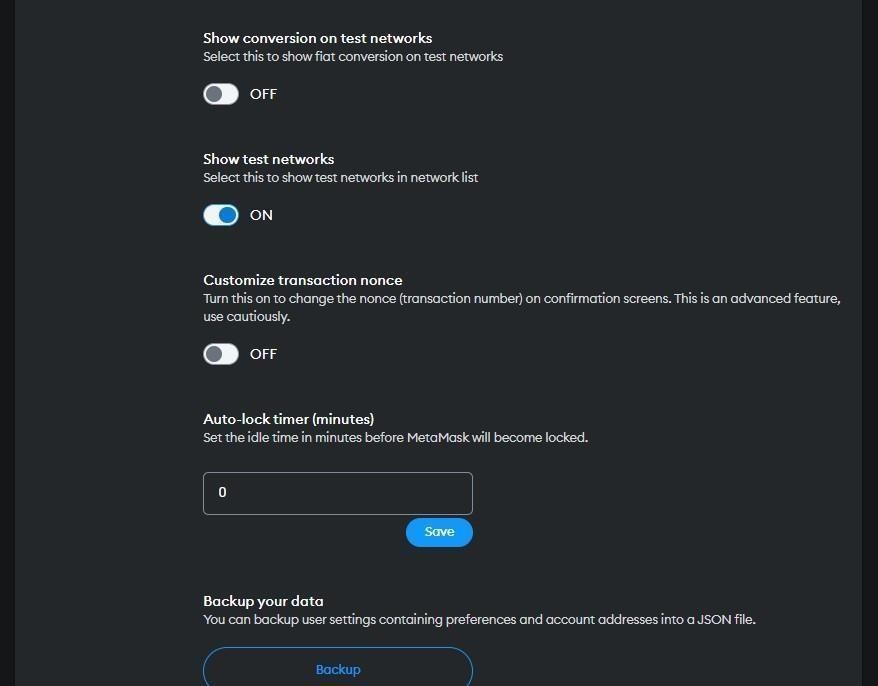


**Step9->** FollowingwillbetheDashboard

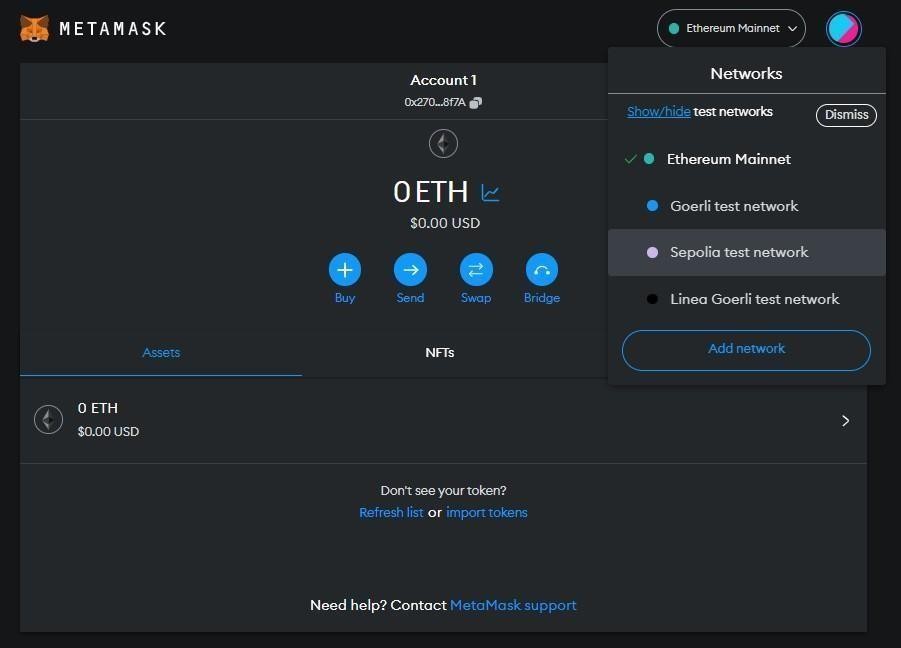


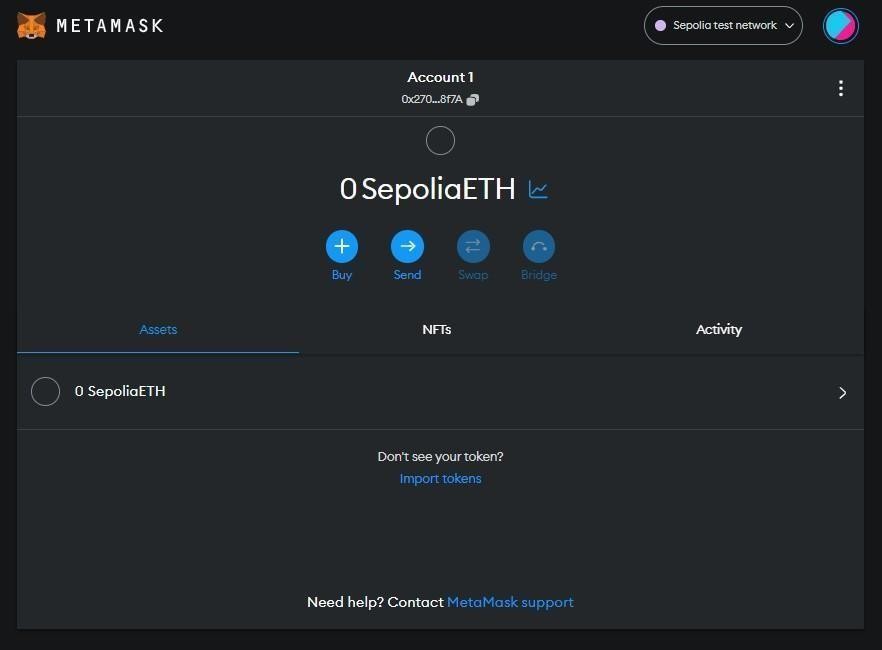
**Step10->**ClickonEthereumMainnet button.NextclickonShow/hidetestnetworks.



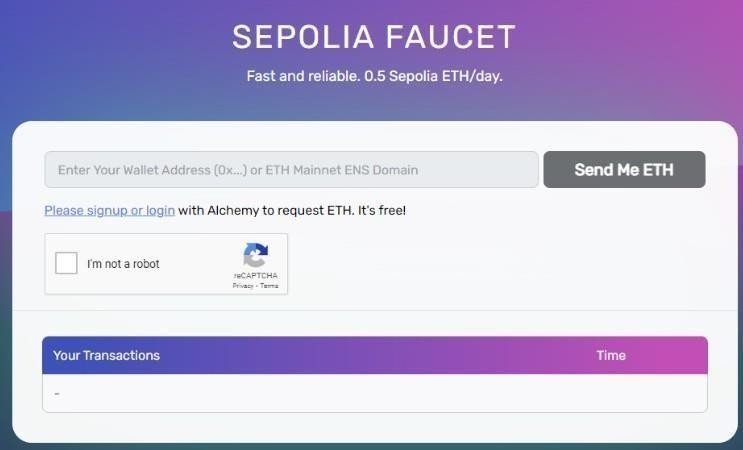


**Step 11->** Check if tesnets are shown byclickingonEtherumMainnetbutton.ClickonSepoliatest network.





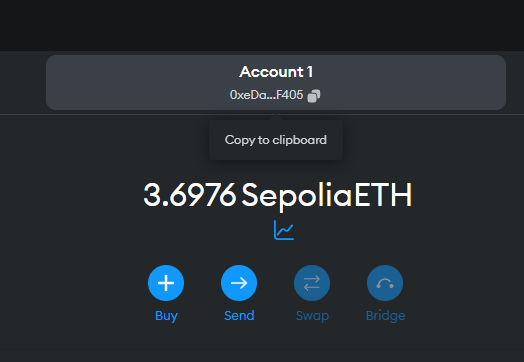
**Step12->**Goto<https://sepoliafaucet.com/>andClickonAlchemyLoginbutton.



