DAYANANDA SAGAR College OF ENGINEERING

(An Autonomous Institution affiliated to

Visvesvaraya Technological University, Belgravia)

Department of Artificial Intelligence and Machine Learning

2022-2023

SIXTH SEMESTER

CYBER SECURITY LAB MANUAL

Sub Code: 20AI6DLCSL

COURSE OBJECTIVES AND COURSE OUTCOMES:

COURSE OBJECTIVES:

1. To be familiar with different types of Tools and methods used in Cyber Crime.
2. To be fluent with various security measures for handling different types of Cyber-attacks.
3. To be able to analyze and implement protection and prevention of Cyber CrimeAttacks.

Course Outcomes: At the end of the course, student will be able to:

CO1	Analyze and apply the security features on web browsers
CO2	Analyze the apply security vulnerabilities of E -Applications
CO3	Analyze and apply different Cyber Security tools
CO4	Analyze the apply Security issues in windows

ExpT.No		Contents of the Experiment	Hours
1	a	Implement Playfair Cipher in C Language.	1
	b	Explore the Quickstego Tool for Hiding and Recovering thetext and image based Information Using.	1
2	a	Analyze and scan the System Vulnerabilities using MicrosoftBaseline Security Analyzer (MBSA) Tool.	1
	b	Implement RSA algorithm using C Language.	1
3	a	Write the steps to Download a website using Website Copier tool (HTTrack).	1
	b	Implement Caesar Cipher in C Language.	1
4		Implement DES algorithm in Java.	2
5	a	Write a program to illustrate Buffer overflow attack.	2
	b	Explore Compare It Tool to Compare of two files for Forensic Investigation.	1
6		write a C program to implement the hill cipher substitution techniques	2
7	a	Implement the Diffie-Hellman Key Exchange algorithm using C language.	1
	b	Write the step by step procedure for Hiding and extracting any Text file behind an image file using Command Prompt.	1
8	a	Setup a honey pot and monitor the honeypot on network (KF Sensor)	1
	b	Explore the Snow Tool for hiding the information in Text File.	1

Text Book:

1.SunitBelapure and Nina Godbole, "Cyber Security: Understanding Cyber Crimes, Computer Forensics And Legal Perspectives", Wiley India Pvt Ltd, ISBN: 978-81-265-21791, Publish Date 2013.

2. Michael E. Whitman and Herbert J. Mattord,"Principles of Information Security Fourth Edition", Course Technology, Cengage Learning, ISBN-13: 978-1-111-13821-9, Publish Date 2012.

EXPT.NO 1(A)	Implement Playfair Cipher in C Language.

AIM:

To write a C program to implement the Playfair Substitution technique.

DESCRIPTION:

The Playfair cipher starts with creating a key table. The key table is a 5×5 grid of letters that will act as the key for encrypting your plaintext. Each of the 25 letters must be unique and one letter of the alphabet is omitted from the table (as there are 25 spots and 26 letters in the alphabet).

To encrypt a message, one would break the message into digrams (groups of 2 letters) such that, for example, "HelloWorld" becomes "HE LL OW OR LD", and map them out on the key table. The two letters of the diagram are considered as the opposite corners of a rectangle in the key table. Note the relative position of the corners of this rectangle. Then apply the following 4 rules, in order, to each pair of letters in the plaintext:

1. If both letters are the same (or only one letter is left), add an "X" after the first letter
2. If the letters appear on the same row of your table, replace them with the letters to their immediate right respectively
3. If the letters appear on the same column of your table, replace them with the letters immediately below respectively
4. If the letters are not on the same row or column, replace them with the letters on the same row respectively but at the other pair of corners of the rectangle defined by the original pair.

EXAMPLE:

D. Playfair Cipher

Example1: Plaintext: CRYPTO IS TOO EASY **Key =** INFOSEC **Ciphertext:** ??

Grouped text: CR YP TO IS TO XO EA SY

Ciphertext: AQ TV YB NI YB YF CB OZ

I / J	N	F	O	S
E	C	A	B	D
G	H	K	L	M
P	Q	R	T	U
V	W	X	Y	Z

ALGORITHM:

STEP-1: Read the plain text from the user.

STEP-2: Read the keyword from the user.

STEP-3: Arrange the keyword without duplicates in a 5*5 matrix in the row order and fill the remaining cells with missed out letters in alphabetical order. Note that 'i' and 'j' takes the same cell.

STEP-4: Group the plain text in pairs and match the corresponding corner letters by forming a rectangular grid.

STEP-5: Display the obtained cipher text.

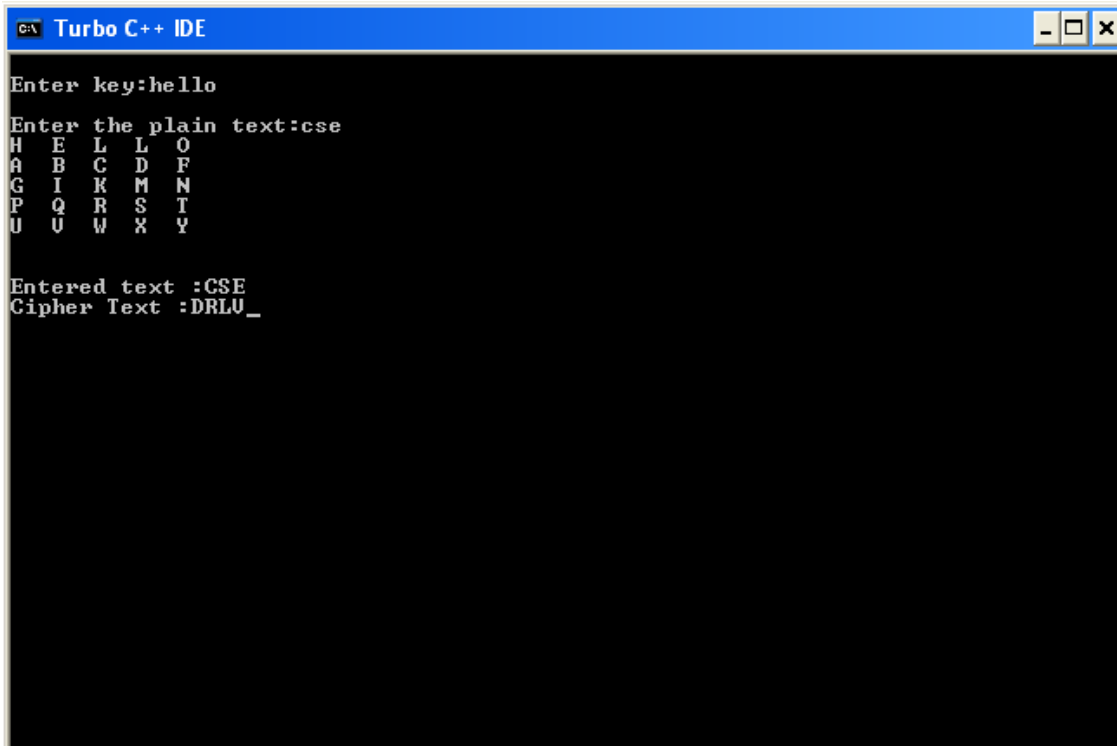
PROGRAM: (Playfair Cipher)

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
#include<ctype.h>
#define MX 5
void playfair(char ch1,char ch2, char key[MX][MX])
{
     int i,j,w,x,y,z;
     FILE *out;
if((out=fopen("cipher.txt","a+"))==NULL)
{
     printf("File Currupted.");
}
for(i=0;i<MX;i++)
{
     for(j=0;j<MX;j++)
     {
     if(ch1==key[i][j])
     {
     w=i;
     x=j;
     }
     else if(ch2==key[i][j])
     {
     y=i;
     z=j;

}}}
//printf("%d%d %d%d",w,x,y,z);
if(w==y)
{
     x=(x+1)%5;z=(z+1)%5;
     printf("%c%c",key[w][x],key[y][z]);
     fprintf(out, "%c%c",key[w][x],key[y][z]);
}
else if(x==z)
{
```

```
        w=(w+1)%5;y=(y+1)%5;
        printf("%c%c",key[w][x],key[y][z]);
        fprintf(out,  "%c%c",key[w][x],key[y][z]);
    }
    else
    {
        printf("%c%c",key[w][z],key[y][x]);
        fprintf(out,  "%c%c",key[w][z],key[y][x]);
    }
    fclose(out);
    }
    void main()
    {
        int i,j,k=0,l,m=0,n;
        char   key[MX][MX],keyminus[25],keystr[10],str[25]={0};
        char
        alpa[26]={'A','B','C','D','E','F','G','H','I','J','K','L'
        ,'M','N','O','P','Q','R','S','T','U','V','W','X','Y','Z'}
        ;
        clrscr();
        printf("\nEnter key:");
        gets(keystr);
        printf("\nEnter the plain text:");
        gets(str);
        n=strlen(keystr);
        //convert the characters to uppertext
        for (i=0; i<n; i++)
        {
            if(keystr[i]=='j')keystr[i]='i';
            else  if(keystr[i]=='J')keystr[i]='I';
            keystr[i] = toupper(keystr[i]);
        }
        //convert all the characters of plaintext to uppertext
        for (i=0; i<strlen(str); i++)
        {
            if(str[i]=='j')str[i]='i';
            else  if(str[i]=='J')str[i]='I';
            str[i] = toupper(str[i]);
        }
        j=0;
        for(i=0;i<26;i++)
        {
            for(k=0;k<n;k++)
            {
            if(keystr[k]==alpa[i])
            break;
            else  if(alpa[i]=='J')
            break;
            }
            if(k==n)
            {
            keyminus[j]=alpa[i];j++;
            }
        }
```

```
        //construct key keymatrix
        k=0;
        for(i=0;i<MX;i++)
        {
              for(j=0;j<MX;j++)
              {
              if(k<n)
              {
              key[i][j]=keystr[k];
              k++;}
              else
              {
              key[i][j]=keyminus[m];m++;
              }
              printf("%c  ",key[i][j]);
              }
              printf("\n");
        }
        printf("\n\nEntered text :%s\nCipher Text :",str);
        for(i=0;i<strlen(str);i++)
        {
        if(str[i]=='J')str[i]='I';
        if(str[i+1]=='\0')
        playfair(str[i],'X',key);
        else
        {
              if(str[i+1]=='J')str[i+1]='I';
              if(str[i]==str[i+1])
              playfair(str[i],'X',key);
              else
              {
              playfair(str[i],str[i+1],key);i++;
              }}

