**Cryptography and 19115045**

**Network Security Lab Kunal Sachdeva**

**LAB-3 6th Sem CSE**

**1. Write a program to implement the concept of AES algorithm.**

**Code:**

*import java.nio.charset.StandardCharsets;*

*import java.security.spec.KeySpec;*

*import java.util.Base64;*

*import javax.crypto.Cipher;*

*import javax.crypto.SecretKey;*

*import javax.crypto.SecretKeyFactory;*

*import javax.crypto.spec.IvParameterSpec;*

*import javax.crypto.spec.PBEKeySpec;*

*import javax.crypto.spec.SecretKeySpec;*

*class AES {*

*private static final String SECRET\_KEY*

*= "my\_super\_secret\_key\_ho\_ho\_ho";*

*private static final String SALT = "ssshhhhhhhhhhh!!!!";*

*public static String encrypt(String strToEncrypt)*

*{*

*try {*

*byte[] iv = { 0, 0, 0, 0, 0, 0, 0, 0,*

*0, 0, 0, 0, 0, 0, 0, 0 };*

*IvParameterSpec ivspec*

*= new IvParameterSpec(iv);*

*SecretKeyFactory factory*

*= SecretKeyFactory.getInstance(*

*"PBKDF2WithHmacSHA256");*

*KeySpec spec = new PBEKeySpec(*

*SECRET\_KEY.toCharArray(), SALT.getBytes(),*

*65536, 256);*

*SecretKey tmp = factory.generateSecret(spec);*

*SecretKeySpec secretKey = new SecretKeySpec(*

*tmp.getEncoded(), "AES");*

*Cipher cipher = Cipher.getInstance(*

*"AES/CBC/PKCS5Padding");*

*cipher.init(Cipher.ENCRYPT\_MODE, secretKey,*

*ivspec);*

*return Base64.getEncoder().encodeToString(*

*cipher.doFinal(strToEncrypt.getBytes(*

*StandardCharsets.UTF\_8)));*

*}*

*catch (Exception e) {*

*System.out.println("Error while encrypting: "*

*+ e.toString());*

*}*

*return null;*

*}*

*public static String decrypt(String strToDecrypt)*

*{*

*try {*

*byte[] iv = { 0, 0, 0, 0, 0, 0, 0, 0,*