**Step13->**Loginto agmailaccountin anotherbrowsertabandclickonSignin withGoogle



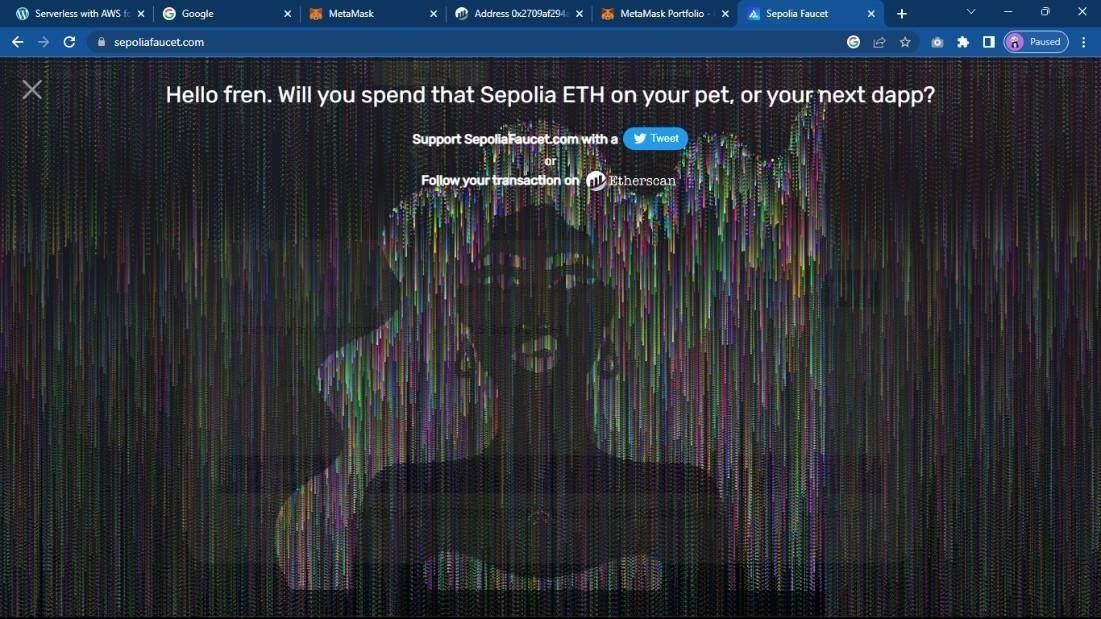
**Step14->**NowgotoMetaMaskandcopytheaccountaddress.



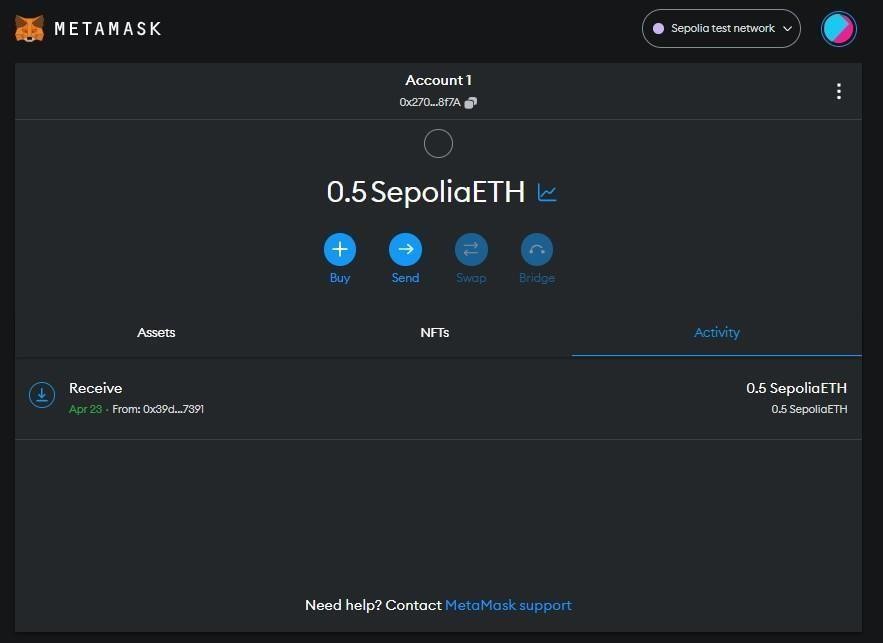
**Step15->**PastetheaddressandclickonSendMeETH.



**Step16->**YourETHtransferissuccesfull.Youshouldseeasimilaranimation.



**Step17->**CheckyourMetaMaskaccountforSepoliatestnetwork.0.5ETHwillbeadded.

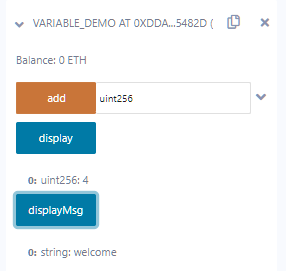


### PRACTICAL-3

**Aim:Implementanddemonstratetheuseofthefollowinginsolidity\**

* 1. TOEXECUTESOLIDITYSCRIPTSGOTO->[HTTPS://REMIX.ETHEREUM.ORG/](https://remix.ethereum.org/)
  2. OPEN CONTRACTS FOLDER AND STARTING WRITING SCRIPTS.THE SCRIPTS ARE COMPILED USING SOLIDITY COMPILER.
  3. THE FOLLOWING SCRIPTS WERE COMPILED USING 0.5.0+COMMIT.1D4F565A SOLIDITY COMPILER
  4. DEPLOYTHESCRIPTSTOEXECUTECODE

###### Variable, Operators, Loops, Decision Making, Strings, Arrays, Enums, Structs,Mappings, Conversions, Ether Units, Special Variables

* 1. **Variable**

pragmasolidity^0.5.0;

contractvariable\_demo{uint256s um=4;

//statevariableu int256x;addressa

;

strings="welcome";

functionadd(uint256)public{uint256y= 2;//local variable sum=sum+x+y:sum=sum+x+y;

}

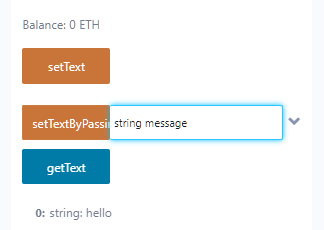
functiondisplay()publicviewreturns(uint256)

{returnsum;

}

Figure1-Displayingvariablevalue

###### Strings



pragmasolidity^0.5.0;

contractLearningStrings{stringt ext;

functiongetText()publicviewreturns(stringmemory){returntex t;

}

functionsetText()public{tex t="hello";

}

functionsetTextByPassing(stringmemorymessage)public{text=m essage;

}

}

Figure2-Beforesettingnewstringvalue

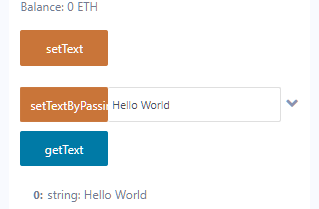
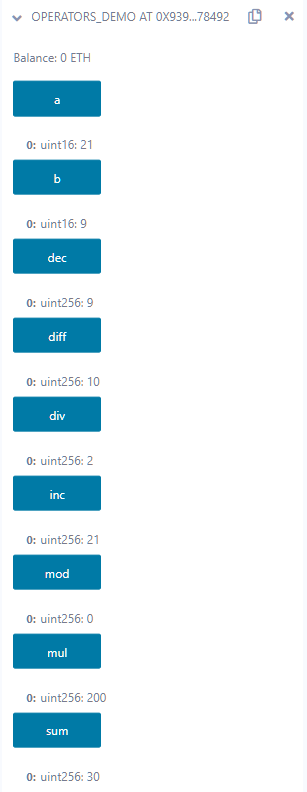


Figure3-Aftersettingstringvalue

###### Operators



pragmasolidity^0.5.0;

contractSolidityTest{uint 16publica=20;uint16pu blicb=10; uint256publicsum=a+b;uint256pub licdiff=a- b;uint256publicmul=a\*b;uint256p ublicdiv=a/b;uint256publicmod=a

%b;uint256publicdec= -- b;uint256publicinc= ++a;

}

Figure4-Alloperatorsofsoliditydisplayed

###### Array

pragmasolidity^0.5.0;contract arraydemo

{

//StaticArray uint[6]arr2=[10,20,30];

functiondispstaticarray()publicviewreturns(uint[6]memory)

{

returnarr2;

}

//DynamicArray uint x=5;

uint[] arr1; functionarrayDemo()public

{

while(x>0)

{

arr1.push(x);x=x- 1;

}

}

functiondispdynamicarray()publicviewreturns(uint[]memory)