        }
    getch();
    }
```

OUTPUT:



RESULT:

Thus the Playfair cipher substitution technique had been implemented successfully.

EXPT.NO 1(B)	Explore the Quickstego Tool for Hiding and Recovering the text and image based Information Using.	DATE

AIM:

The main aim is to hide and recover the information using QUICKSTEGO TOOL.

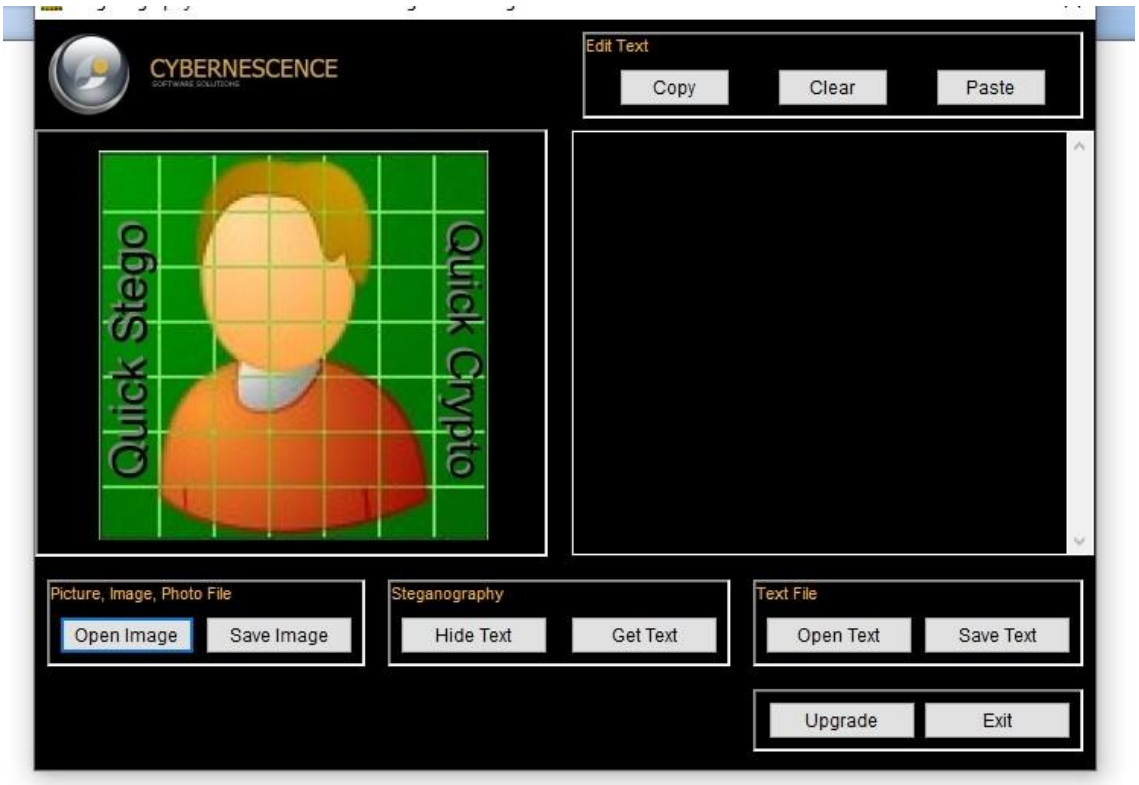
PROCEDURE:

- Steganography is the science of writing hidden messages in such a way that no one apart from the sender and intended recipient even realizes there is a hidden message.
- QuickStego lets you hide text in pictures so that only other users of QuickStego can retrieve and read the hidden secret messages. Once text is hidden in an image the saved picture is still a 'picture', it will load just like any other image and appear as it did before.

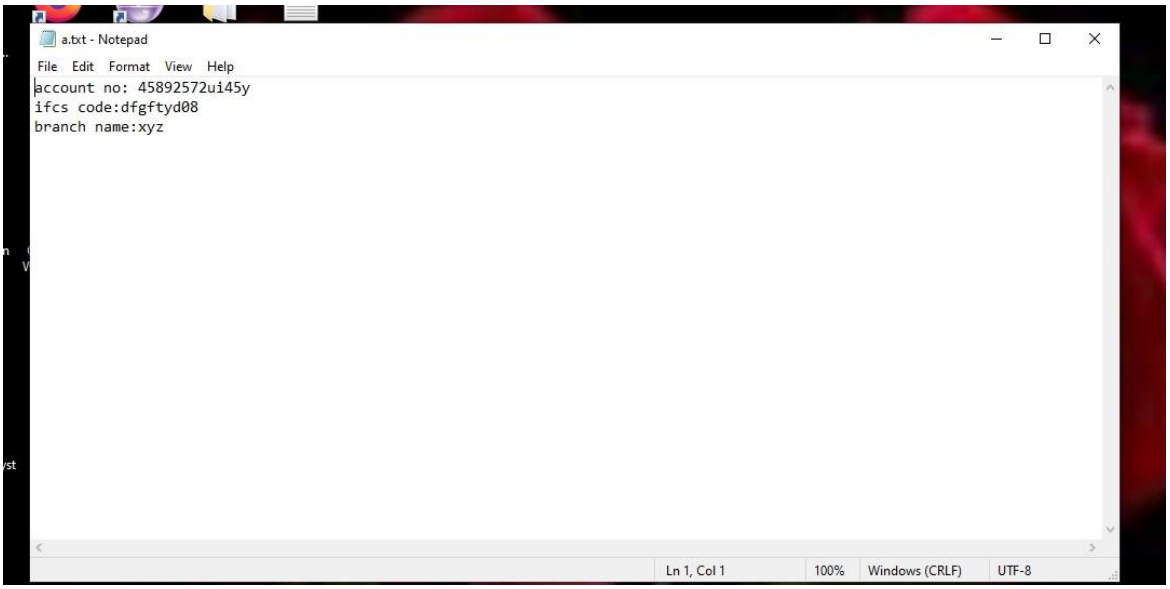
Step 1 : Download the QuickStego tool

Step 2 : Install the QuickStego tool and launch the desktop icon

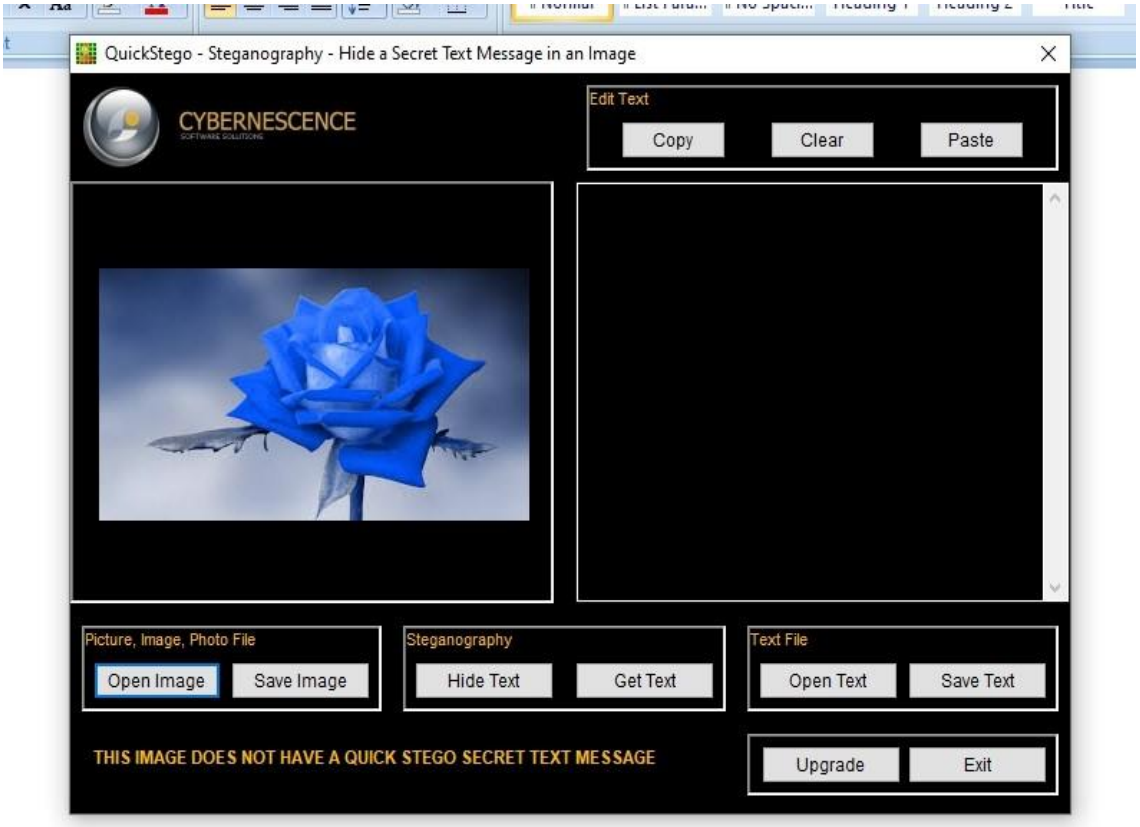
Step 3 : Open the QuickStego application



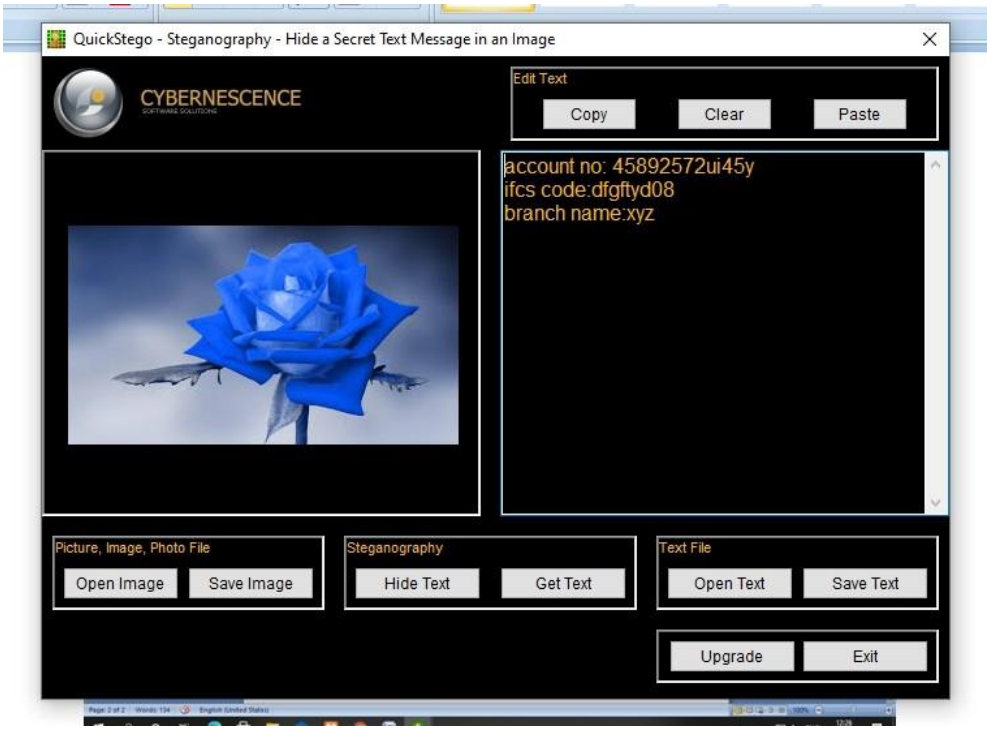
Step 4 : Create a text file or otherwise directly we can give the text data here we are creating a secrete text file with the extension .txt to upload in the image



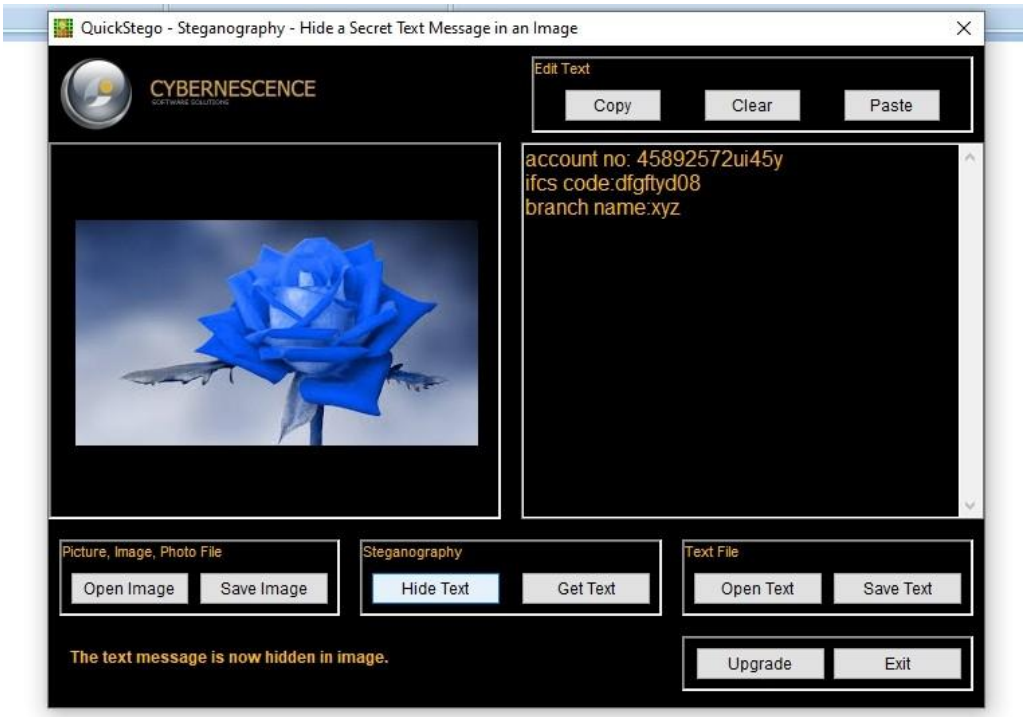
Step 5 : upload the image file to the QuickStego application



Step 6: Upload the text file to the QuickStego application



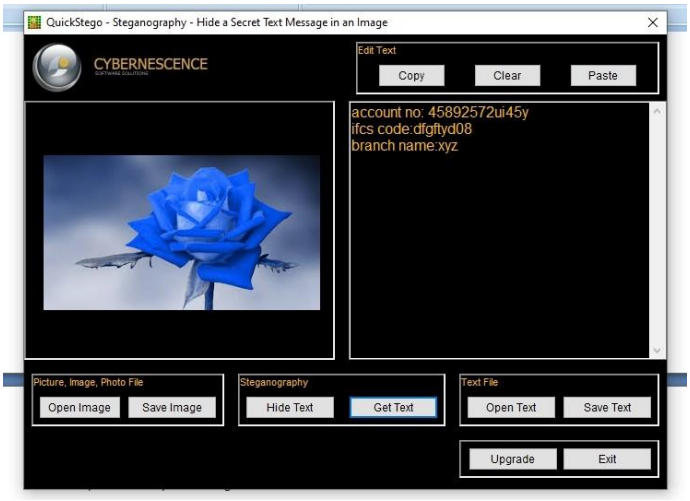
Step 7 : Click hide text to hide the text document to image



Step 8 : Click save image to upload the secret data to image a new image file is created and saved

Step 9: Now close the stego application and open it again

Step 10 : Now open the newly saved image and click the Get Text



RESULT:

The main aim is to hide and recover the information using QUICKSTEGO TOOL is completed successfully.