*0, 0, 0, 0, 0, 0, 0, 0 };*

*IvParameterSpec ivspec*

*= new IvParameterSpec(iv);*

*SecretKeyFactory factory*

*= SecretKeyFactory.getInstance(*

*"PBKDF2WithHmacSHA256");*

*KeySpec spec = new PBEKeySpec(*

*SECRET\_KEY.toCharArray(), SALT.getBytes(),*

*65536, 256);*

*SecretKey tmp = factory.generateSecret(spec);*

*SecretKeySpec secretKey = new SecretKeySpec(*

*tmp.getEncoded(), "AES");*

*Cipher cipher = Cipher.getInstance(*

*"AES/CBC/PKCS5PADDING");*

*cipher.init(Cipher.DECRYPT\_MODE, secretKey,*

*ivspec);*

*return new String(cipher.doFinal(*

*Base64.getDecoder().decode(strToDecrypt)));*

*}*

*catch (Exception e) {*

*System.out.println("Error while decrypting: "*

*+ e.toString());*

*}*

*return null;*

*}*

*}*

*public class Main {*

*public static void main(String[] args)*

*{*

*String originalString = "GeeksforGeeks";*

*String encryptedString= AES.encrypt(originalString);*

*String decryptedString= AES.decrypt(encryptedString);*

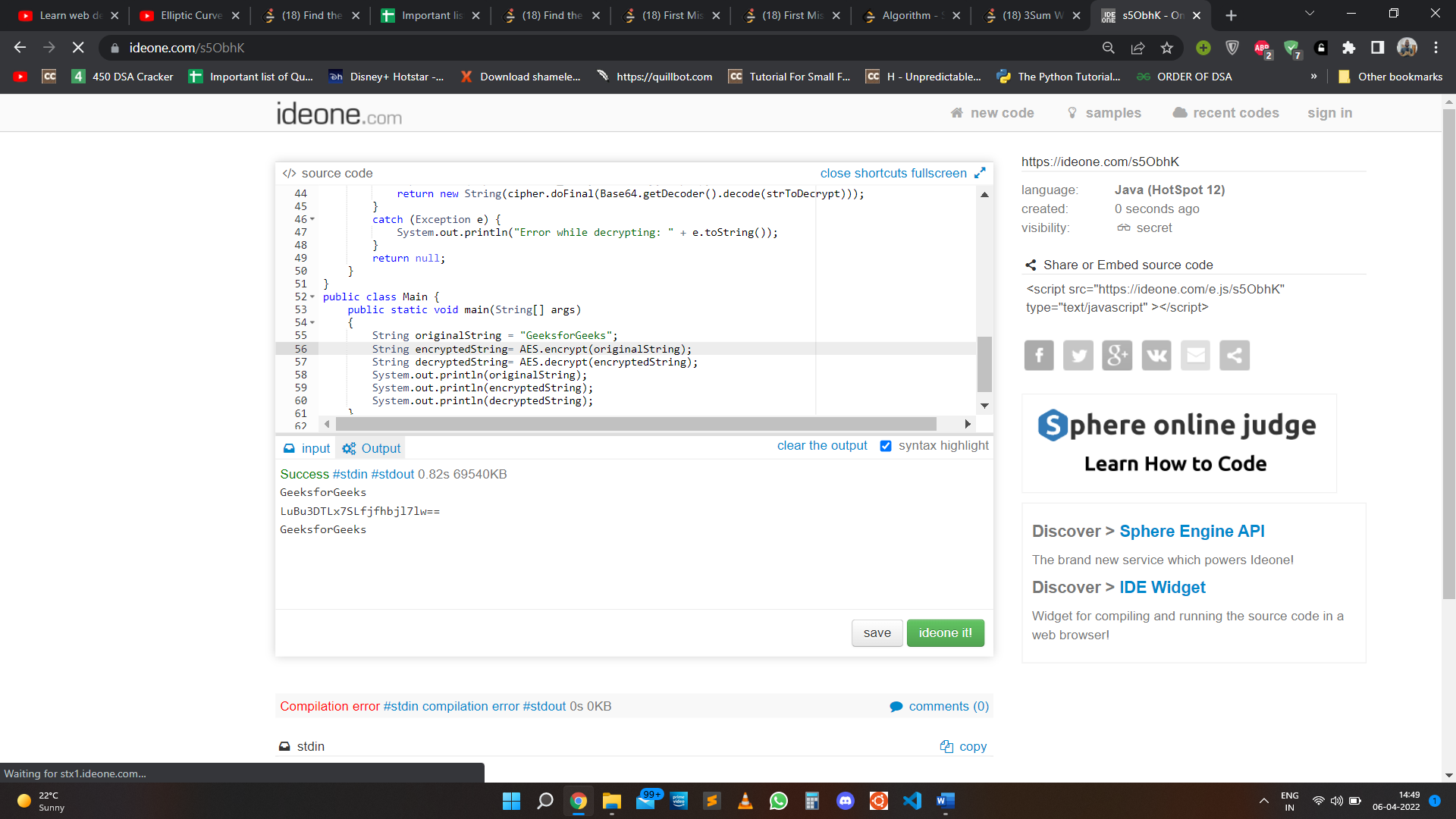
*System.out.println(originalString);*

*System.out.println(encryptedString);*

*System.out.println(decryptedString);*

*}*

*}*

**Output:**

**2. Write a program to implement the concept of RSA algorithm..**

**Code :**

*#include<stdio.h>*

*#include<math.h>*

*int gcd(int a, int h)*

*{*

*int temp;*

*while (1)*

*{*

*temp = a%h;*

*if (temp == 0)*

*return h;*

*a = h;*

*h = temp;*

*}*

*}*

*int main()*

*{*

*double p = 3,q = 7,n = p\*q,e = 2;*

*double phi = (p-1)\*(q-1);*

*while (e < phi)*

*{*

*if (gcd(e, phi)==1)*

*break;*

*else*

*e++;*

*}*

*int k = 2;*

*double d = (1 + (k\*phi))/e;*