{

returnarr1;

}

}

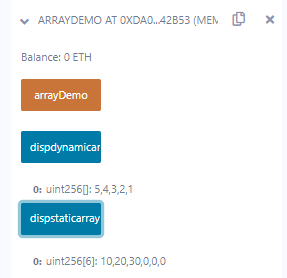


Figure5-Arraydisplayed

###### DecisionMaking

IfElse

pragmasolidity^0.5.0;contract ifelsedemo

{

uinti=10; functiondecision\_making()publicviewreturns(stringmemory)

{

if(i%2==0)

{

}

else

{

}

}

}

return"even";

return"Odd";



Figure6-Ifelseoutput

###### Loops

pragmasolidity^0.5.0;contract loopDemo

{

ForLoop

uint[] data;

functionforDemo()publicreturns(uint[]memory)

{

for(uinti=0;i<10;i++){ data.push(i);

}

returndata;

}

functiondisp()publicviewreturns(uint[]memory)

{

returndata;

}

}

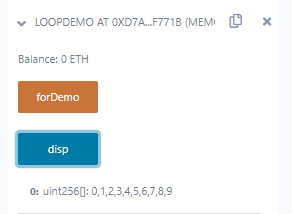


Figure7-Appendingvaluestoarrayusingforloop

pragmasolidity^0.5.0;contract whiledemo

WhileLoop

{

uint[]data;uintx=0;

functionwhileLoopDemo()public

{

while(x<5)

{

data.push(x); x=x+1;

}

}

functiondispwhileloop()publicviewreturns(uint[]memory)

{

returndata;

}

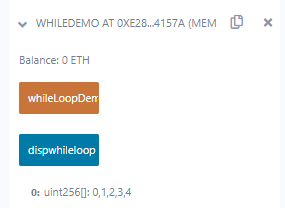
}

Figure8-Appendingvaluestoarrayusingwhileloop



Do

While

pragmasolidity^0.5.0;

//Creatingacontractcontra ctDoWhile{

//Declaringadynamicarrayuint256 []data;

//Declaringstatevariable uint8j=0;

//Definingfunctiontodemonstrate

//'Do-Whileloop' functionloop()publicreturns(uint256[]memory){do{

j++;

data.push(j);

}while(j<5);retur ndata;

}

functiondisplay()publicviewreturns(uint256[]memory){return data;

}

}

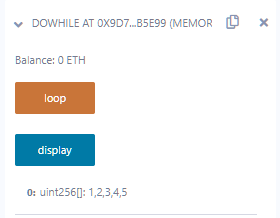
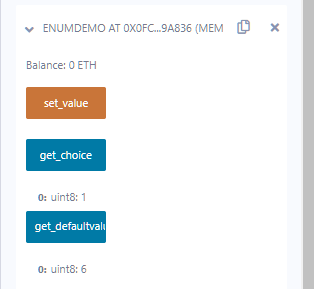


Figure9Appendingvaluestoarrayusingdowhileloop

###### Enums



pragmasolidity^0.5.0;

contractenumdemo{enumw eek\_days{

Monday,Tuesday

,Wednesday,Thu rsday,Friday,S aturday,Sunday

}

week\_days week;week\_dayschoice;

week\_daysconstantdefault\_value=week\_days.Sunday;

functionset\_value()public{choic e=week\_days.Tuesday;

}

functionget\_choice()publicviewreturns(week\_days){returncho ice;

}

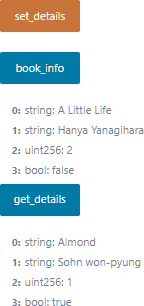
functionget\_defaultvalue()publicviewreturns(week\_days){returnd efault\_value;

}

}

Figure10-Accessingenumvalues

###### Structs



pragmasolidity^0.5.0;

contractstructdemo{struct Book{

stringname;string author;uint256id; boolavailability;

}

Bookbook2; Bookbook1=Book("ALittleLife","HanyaYanagihara",2,false);

functionset\_details()public{ book2=Book("Almond","Sohnwon-pyung",1,true);

}

functionbook\_info()p ublic

viewretur ns(

stringmemory,s tringmemory,ui nt256,

bool

)

{

return(book1.name,book1.author,book1.id,book1.availability);

}

functionget\_details()public viewreturn

s(

stringmemory,stringmemory,uint256,bool

)

{

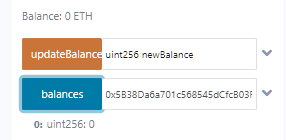
return(book2.name,book2.author,book2.id,book2.availability);

}

}

Figure 11-Structuredatatypeinsolidity

###### Mappings



pragmasolidity^0.5.0;

contractLedgerBalance{ mapping(address=>uint256)publicbalances;

functionupdateBalance(uint256newBalance)public{balances[ms g.sender] =newBalance;

}

}

contractUpdater{ functionupdateBalance()publicreturns(uint256)

{LedgerBalanceledgerBalance=newLedgerBalance();returnl edgerBalance.balances(address(this));

}

}

Figure12-Beforeupdatingbalance

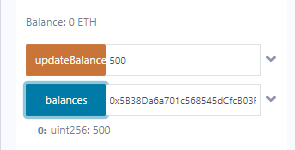


Figure13-Afterupdatingbalance

###### Conversions

//SPDX-License- Identifier:MITpragma solidity^0.8.0;

contractImplicitConversion{ functionadd()publicpurereturns(uint256){uint256a=10

;

uint256b=20;retur na+b;

}

}

contractExplicitConversion{ functionconvert()publicpurereturns(bytesmemory){stringmemo

rystr="Hello World"; bytesmemoryb=bytes(str);returnb

;

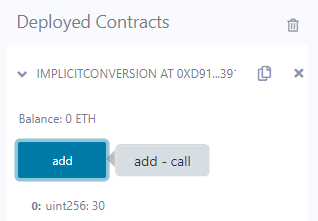
}

}

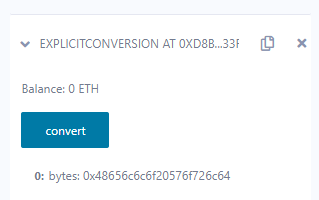
**Step1->** Deploybothcontracts



**Step2->** OpenImplicitConversionandclickonaddbuttontosumanddisplayvalue



**Step3->** OpenExplicitConversionandclickonconvertbutton



###### EtherUnits

//SPDX-License- Identifier:MITpragma solidity^0.8.0;

contractSolidityTest{ functionconvert\_Amount\_to\_Wei(uint256Amount)public

pure returns(uint256)

{

returnAmount\*1wei;

}

functionconvert\_Amount\_To\_Ether(uint256Amount)publi c

pure returns(uint256)

{

returnAmount\*1ether;

}

functionconvert\_Amount\_To\_Gwei(uint256Amount)public pure

returns(uint256)

{

returnAmount\*1gwei;

}

functionconvert\_seconds\_To\_mins(uint256\_seconds)pub lic

pure returns(uint256)

{

return\_seconds/60;

}

functionconvert\_seconds\_To\_Hours(uint256\_seconds)pu blic

pure returns(uint256)

{

return\_seconds/3600;

}

functionconvert\_Mins\_To\_Seconds(uint256\_mins)public pure

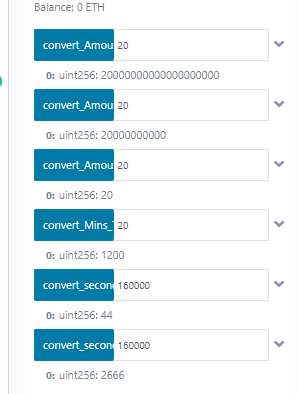
returns(uint256)

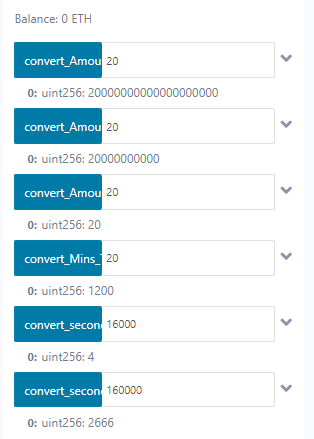
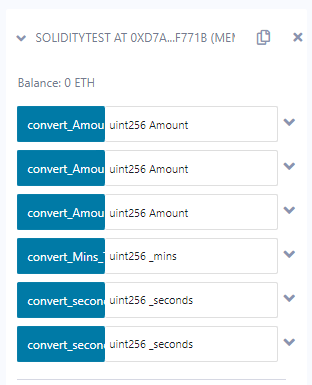
{

return\_mins\*60;