EXPT.NO 2(A)	Analyze and scan the System Vulnerabilities using Microsoft Baseline Security Analyzer (MBSA) Tool.	DATE:

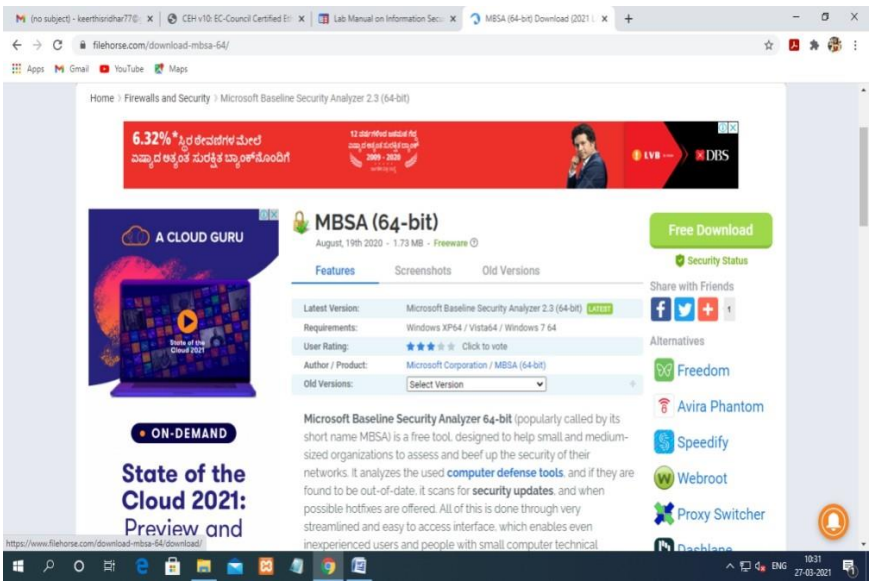
AIM:

The main aim is to scan the system vulnerabilities using Microsoft baseline security analyzer **(MBSA)**

PROCEDURE:

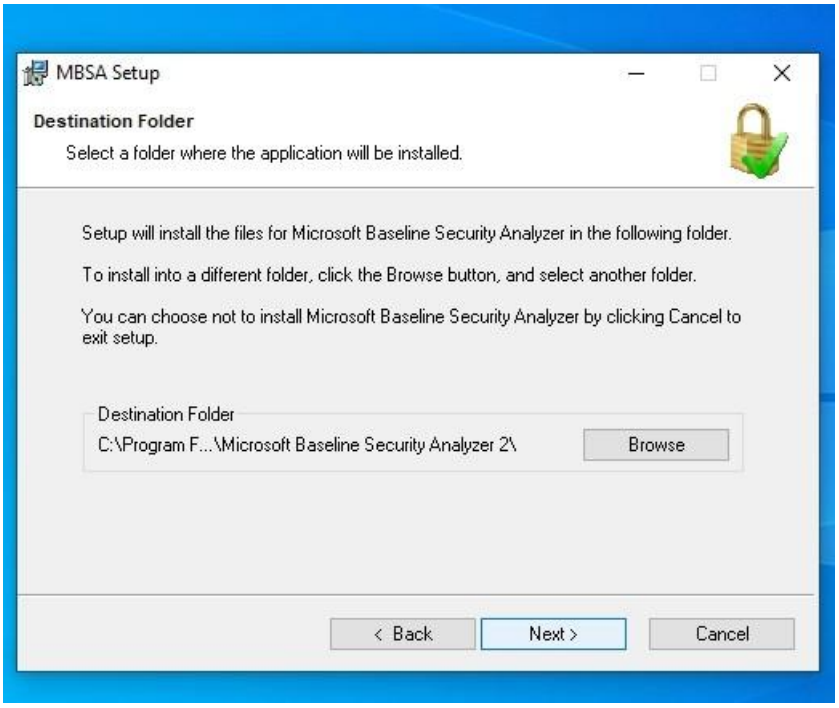
- Microsoft Baseline Security Analyzer (MBSA) is used to verify patch compliance. MBSA also performed several other security checks for Windows, IIS, and SQL Server.
- Unfortunately, the logic behind these additional checks had not been actively maintained since Windows XP and Windows Server 2003.
- Changes in the products since then rendered many of these security checks obsolete and some of their recommendations counterproductive.
- MBSA was largely used in situations where neither Microsoft Update nor a local WSUS or Configuration Manager server was available, or as a compliance tool to ensure that all security updates were deployed to a managed environment.
- While MBSA version 2.3 introduced supports for Windows Server 2012 R2 and Windows 8.1, it has since been deprecated and no longer developed. MBSA 2.3 is not updated to fully support Windows 10 and Windows Server 2016.

Step 1: download the Microsoft Baseline Security Analyzer (MBSA)

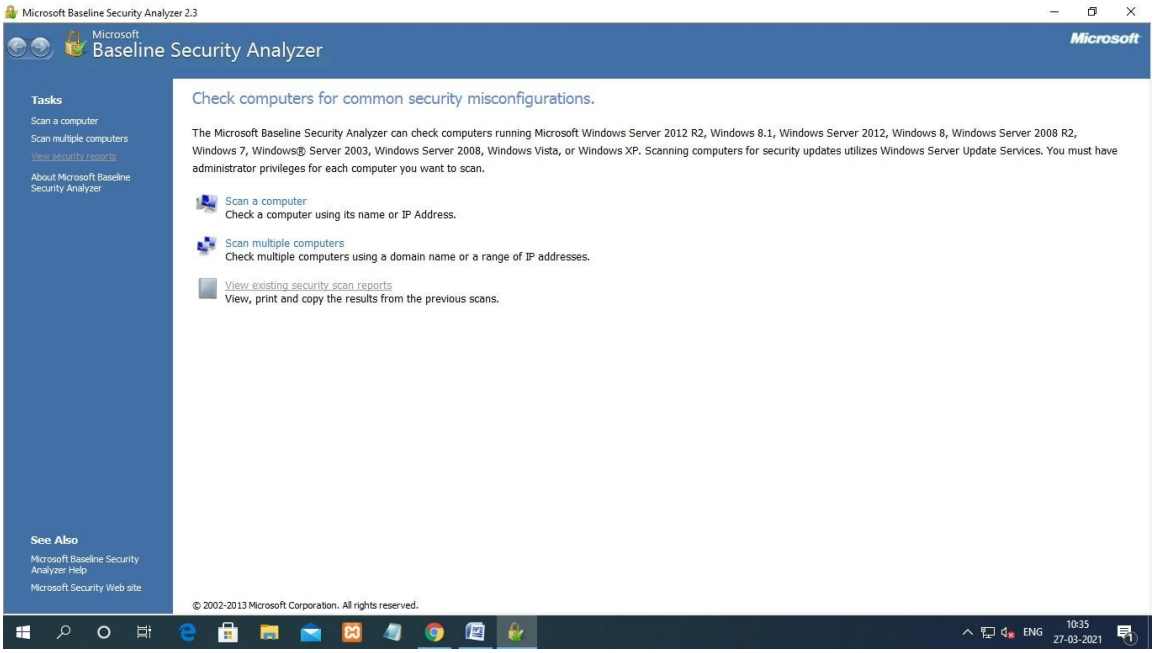


Step2: start and install the Microsoft Baseline Security Analyzer (MBSA) click next to start install

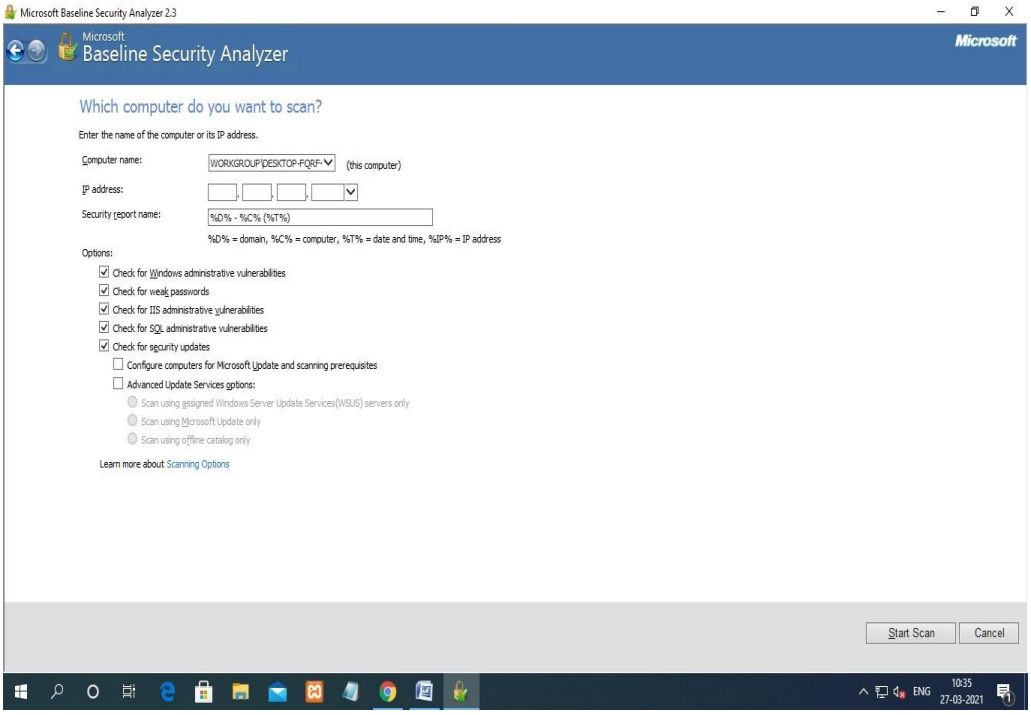
Step3: choose the location to install MBSA



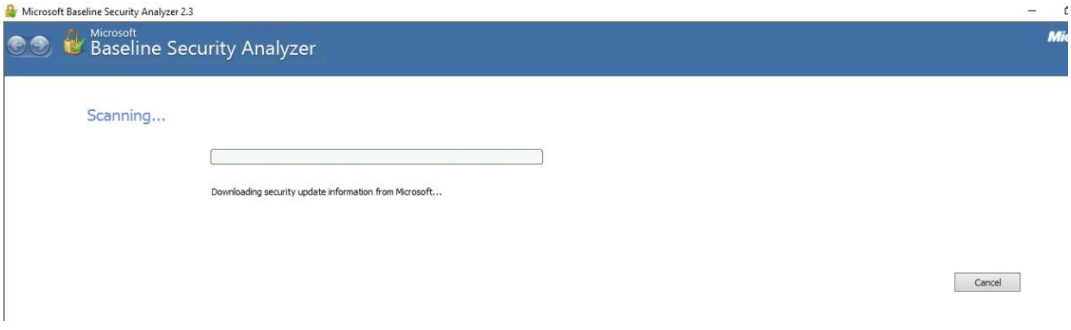
STEP 4:click the scan a computer to start the MBSA



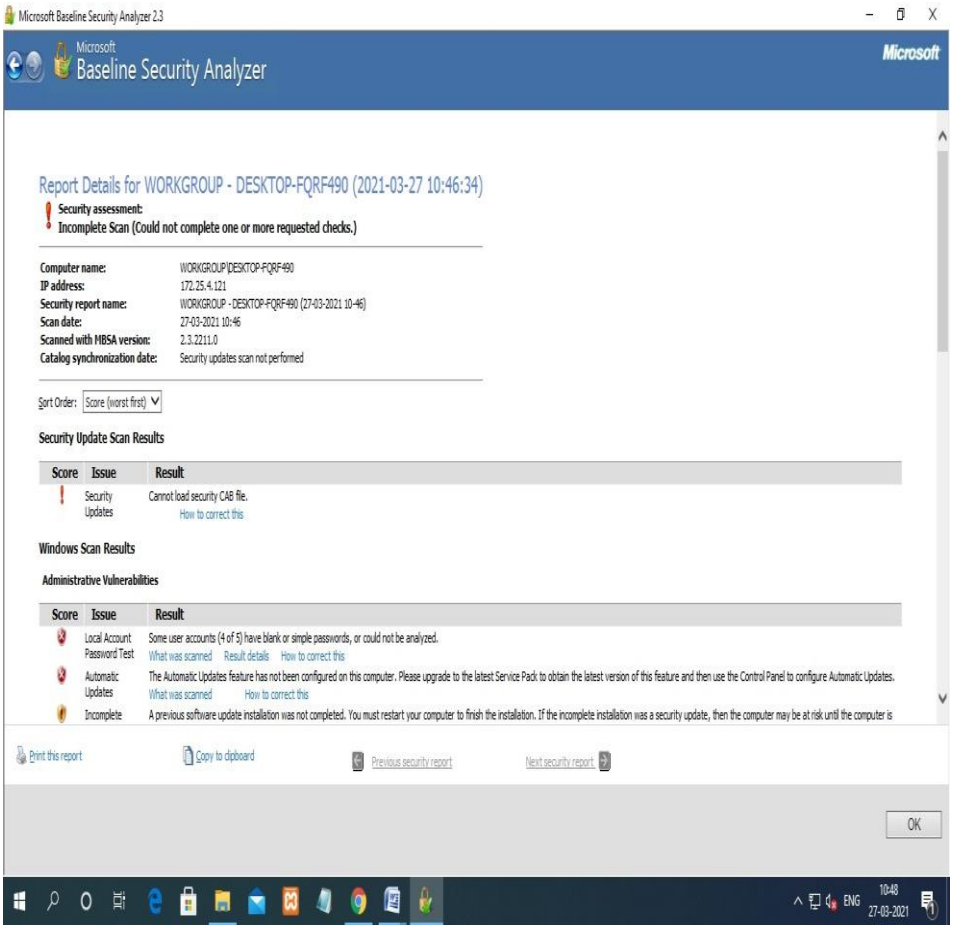
Step 5: provide the IP address and click start scan



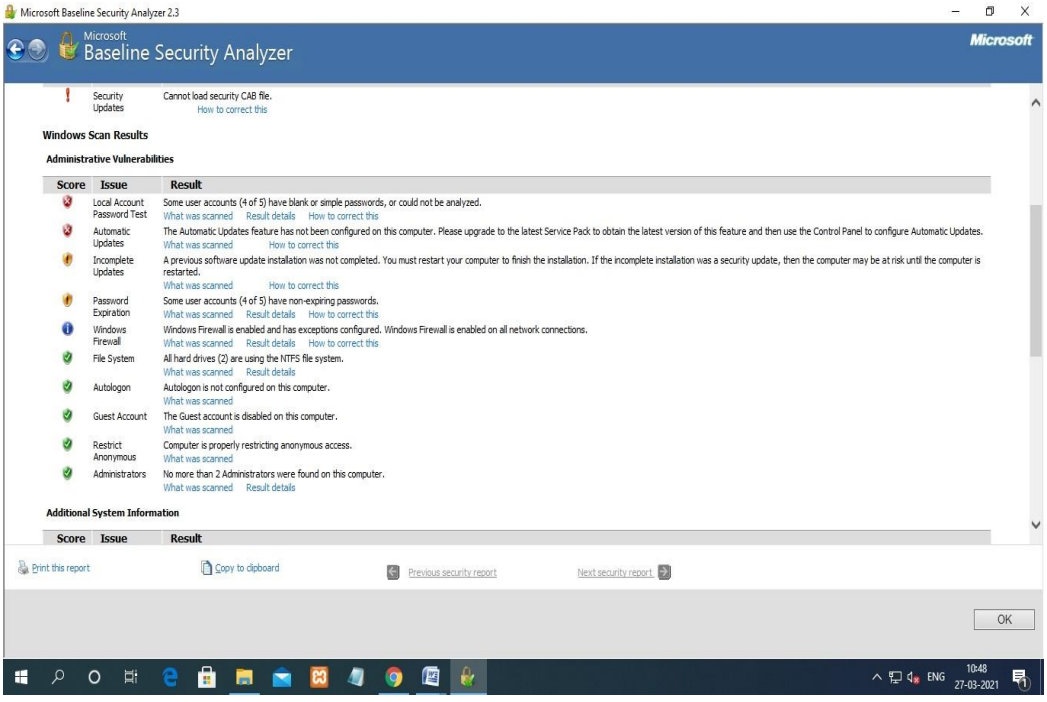
Step 7: The scanning process is GET STARTED



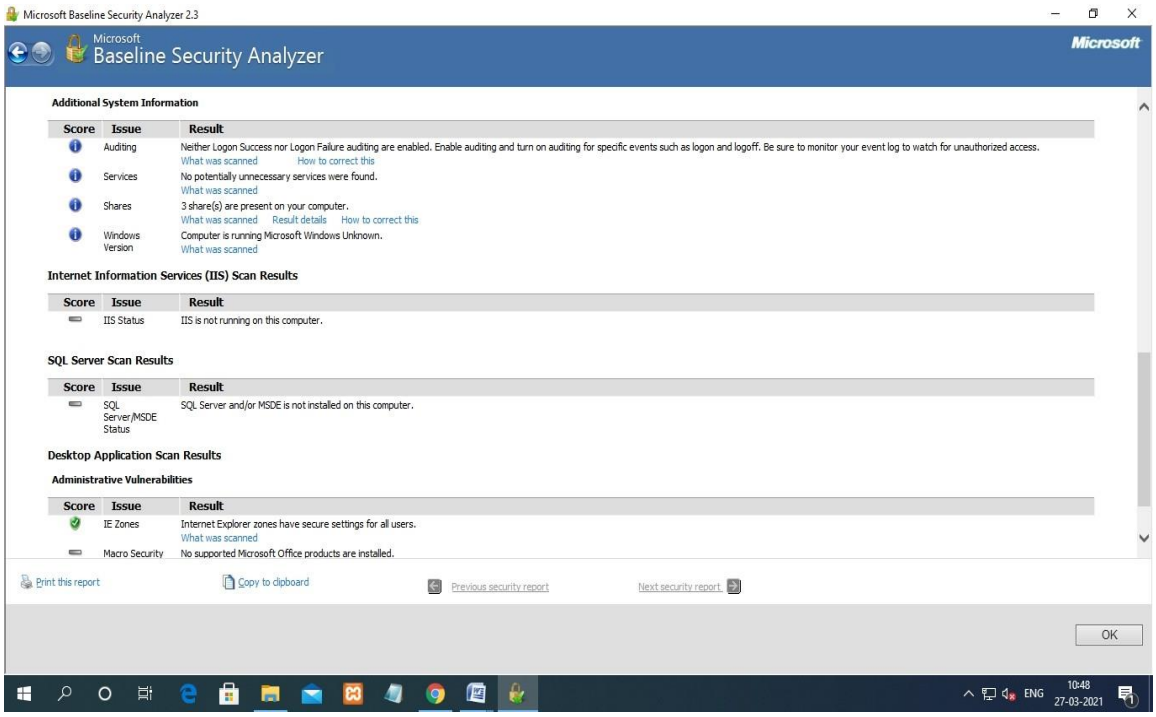
STEP 8: The detail report is generated for the system



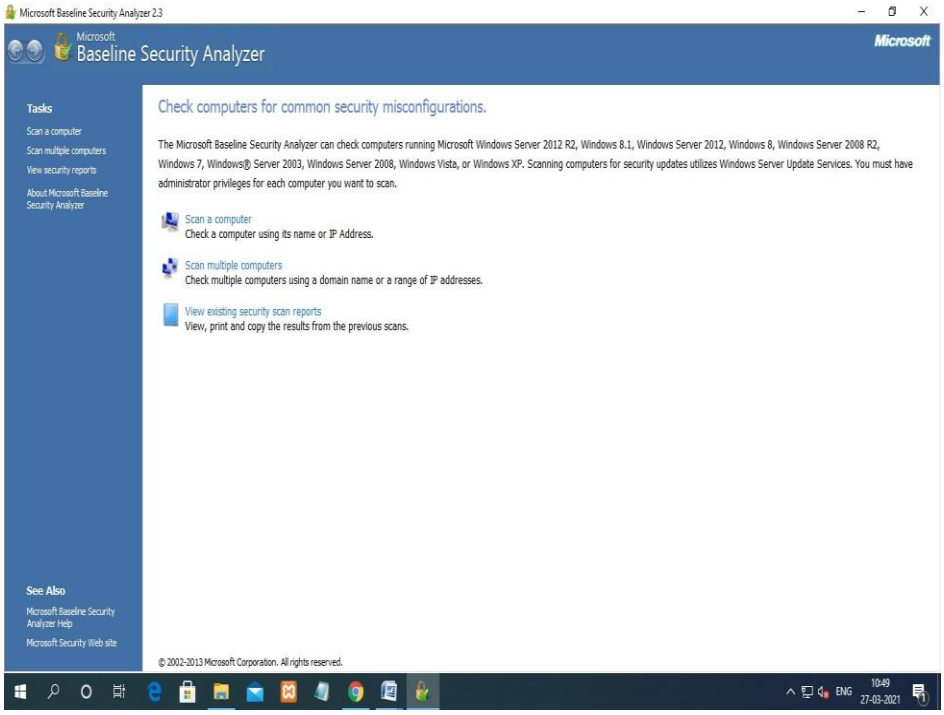