*double msg = 324;*

*printf("Message data = %lf", msg);*

*double c = pow(msg, e);*

*c = fmod(c, n);*

*printf("\nEncrypted data = %lf", c);*

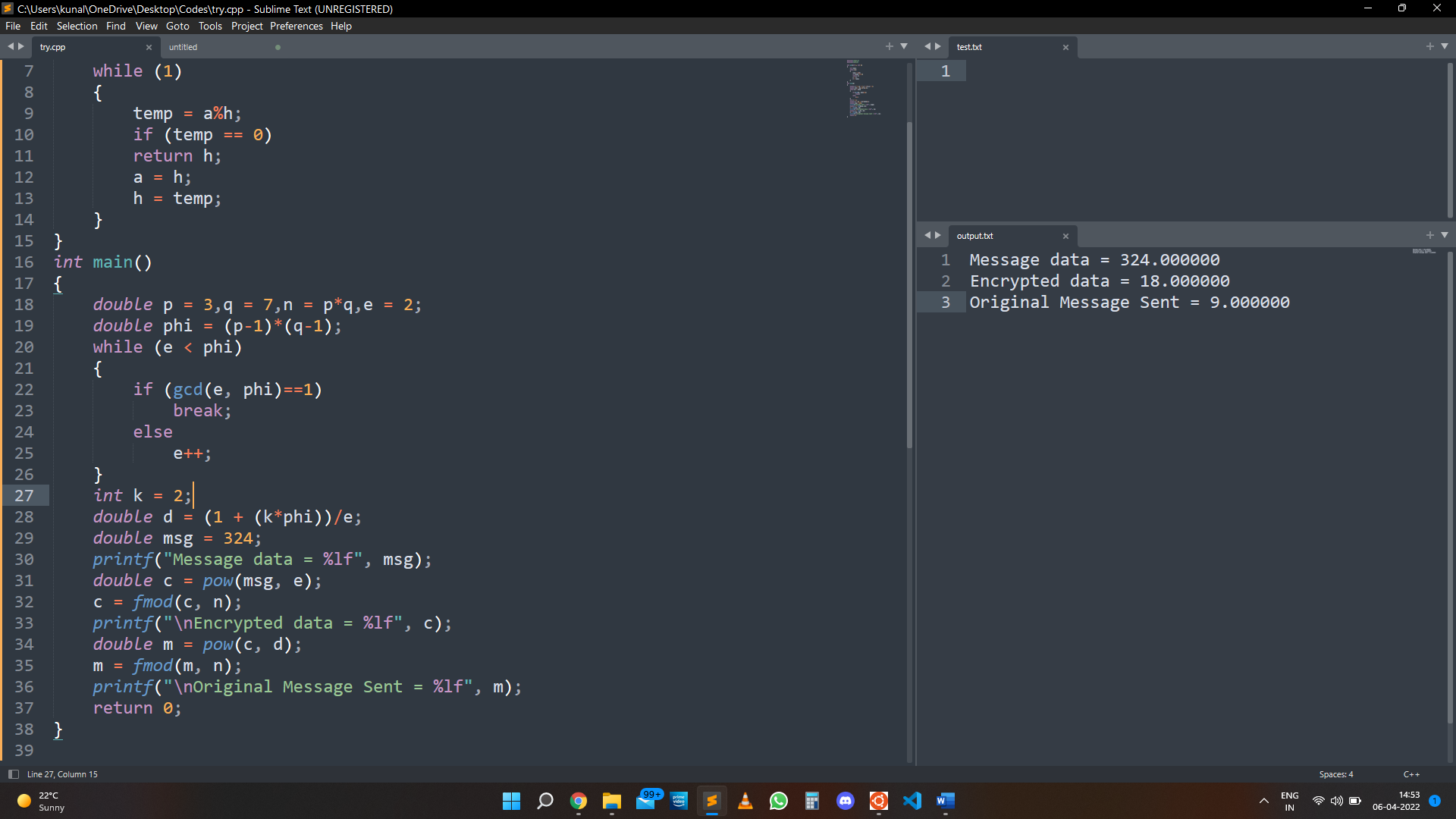
*double m = pow(c, d);*

*m = fmod(m, n);*

*printf("\nOriginal Message Sent = %lf", m);*

*return 0;*

*}*

**Output:**

**3. Write a program to implement the concept of Diffie-Hellman Key Exchange algorithm.**

**Code:**

*#include<stdio.h>*

*#include<math.h>*

*// Power function to return value of a ^ b mod P*

*long long int power(long long int a, long long int b, long long int P)*

*{*

*if (b == 1)*

*return a;*

*else*

*return (((long long int)pow(a, b)) % P);*

*}*

*//Driver program*

*int main()*

*{*

*long long int P, G, x, a, y, b, ka, kb;*

*// Both the persons will be agreed upon the*

*// public keys G and P*

*P = 28; // A prime number P is taken*

*printf("The value of P : %lld\n", P);*

*G = 12; // A primitive root for P, G is taken*

*printf("The value of G : %lld\n\n", G);*

*// Alice will choose the private key a*

*a = 4; // a is the chosen private key*

*printf("The private key a for Alice : %lld\n", a);*

*x = power(G, a, P); // gets the generated key*

*// Bob will choose the private key b*

*b = 3; // b is the chosen private key*

*printf("The private key b for Bob : %lld\n\n", b);*

*y = power(G, b, P); // gets the generated key*

*// Generating the secret key after the exchange of keys*

*ka = power(y, a, P); // Secret key for Alice*

*kb = power(x, b, P); // Secret key for Bob*

*printf("Secret key for the Alice is : %lld\n", ka);*

*printf("Secret Key for the Bob is : %lld\n", kb);*

*return 0;*

*}*

**Output:** 