}

}



**Step1->** Providevaluestoeachfunctionandclickonthem

###### SpecialVariables

//SPDX-License- Identifier:MITpragma solidity^0.8.0;

contractSpecial\_Variables{mapping(addre ss=>uint256)rollNo;

functionsetRollNO(uint256\_myNumber)public{rollNo[msg.sende r] =\_myNumber;

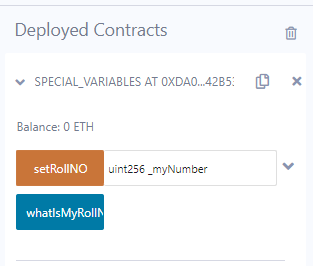
}

functionwhatIsMyRollNumber()publicviewreturns(uint256){returnr ollNo[msg.sender];

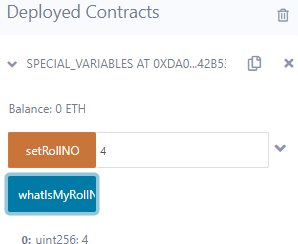
}

}

**Step1->** DeploycontractSpecialVariables

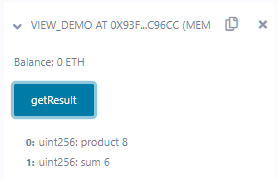


**Step 2->** Input a number for setRollNO function and click on it &whatIsMyRollNumber button



###### Functions,FunctionModifiers,Viewfunctions,PureFunctions,FallbackFunction,Function Overloading, Mathematical functions, Cryptographic functions

* 1. **ViewFunctions**



pragmasolidity^0.5.0;

contract

view\_demo{uint256 num1

=2;uint256 num2=4;

functiongetResult()publicviewreturns(uint256product,uint256sum)

{product=num1\*num2; sum=num1+num2;

}

Figure14-Viewfunctiondemo

###### PureFunctions

pragmasolidity^0.5.0;

contractpure\_demo{ functiongetResult()publicpurereturns(uint256product,uint256sum)

{uint256num1=2; uint256num2=4;product=n um1\*num2;sum=num1

+num2;

}

}

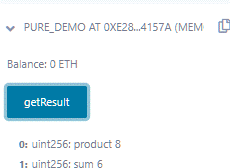


Figure15-Purefunctionoutput

###### MathematicalFunctions

pragmasolidity^0.5.0; contract Test{

functionCallAddMod()publicpurereturns(uint){return addmod(7,3,3);

}

functionCallMulMod()publicpurereturns(uint){return mulmod(7,3,3);

}

}

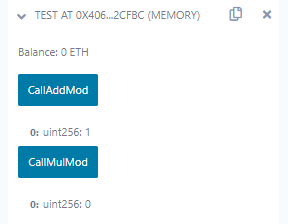


Figure16-Mathematicalfunctionsinsolidity

### CryptographicFunctions

pragmasolidity^0.5.0;contract Test{

functioncallKeccak256()publicpurereturns(bytes32result){return keccak256("BLOCKCHAIN");

}

functioncallsha256()publicpurereturns(bytes32result){return sha256("BLOCKCHAIN");

}

functioncallripemd()publicpurereturns(bytes20result){return ripemd160("BLOCKCHAIN");

}

}

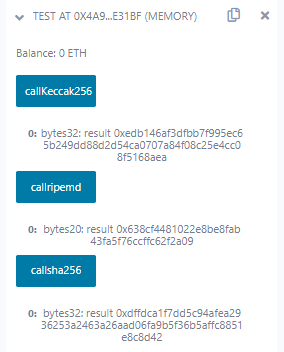


Figure17-Cryptographyalgorithmsinsolidity

###### Functions

// SPDX-License-Identifier: MITpragmasolidity>=0.4.22<0.9.0;

contractTest{

functionreturn\_example()pub lic

purereturn s(

uint256,uint2 56,uint256,st ringmemory

)

{

uint256num1=10;uint256n um2=16;

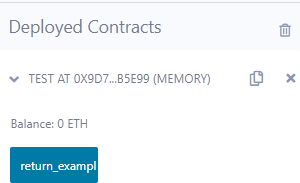
uint256sum=num1+num2;uint25 6prod=num1\*num2;uint256diff

=num2-num1; stringmemorymessage="Multiplereturnvalues";return(sum, prod,diff,message);

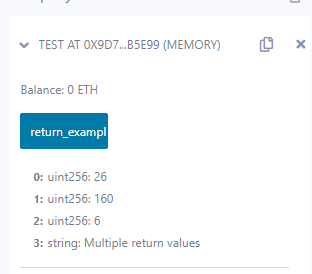
}

}

**Step1->** DeployTestContract



**Step2->** Clickonreturn\_examplebuttontodisplayallvalues



###### FallbackFunction

//SPDX-License- Identifier:MITpragma solidity^0.5.12;

contractA{

uint256n;

functionset(uint256value)external{n=va lue;

}

function()externalpayable{n=0

;

}

}

contractexample{ functioncallA(Aa)publicreturns(bool){

(boolsuccess,)=address(a).call(abi.encodeWithSignature("setter()"));require(success

);

addresspayablepayableA=address(uint160(address(a)));return (payableA.send(2ether));

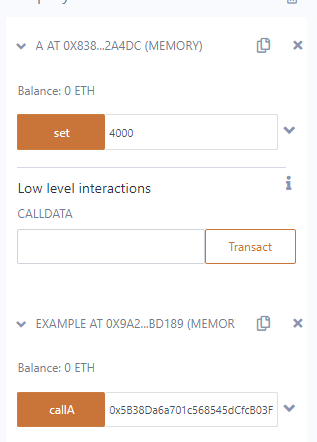
}

}

**Step1->** DeploybothA&examplecontracts



**Step2->** Providevaluestobothdeployedcontractsaccordingly(useanyaddress)



### FunctionOverloading

//SPDX-License- Identifier:MITpragma solidity^0.8.0;

contractOverloadingExample{ functionadd(uint256a,uint256b)publicpurereturns(uint256){returna+b;

}

functionadd(stringmemorya,stringmemoryb)public pure

returns(stringmemory)

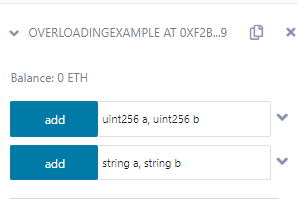
{

returnstring(abi.encodePacked(a,b));

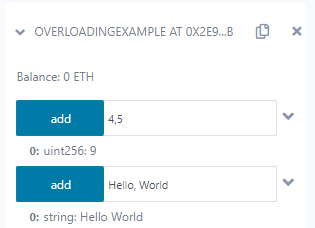
}

}

**Step1->** DeployOverloadingExamplecontract



**Step2->** Giveintegerandstringvaluestobothaddfunctionsasbelow



### Functionmodifiers

//SPDX-License- Identifier:MITpragma solidity^0.5.0;

contractExampleContract{ addresspublicowner=0x5B38Da6a701c568545dCfcB03FcB875f56beddC4;uint256pub liccounter;

modifieronlyowner(){ require(msg.sender==owner,"Onlythecontractownercancall");

\_;

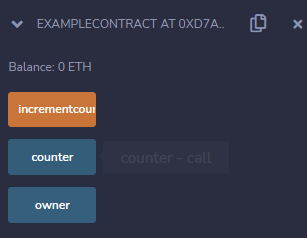
}

functionincrementcounter()publiconlyowner{counter++

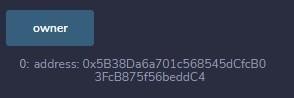
;

}

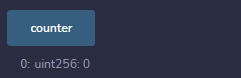
}



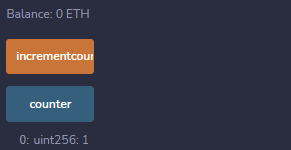
**Step1->** Clickonownerbutton



**Step2->** Clickoncounterbuttoninitiallyitis 0.



**Step 3->** Then click on increment counter button and again clickoncounterbutton,thecounterhas been increased



### PRACTICAL-4

**Aim:Implementanddemonstratetheuseofthefollowinginsolidity**

### WithdrawalPattern,RestrictedAccess

##### WithdrawalPattern

//SPDX-License- Identifier:MITpragma solidity0.8.18;

contractWithdrawalPattern{addre sspublicowner; uint256publiclockedbalance;uint256p ublicwithdrawablebalance;

constructor(){ owner=msg.sender;