Result about the administrative vulnerabilities



Result about the additional system information, IIS scans result, desktop application



And also we can view the existing scan report



RESULT:

The main aim is to scan the system vulnerabilities using Microsoft baseline security analyzer **(MBSA)** is completed successfully.

EXPT.NO 2(B)	Implement RSA algorithm using C Language.	DATE:

AIM:

To write a C program to implement the RSA encryption algorithm.

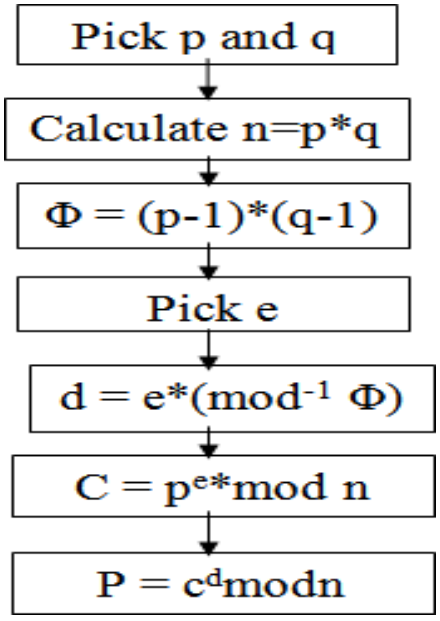
DESCRIPTION:

RSA is an algorithm used by modern computers to encrypt and decrypt messages. It is an asymmetric cryptographic algorithm. Asymmetric means that there are two different keys. This is also called public key cryptography, because one of them can be given toeveryone. A basic principle behind RSA is the observation that it is practical to find three very large positive integers e, d and n such that with modular exponentiation for all integer m:

$$(m^e)^d = m \pmod n$$

The public key is represented by the integers n and e; and, the private key, by the integer d. m represents the message. RSA involves a public key and a private key. The public key can be known by everyone and is used for encrypting messages. The intention is that messages encrypted with the public key can only be decrypted in a reasonable amount oftime using the private key.

EXAMPLE:



ALGORITHM:

STEP-1: Select two co-prime numbers as p and q.

STEP-2: Compute n as the product of p and q.

STEP-3: Compute (p-1)*(q-1) and store it in z.

STEP-4: Select a random prime number e that is less than that of z.

STEP-5: Compute the private key, d as e * mod^{-1}(z).

STEP-6: The cipher text is computed as $message^{e}$ *

mod n.**STEP-7:** Decryption is done as $cipher^{d}$mod n.

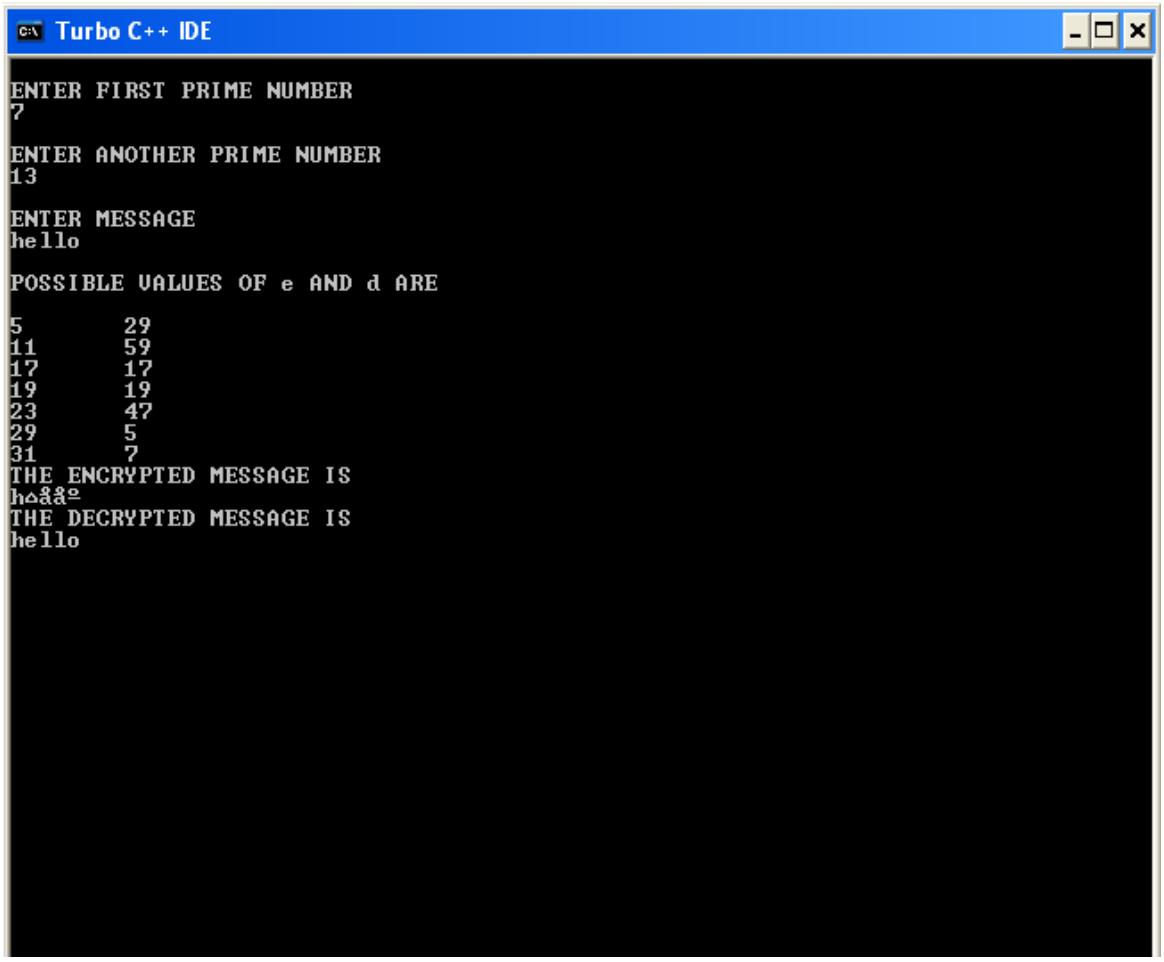
PROGRAM: (RSA)

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#include<math.h>
#include<string.h>
