**CRYPOGRAPHY**

**Practical No:-1**

1. **Write a program to implement symmetric encryption using Ceaser Cipher algorithm.**

def encrypt(text,s):

result=""

for i in range(len(text)):

char = text[i]

if(char==""):

result +="";

else:

if(char.isupper()):

result += chr((ord(char) + s - 65) % 26 + 65)

else:

result += chr((ord(char) + s - 97) % 26 + 97)

return result

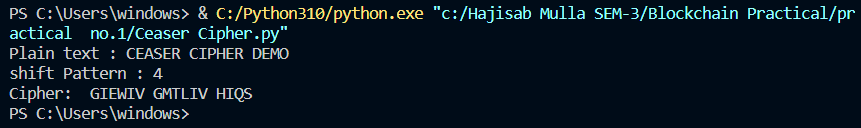
text = "CEASER CIPHER DEMO"

s=4

print("Plain Text :" +text)

print("Shift Pattern :" +str(s))

print("Cipher :" +encrypt(text ,s))



1. **Write a program to implement asymmetric encryption using RSA algorithm. Generate both the keys public key and private key and store it in file. Also encrypt and decrypt the message using keys.**

from Crypto.Cipher import PKCS1\_OAEP

from Crypto.PublicKey import RSA

from binascii import hexlify

message=b"Public and Private keys encryption"

private\_key =RSA.generate(1024)

public\_key=private\_key.publickey()

print(type(private\_key),type(public\_key))

private\_pem=private\_key.export\_key().decode()

public\_pem=public\_key.export\_key().decode()

print(type(private\_pem),type(public\_pem))

with open('private\_pem.pem','w') as pr:

pr.write(private\_pem)

with open('public\_pem.pem','w') as pu:

pu.write(public\_pem)

pr\_key =RSA.import\_key(open('private\_pem.pem','r').read())

pu\_key =RSA.import\_key(open('public\_pem.pem','r').read())

print(type(pr\_key),type(pu\_key))

cipher =PKCS1\_OAEP.new(key=pu\_key)

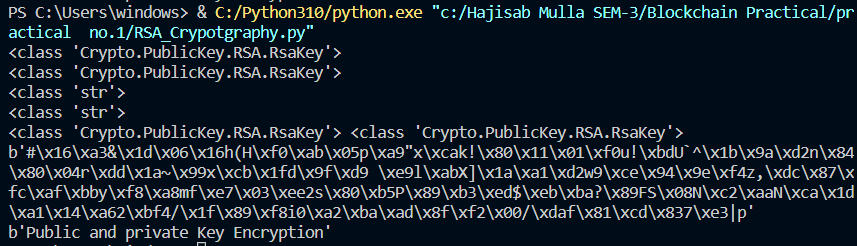
cipher\_text=cipher.encrypt(message)

print(cipher\_text)

decrypt=PKCS1\_OAEP.new(key=pr\_key)

decrypted\_message=decrypt.decrypt(cipher\_text)

print(decrypted\_message)



**3. Write a program to demonstrate the use of Hash Functions (SHA-256).**

import hashlib

string="Hello Good Morning"

encoded=string.encode()

result=hashlib.sha256(encoded)

print("String:",end="")

print(string)

print("Hash Value:",end="")

print(result)

print("Hexadecima equivalent",result.hexdigest())

print("Digest Size:",end="")

print(result.digest\_size)

print("Block Size:",end="")

print(result.block\_size)



**4.Write a program to demonstrate Merkle Tree.**

var merkle=require('merkle')

var str = 'Hajisab, Sanket, Omkar, Ketan, Saurabh, Sameer';

var arr=str.split(',');

console.log("Input:\t\t",arr);

var tree=merkle('sha1').sync(arr);

console.log("Root hash:\t",tree.root());

console.log("Tree depth:\t",tree.depth());

console.log("Tree levels:\t",tree.levels());

console.log("Tree nodes:\t",tree.nodes());

var i;

for(i=0;i< tree.levels();i++)

{

console.log("\nLevels ",i);

console.log(tree.level(i));

}

****