}

modifieronlyowner(){ require(msg.sender==owner,"Onlytheownercancallthisfunction");

\_;

}

functiondeposit(uint256amount)public payable{require(amount>0,"Amountmustbegreaterthanzero");lo ckedbalance+=amount;

}

functionwithdraw(uint256amount)publicpayableonlyowner{require( amount<=withdrawablebalance,"Insufficient withdrawablebalance"

);

withdrawablebalance-

=amount;payable(msg.sender).transfer(amount);

}

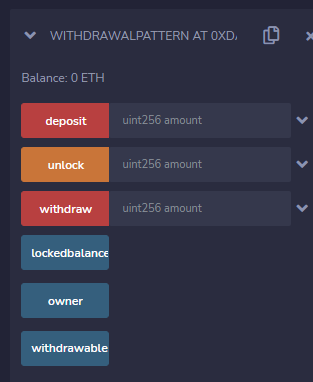
functionunlock(uint256amount)publiconlyowner{ require(amount<=lockedbalance,"Insufficientlockedbalance");lockedbal ance-=amount;

withdrawablebalance+=amount;

}

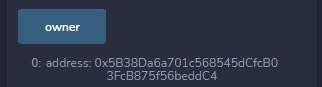
}

###### Output:

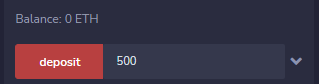


**Flowofexecution**

**Step1->** Clickonowner



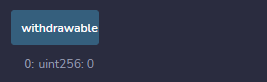
**Step2->** Enteranamountandclickondeposit



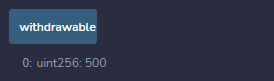
**Step3->** Clickonlockedbalancebuttontodisplaythelockedamountintheaccount



**Step4->** Clickonwithdrawablebalancebutton

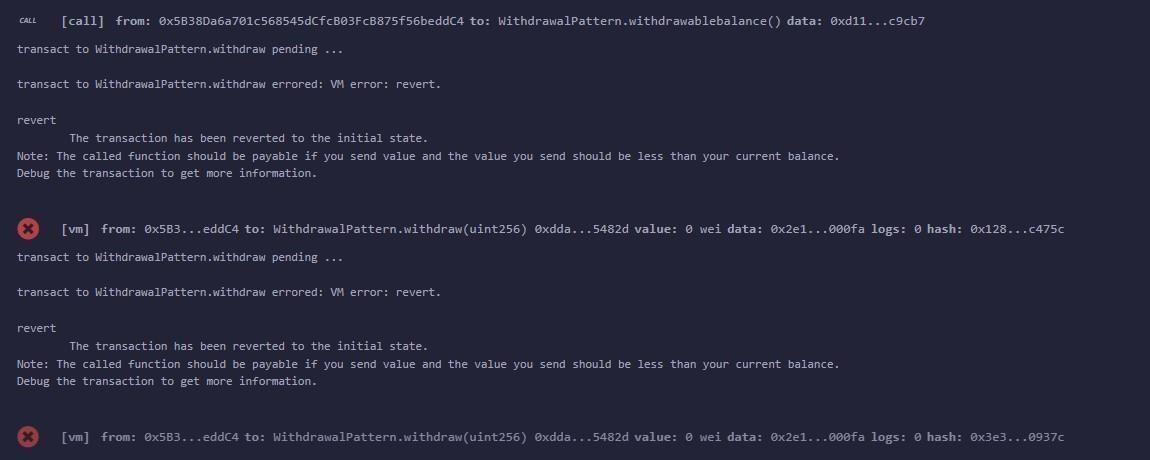


**Step 5->** Clickonunlockbuttonandenteranyamounttotransferamounttowithdrawable balance.Check locked balance and withdrawable balance.



**Step6->** EnteranyamountyouwanttowithdrawandClickthewithdrawbutton.

Youshouldgetanerrorandthetransactionshouldbereverted.



##### RestrictedAccess

//SPDX-License- Identifier:MITpragma solidity^0.8.18;

contractRestrictedAccess{ addresspublicowner=msg.sender; uint256publiccreationTime=block.timestamp;

modifieronlyBy(address\_account){ require(msg.sender==\_account,"Sendernotauthorized!");

\_;

}

modifieronlyAfter(uint256\_time){ require(block.timestamp>=\_time,"Functionwascalledtooearly!");

\_;

}

modifiercosts(uint256\_amount){ require(msg.value>=\_amount,"NotenoughEtherprovided!");

\_;

}

functionforceOwnerChange(address\_newOwner)public payablecosts(200

ether)

{

owner=\_newOwner;

}

functionchangeOwner(address\_owner)publiconlyBy(owner){owner=\_o wner;

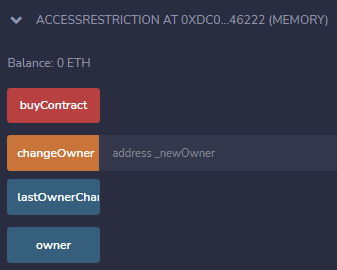
}

functiondisown()publiconlyBy(owner)onlyAfter(creationTime+3weeks){deleteowner;

}

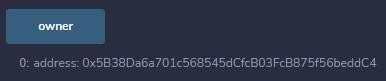
}

###### Output:

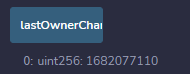


**Flowofexecution**

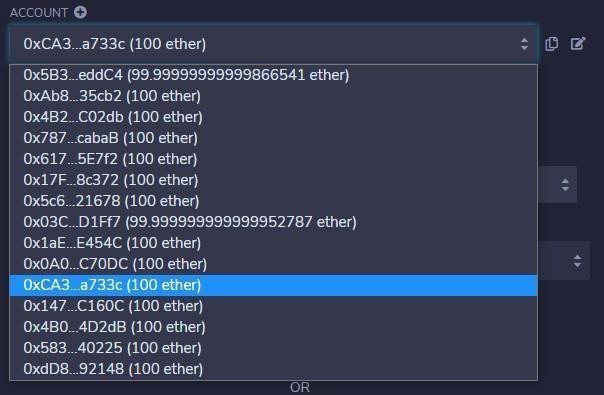
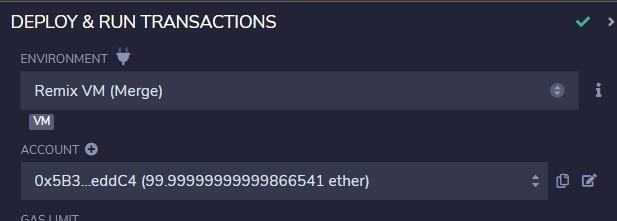
**Step1->** Clickonownertocreateanownerobject



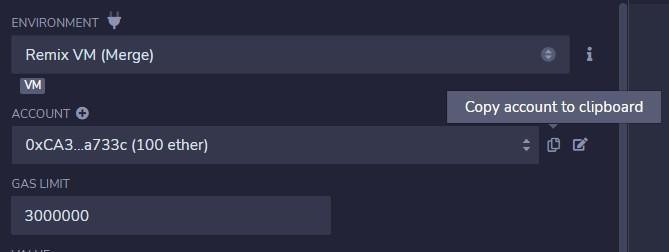
**Step2->** ClickonlastOwnerChangebutton



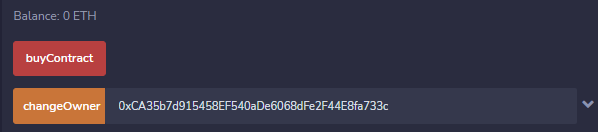
**Step3->** ChangetheaddressoftheaccountfromAccountdropdowninDeploytabofRemix IDE.