     long int
     p,q,n,t,flag,e[100],d[100],temp[100],j,m[100],en[100],i;
     char msg[100];
     int prime(long int);
     void ce();
     long int cd(long int);
     void encrypt();
     void decrypt();
void main()
{
     clrscr();
     printf("\nENTER FIRST PRIME NUMBER\n");
     scanf("%d",&p);
     flag=prime(p);
     if(flag==0)
       {
           printf("\nWRONG  INPUT\n");
           getch();
       }
       printf("\nENTER ANOTHER PRIME NUMBER\n");
       scanf("%d",&q);
       flag=prime(q);
       if(flag==0||p==q)
       {
        printf("\nWRONG  INPUT\n");
        getch();
       }
       printf("\nENTER  MESSAGE\n");
       fflush(stdin);
       scanf("%s",msg);
       for(i=0;msg[i]!=NULL;i++)
       m[i]=msg[i];
       n=p*q;
```

```
        t=(p-1)*(q-1);
        ce();
        printf("\nPOSSIBLE VALUES OF e AND d ARE\n");
        for(i=0;i<j-1;i++)
        printf("\n%ld\t%ld",e[i],d[i]);
        encrypt();
        decrypt();
        getch();
}
int prime(long int pr)
{
int  i;
j=sqrt(pr);
for(i=2;i<=j;i++)
{
if(pr%i==0)
return 0;
}
return 1;
}
void ce()
{
int k;
k=0;
for(i=2;i<t;i++)
{
 if(t%i==0)
 continue;
 flag=prime(i);
 if(flag==1&&i!=p&&i!=q)
 {
 e[k]=i;
 flag=cd(e[k]);
 if(flag>0)
 {
 d[k]=flag;
 k++;
 }
 if(k==99)
 break;
 } } }
 long int cd(long int x)
 {
 long int k=1;
 while(1)
 {
 k=k+t;
 if(k%x==0)
 return(k/x);
 } }
 void encrypt() {
 long int pt,ct,key=e[0],k,len;
 i=0;
 len=strlen(msg);
```

```
while(i!=len) {
pt=m[i];
pt=pt-96;
k=1;
for(j=0;j<key;j++)
{ k=k*pt;
k=k%n;
}
temp[i]=k;
ct=k+96;
en[i]=ct;
i++;
}
en[i]=-1;
printf("\nTHE ENCRYPTED MESSAGE IS\n");
for(i=0;en[i]!=-1;i++)
printf("%c",en[i]);
}
void decrypt()
{
long int pt,ct,key=d[0],k;
i=0;
while(en[i]!=-1)
{
ct=temp[i];
k=1;
for(j=0;j<key;j++)
{
k=k*ct;
k=k%n;
}
pt=k+96;
m[i]=pt;
i++;
}
m[i]=-1;
printf("\nTHE DECRYPTED MESSAGE IS\n");
for(i=0;m[i]!=-1;i++)
printf("%c",m[i]);
}
```

OUTPUT:



RESULT:

Thus the C program to implement RSA encryption technique had been implementedsuccessfully

EXPT.NO 3(A)	Write the steps to Download a website using Website Copier tool (HTTrack)	DATE:

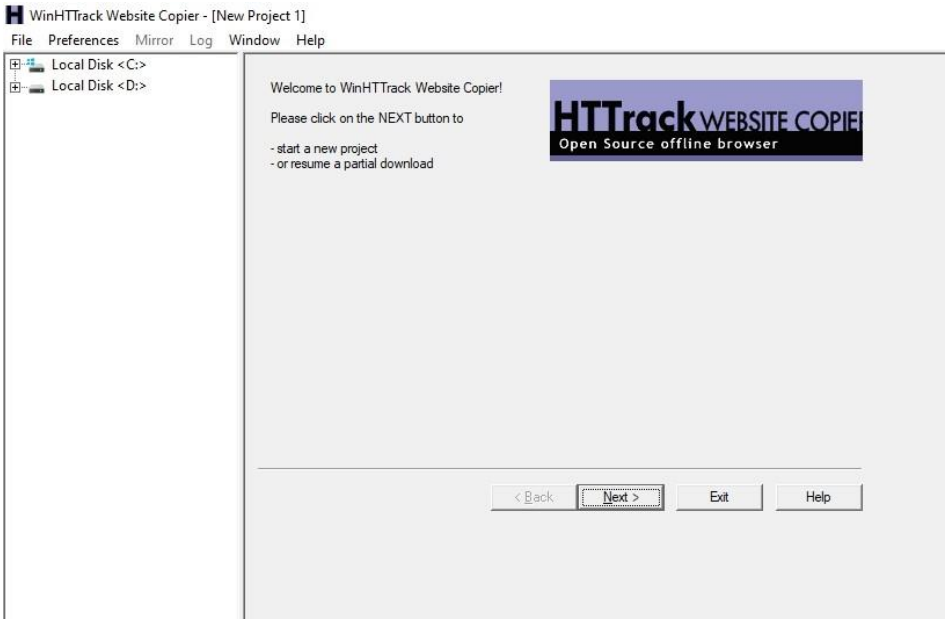
AIM:

The main aim is to downloading a website using website copier tool (HTTtack)

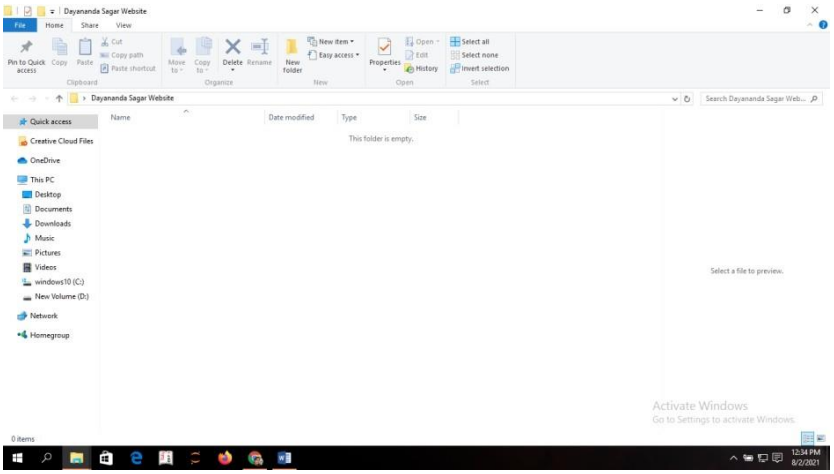
PROCEDURE:

- HTTrack is a free (GPL, libre/free software) and easy-to-use offline browser utility.
- It allows you to download a World Wide Web site from the Internet to a local directory, building recursively all directories, getting HTML, images, and other files from the server to your computer.
- HTTrack arranges the original site's relative link-structure.
- Simply open a page of the "mirrored" website in your browser, and you can browse the site from link to link, as if you were viewing it online.
- HTTrack can also update an existing mirrored site, and resume interrupted downloads. HTTrack is fully configurable, and has an integrated help system.
- WinHTTrack is the Windows (from Windows 2000 to Windows 10 and above) release of HTTrack, and WebHTTrack the Linux/Unix/BSD release.

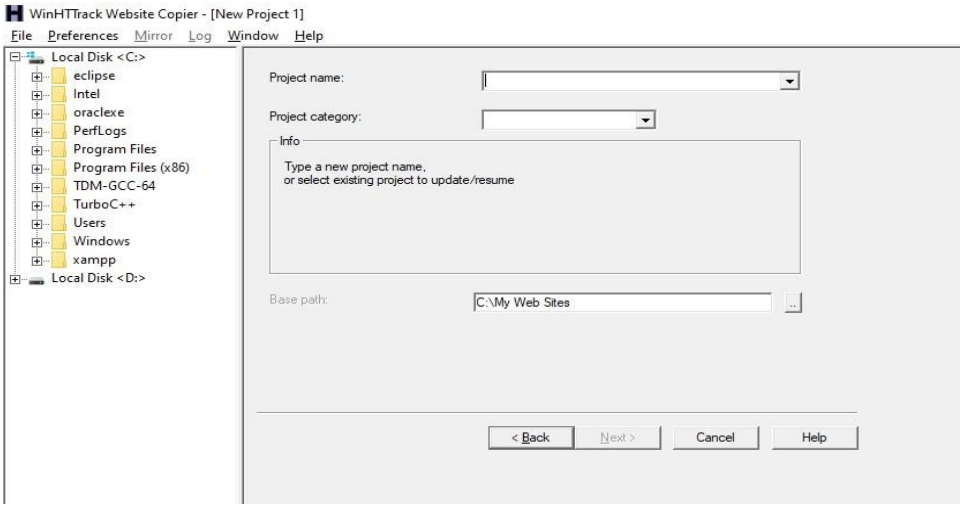
STEP 1: Install WinHTTrack



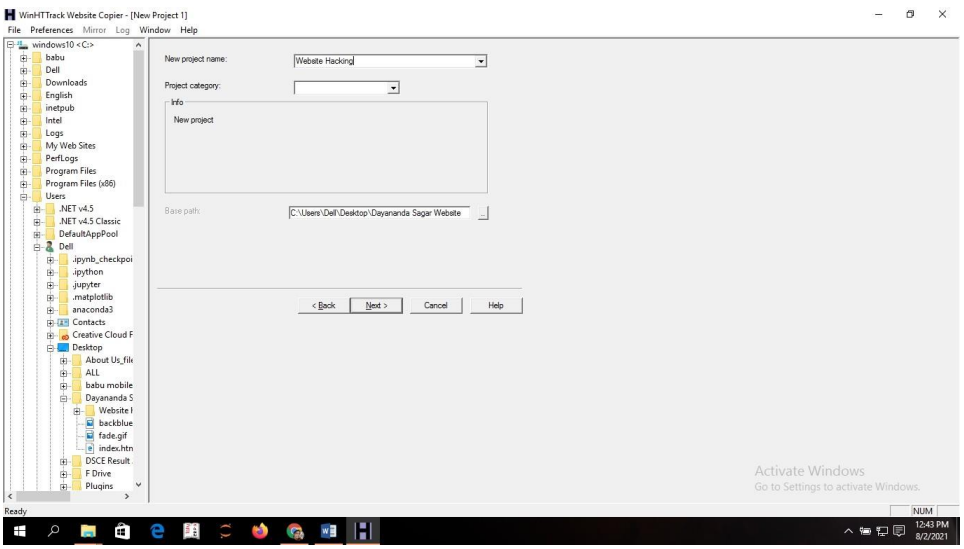
STEP 2: Create a folder on the Desktop and rename the folder
For Example: Folder name is “Dayananda Sagar Website”.
Open the folder “Dayananda Sagar Website”. The content of the folder is empty.