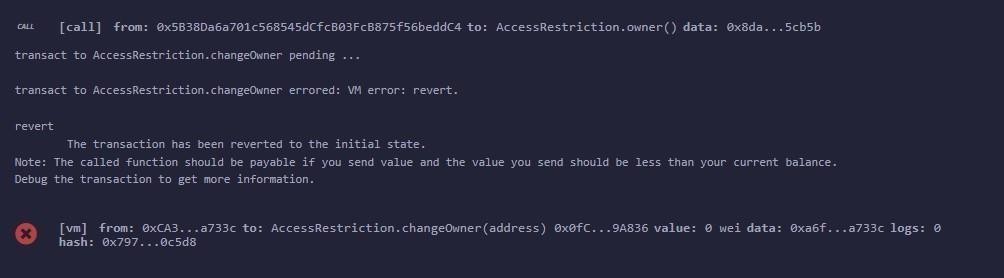
**Step4->** Copytheaddress



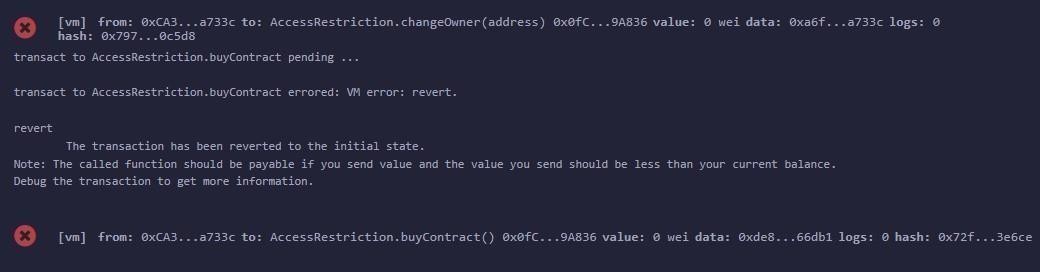
**Step5->** PastetheaddressinchangeOwnerinputandclickonchangeOwner.



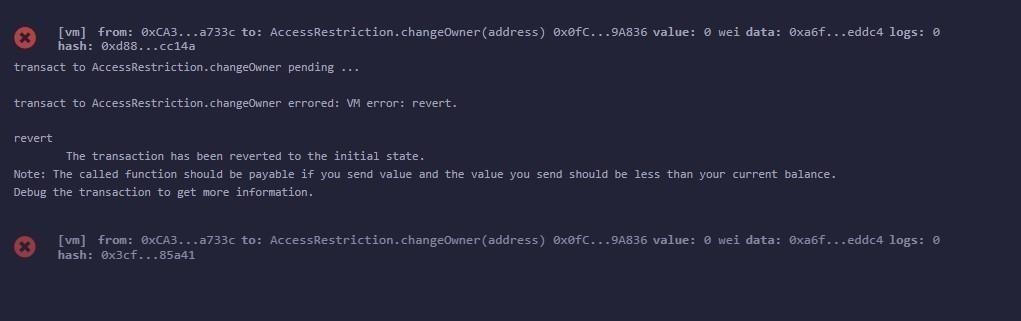
**Step6->** Youshouldgetanerrorasfollowing



**Step7->** Ifyouclickonbuycontractitshouldgiveanerrorasfollows

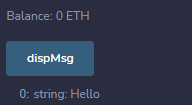


**Step 8->** Now,pastetheactualaddressoftheaccountinthechangeownerinputand click on changeowner



### Contracts,Inheritance,Constructors,AbstractContracts,Interfaces

* 1. Contract



Output

pragmasolidity^0.5.0;

contractContract\_demo{stringm essage="Hello";

functiondispMsg()publicviewreturns(stringmemory){returnmes sage;

}

}

##### Inheritance

}

functionshow\_value()publicviewreturns(uint256){returnc

c.getValue();

}

pragmasolidity>=0.4.22<0.6.0;

contractParent{uint256in ternalsum;

functionsetValue()external{uint256a= 10;

uint256b=20;sum=a

+b;

}

}

contractchildisParent{ functiongetValue()externalviewreturns(uint256){return

sum;

}

}

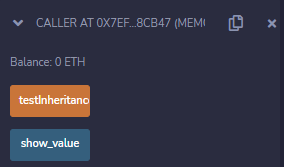
contractcaller{ childcc=newchild();

functiontestInheritance()publicreturns(uint256){ cc.setValue();

returncc.getValue();

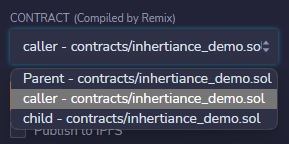
}

**Output:**

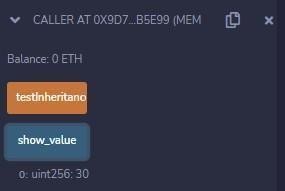


### Flowofexecution

**Step1->** SelectcallercontracttodeployinContractanddeploy



**Step2->** ClicktestInheritanceandthenclickonshow\_valuetoviewvalue



* 1. AbstractContracts

//SPDX-License- Identifier:MITpragma solidity^0.5.17;

contractCalculator{ functiongetResult()externalviewreturns(uint256);

}

contractTestisCalculator{constr uctor()public{}

functiongetResult()externalviewreturns(uint256){uint256a=1

;

uint256b=2; uint256result=a+b;retur nresult;

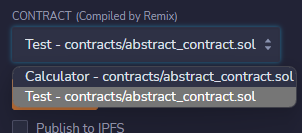
}

}

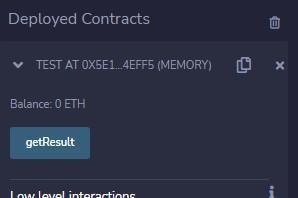
###### Outputs:

**Flowofexecution**

**Step1->** SelectTestcontractanddeploy

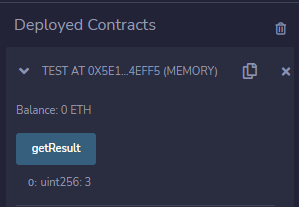


**Step2->** Thecontactwilldeployasbelow



Outputs

**Step3->** ClickongetResulttogetsumofa+b



1. Constructors



//SPDX-License- Identifier:MITpragma solidity^0.5.0;

// Creating a contractcontractconstructorEx ample{

stringstr;

constructor()public{ str="GeeksForGeeks";

}

functiongetValue()publicviewreturns(stringmemory){returnst r;

}

###### Flowofexecution

**Step1->** ClickongetValuetoprintstrin



##### Interfaces

pragmasolidity^0.5.0;

interfaceCalculator{ functiongetResult()externalviewreturns(uint);

}

contractTestisCalculator{constr uctor()public{}

functiongetResult()externalviewreturns(uint){uinta=1

;

uintb=2; uintresult=a+b;returnresu lt;

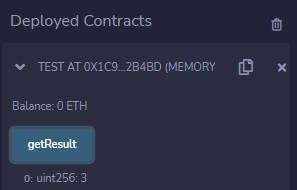
}

}

###### Outputs:

**Flowofexecution**

**Step1->** ClickongetResulttodisplaysum



1. **Libraries,Assembly,Events,Errorhandling.**

##### Libraries myLib.sol Code

}

//SPDX-License- Identifier:MITpragmasolidity>=0

.7.0<0.9.0;

librarymyMathLib{ functionsum(uint256a,uint256b)publicpurereturns(uint256){returna+b;

}

functionexponent(uint256a,uint256b)publicpurereturns(uint256){returna\*\*b

;

using}\_library.solCode

//SPDX-License- Identifier:MITpragmasolidity>=0

.7.0<0.9.0;

import"contracts/myLIB.sol";con tractUseLib{

functiongetsum(uint256x,uint256y)publicpurereturns(uint256){ returnmyMathLib.sum(x,y);

}

functiongetexponent(uint256x,uint256y)publicpurereturns(uint256){returnmyMathLi b.exponent(x,y);

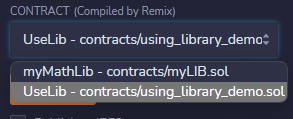
}

}

###### Outputs:

**Flowofexecution**

**Step1->** ChangecontracttoUseLibanddeploy.



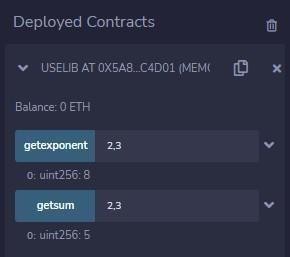
**Step2->** Thedeployedcontractshouldbesameasbelow



**Step3->** Inputvaluestobothgetexponentandgetsumfunctionsasbelow



**Step4->** Executebothfunctions.Youwillgetbelowoutpu



##### Assembly

//SPDX-License-Identifier:GPL- 3.0pragma solidity>=0.4.16<0.9.0;

contractInlineAssembly{

//Definingfunction functionadd(uint256a)publicviewreturns(uint256b){assembly{

letc:=add(a,16)msto re(0x80,c)

{

letd:=add(sload(c),12)b := d

}

b:=add(b,c)

}

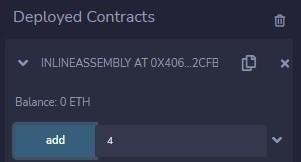
}

}



###### Flowofexecution

**Step1->** Inputanumberforaddfunction



**Step2->** Clickaddtooutputsum



##### Events

//SPDX-License- Identifier:MITpragma solidity^0.5.0;

// Creating a contractcontracteventExam ple{

//Declaringstatevariablesuint25 6publicvalue=0;

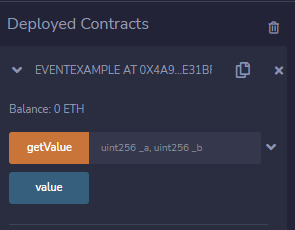
//Declaringanevent eventIncrement(addressowner);

//Definingafunctionforloggingevent functiongetValue(uint256\_a,uint256\_b)public{emitInc

rement(msg.sender); value=\_a+\_b;

}

###### Output



**Flowofexecution**

**Step1->** ProvidevaluestogetValuefunctionandclickonit.



**Step2->** Intheterminalcheckforlogs



##### ErrorHandling

contractErrorDemo{

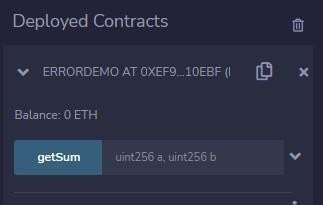
functiongetSum(uint256a,uint256b)publicpurereturns(uint256){uint256sum=a+b;

//require(sum<255,"Invalid");assert(sum<255); returnsum;

}

}

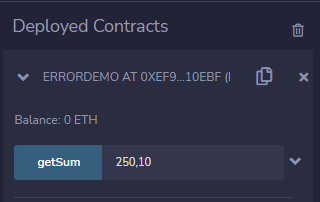
//SPDX-License-Identifier:MITpragma solidity^0.5.17;



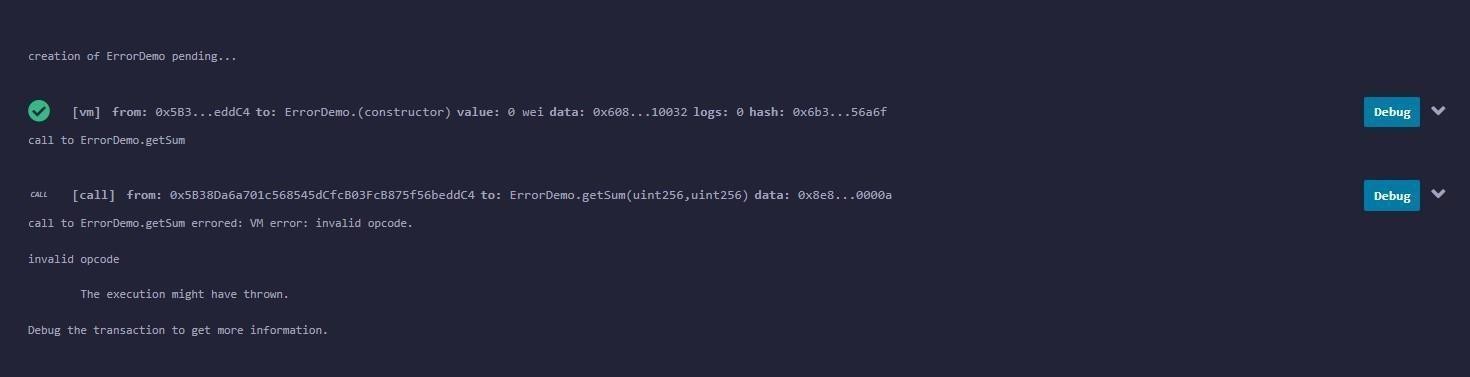
Output

###### Flowofexecution

**Step1->** ProvidesomevaluesandpressongetSum



**Step2->** Checkterminalpanel



### PRACTICAL-5

**Aim:Writeaprogramtodemonstrateminingofether**

constWeb3=require('web3');

constweb3=newWeb3(new Web3.providers.HttpProvider('http:127.0.0.1:7545'));ReplacewithyourGanacheHTTPprovider

asyncfunctionmine(){ constaccounts=awaitweb3.eth.getAccounts();constcoinbaseacc1= accounts[0];

constcoinbaseacc2=accounts[1]; console.log(`MiningetheronGanachewithcoinbaseaddress:

${coinbaseacc1}`);

while(true){try{

awaitweb3.eth.sendTransaction({from: coinbaseacc1, to:coinbaseacc2,value:

50,

});

console.log(`Minedanewblock!`);

}catch(err){console.error(err);

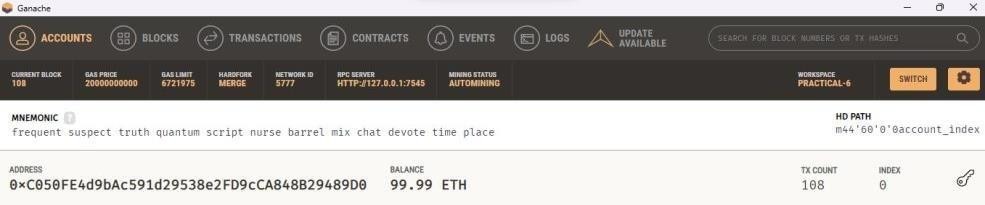
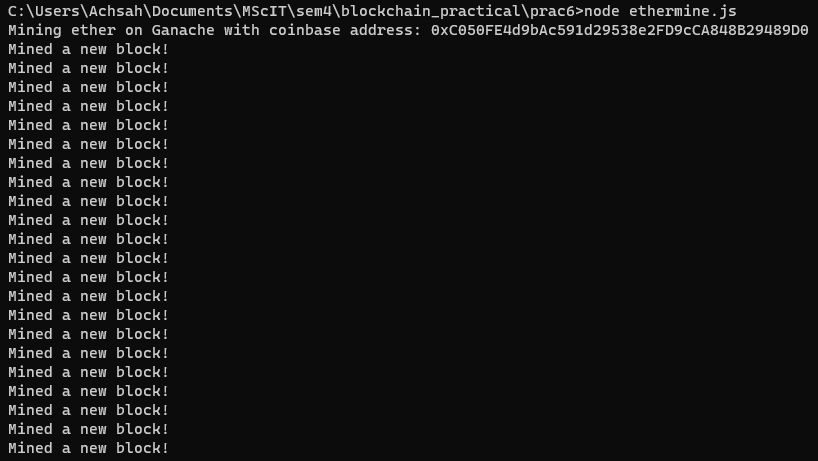
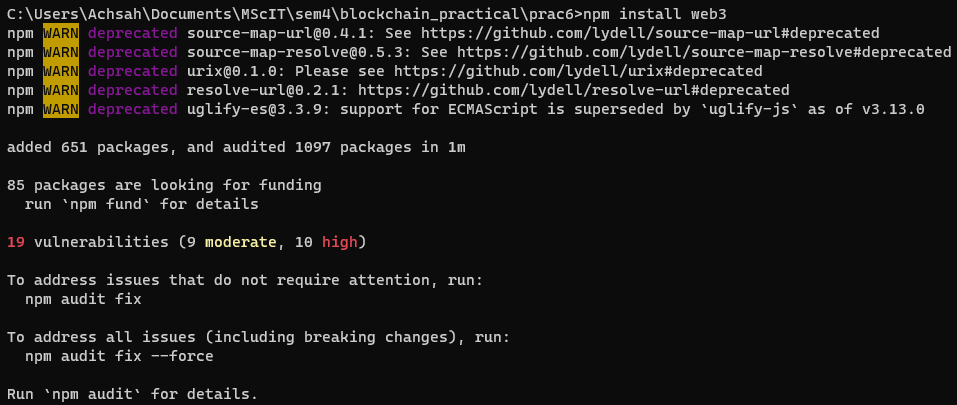
}