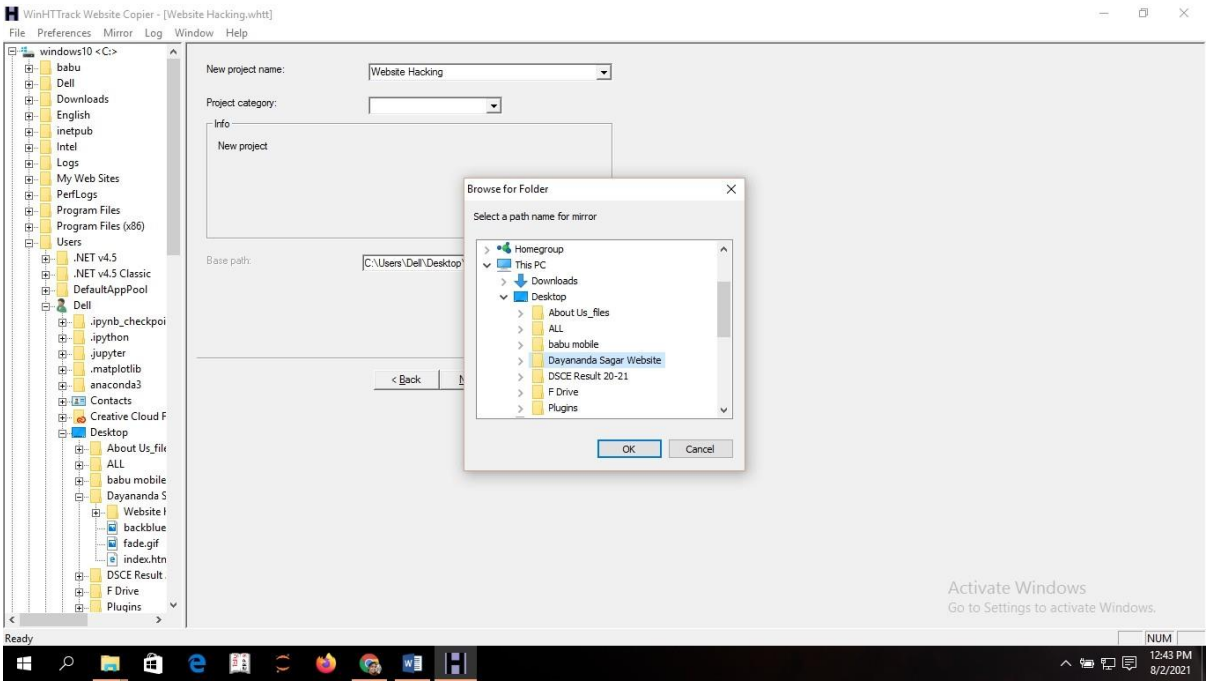
STEP 3: Select the new project from the file menu.



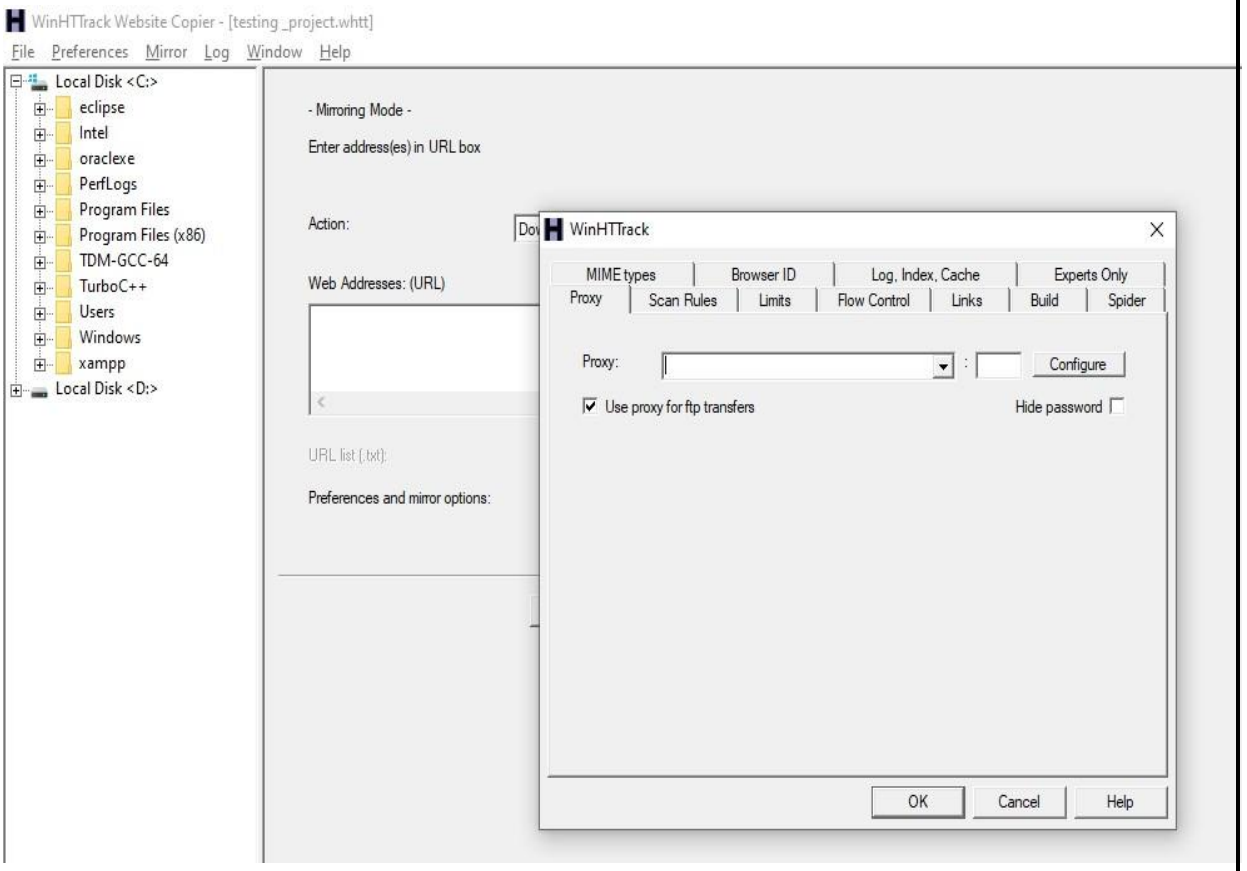
STEP 3: Enter the project name in new project field: **Example: Website hacking**



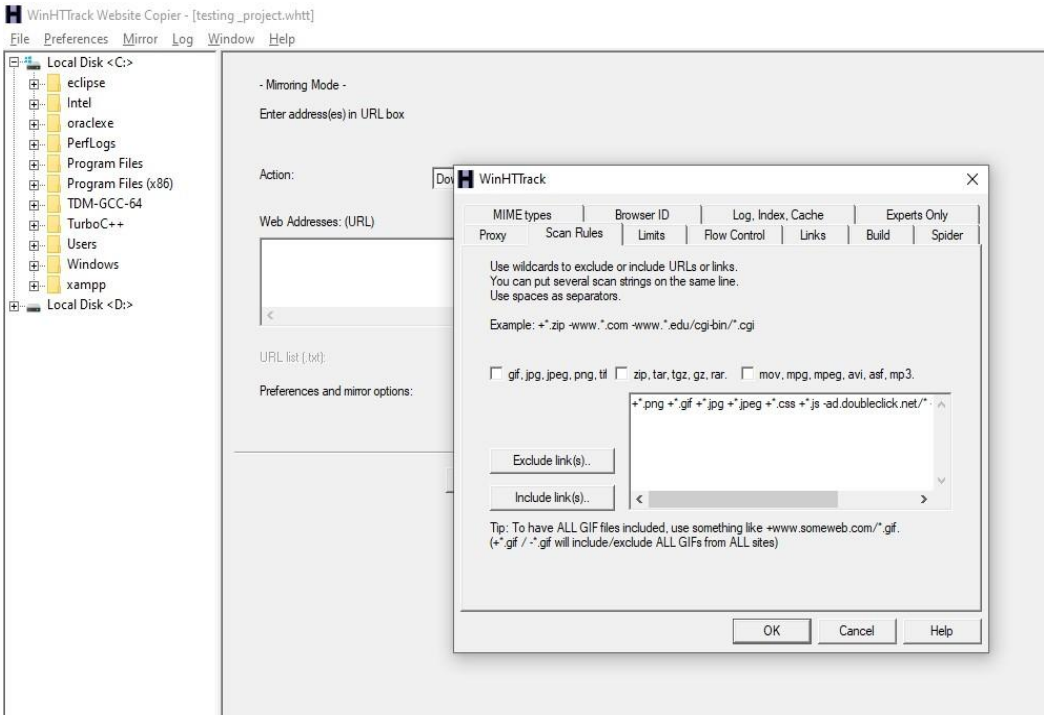
Step 4: Give the path where you need to download the files. In order to do this Click on Desktop and then click the folder “Dayananda Sagar Website”. Press OK