}

}

mine();

**Output:**



### PRACTICAL-6

**Aim:Demonstratetherunningoftheblockchainnode**

**Step1->**Create a folder named ethermine and a JSON file named genesis.jsonandwritethefollowinglines in it.

{

"config":{

"chainId":3792, "homesteadBlock":0, "eip150Block":0, "eip155Block":0, "eip158Block":0

},

"difficulty":"2000", "gasLimit":"2100000","alloc":

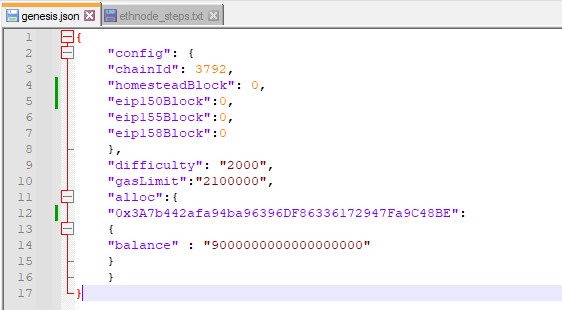
{

"0×0b6C4c81f58B8d692A7B46AD1e16a1147c25299F": {"balance": "9000000000000000000"

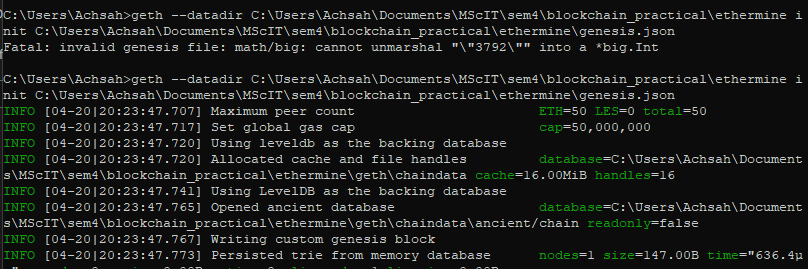
}

}

}



###### Step 2->Run command geth account new – datadirC:\Users\Achsah\Documents\MScIT\sem4\blockchain\_practical\etherminetestnet- blockchain

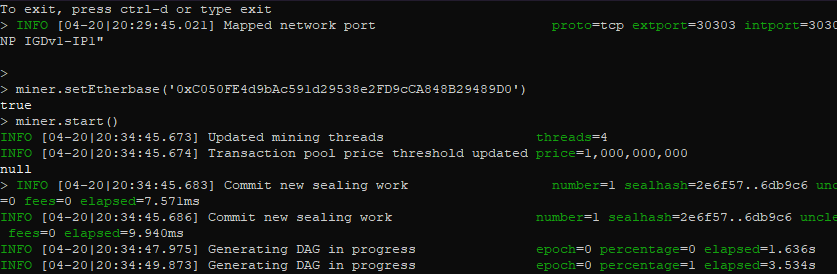
**Step 3->** Run command **geth account new -- datadirC:\Users\Achsah\Documents\MScIT\sem4\blockchain\_practical\eth ermine**

###### Step4->Runcommandgeth--identity"localB"--http--http.port"8280"--http.corsdomain"\*"-- http.api"db,eth,net,web3" –datadir "C:\Users\Achsah\Documents\MScIT\sem4\blockchain\_practical\ethermine" --port "30303" - nodiscover --networkid 5777 console. This command willenablegeth console.

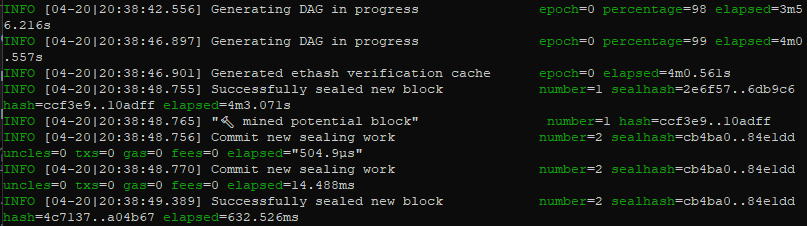
**PillaiHOCCollegeofArts,ScienceandCommerce Page| 67**

**Step5->** Run the command

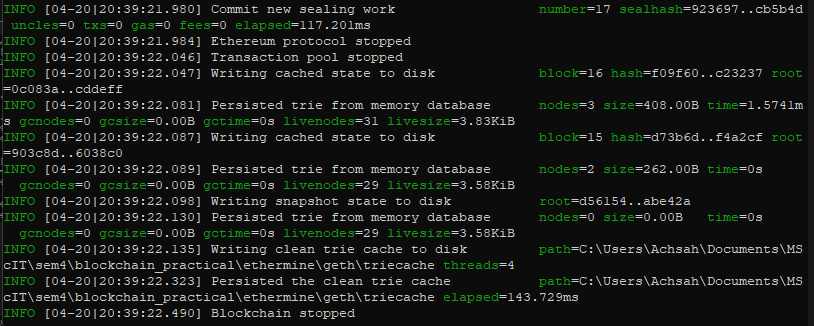
**miner.setEtherbase('0xC050FE4d9bAc591d29538e2FD9cCA848B29489D0’)**inthegethconsole **Step6->** Runthecommand**miner.start()**tostartmining



**Step7->** Belowscreenshotsaretheminingprocessesrunningonyourlocalmachine.



**Step8->** Tostoptheminingpress**Ctrl+D**



### PRACTICAL-7

**Aim:Createyourownblockchainanddemonstrateitsuse**

Createajavascriptfolderwiththefollowingcodeinanyfolderofyourchoice.

###### JavaScript Code

constSHA256=require("crypto-js/sha256");classBlock{ constructor(index,timestamp,data,previousHash=""){this.index=index;

this.timestamp = timestamp; this.data = data; this.previousHash=previousHash;this.hash= this.calculateHash();

}

calculateHash(){return SHA256(

this.index+this.previousHash+ this.timestamp + JSON.stringify(this.data)

).toString();

}

}

classBlockchain{ constructor() {

this.chain=[this.createGenesisBlock()];

}

createGenesisBlock(){ returnnewBlock(0,"21/04/2023","GenesisBlock","0");

}

getLatestBlock(){ returnthis.chain[this.chain.length-1];

}

addBlock(newBlock){ newBlock.previousHash=this.getLatestBlock().hash;

newBlock.hash=newBlock.calculateHash();this.chain.push(newBlock);

}

isChainValid(){ for(leti=1;i<this.chain.length;i+){constcurrentBlock= this.chain[i];

constpreviousBlock=this.chain[i-1];

if(currentBlock.hash currentBlock.calculateHash()){returnfalse;

}

if(currentBlock.previousHash previousBlock.hash){return false;

}

}

returntrue;

}

}

BlockchainImplementation

letmyCoin=newBlockchain(); myCoin.addBlock(newBlock(1,"22/04/2023",{amount:4}));myCoin.addBlock(newBlock(2,"22/04/2023",

{amount:8})); console.log('Isblockchainvalid?'+myCoin.isChainValid());console.log(JSON.stringify(myCoin,null, 4));

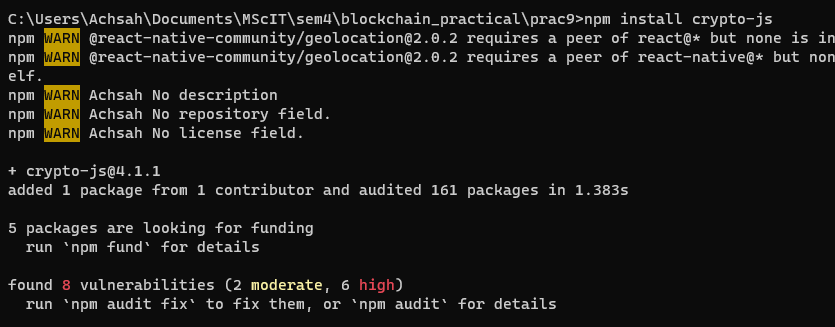
###### Flowofexecution

Output

**Step1->** Makesureyouhaveinstallednodejsinyoursystem



**Step2->**Weneed**crypto–js**nodemoduletomakeourownblockchain.Soinstallitas following



**Step3->** Runtheabovecodeincommandlineusingcomman