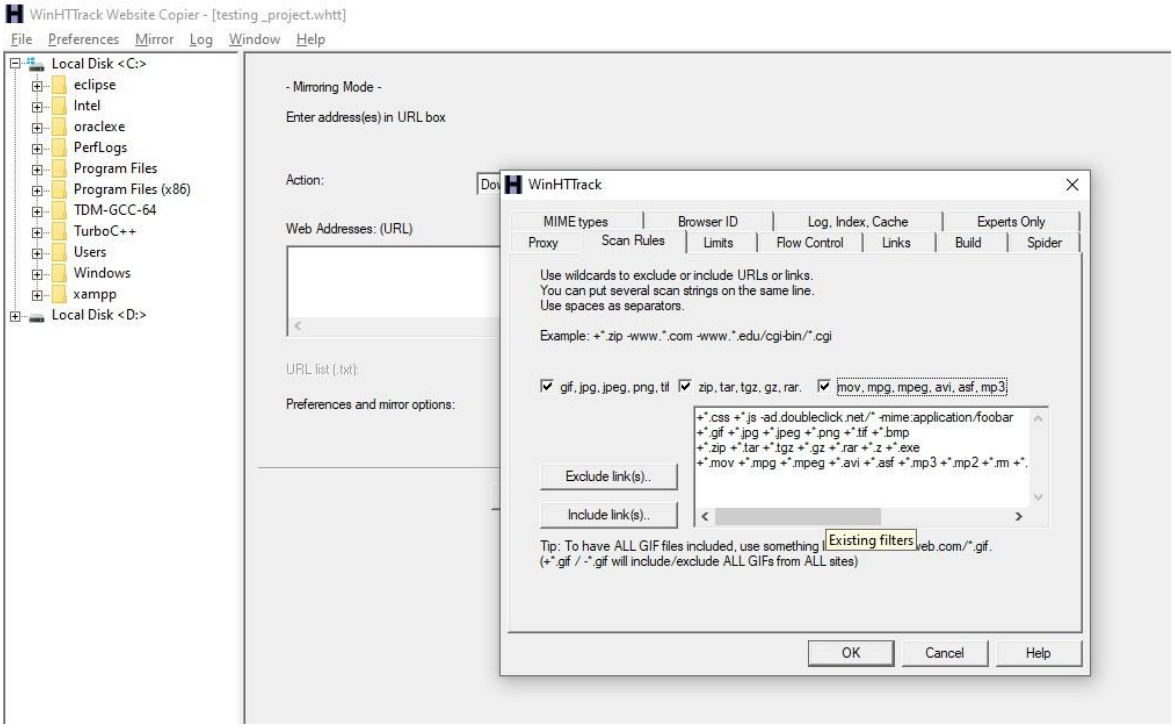
Step 5: WinHTTrack option window is opened select the scan rules



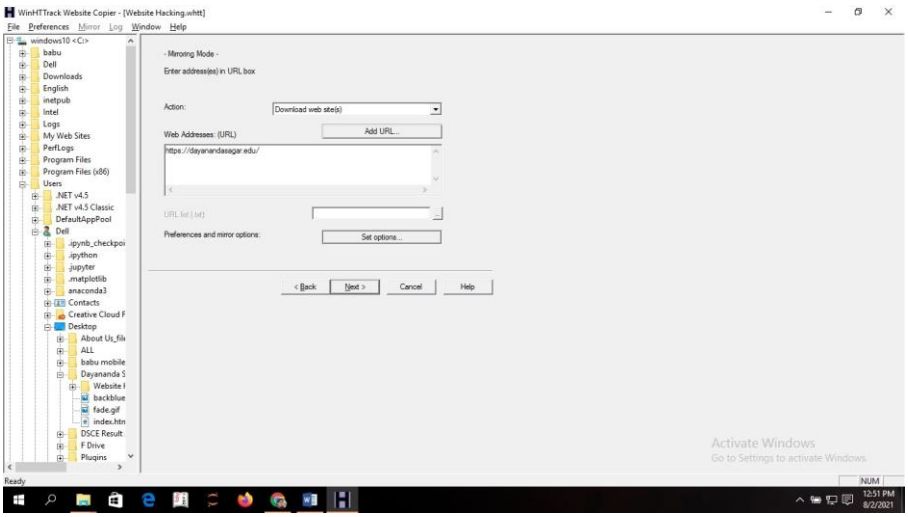
Step 6: Select all type of file to start the scan.



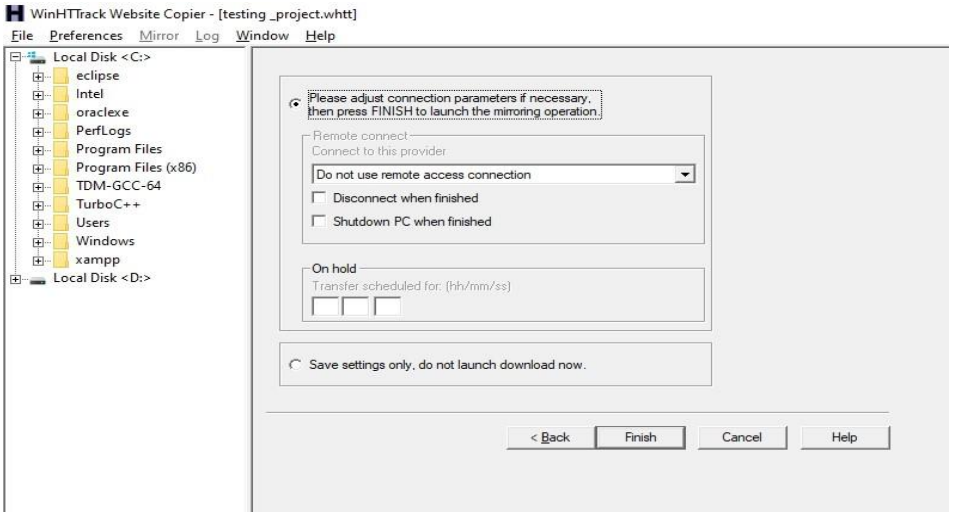
Step 7: Now all the extension is added for the scan.



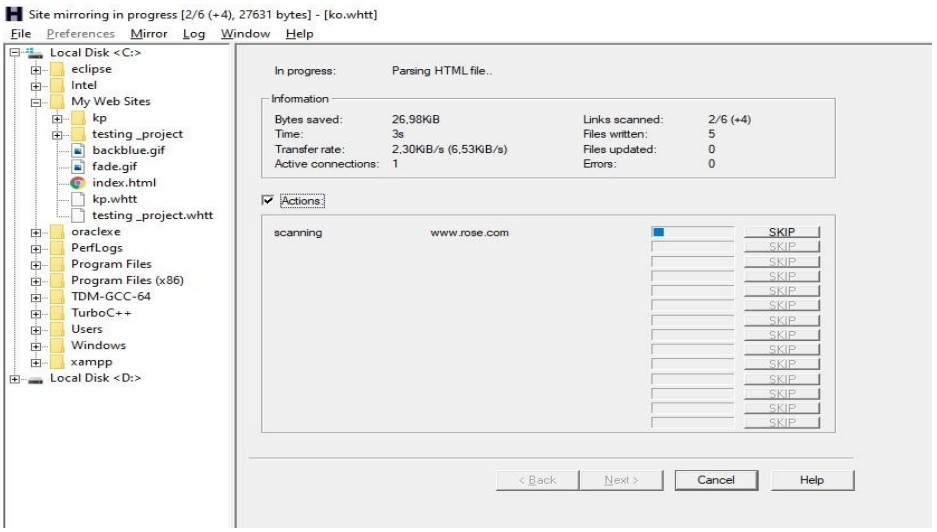
Step 8: Now type the URL address to scan



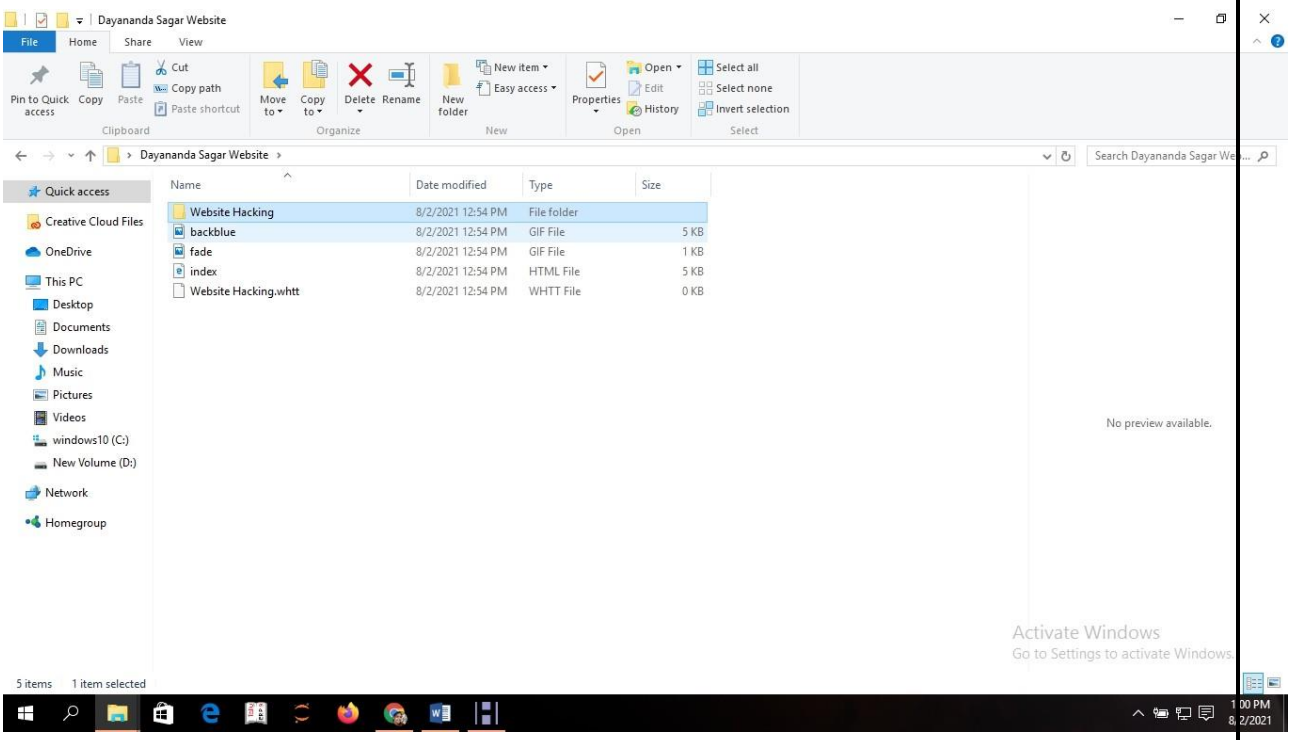
Step 9: Enable the connection adjustment if needed and click the finish button.



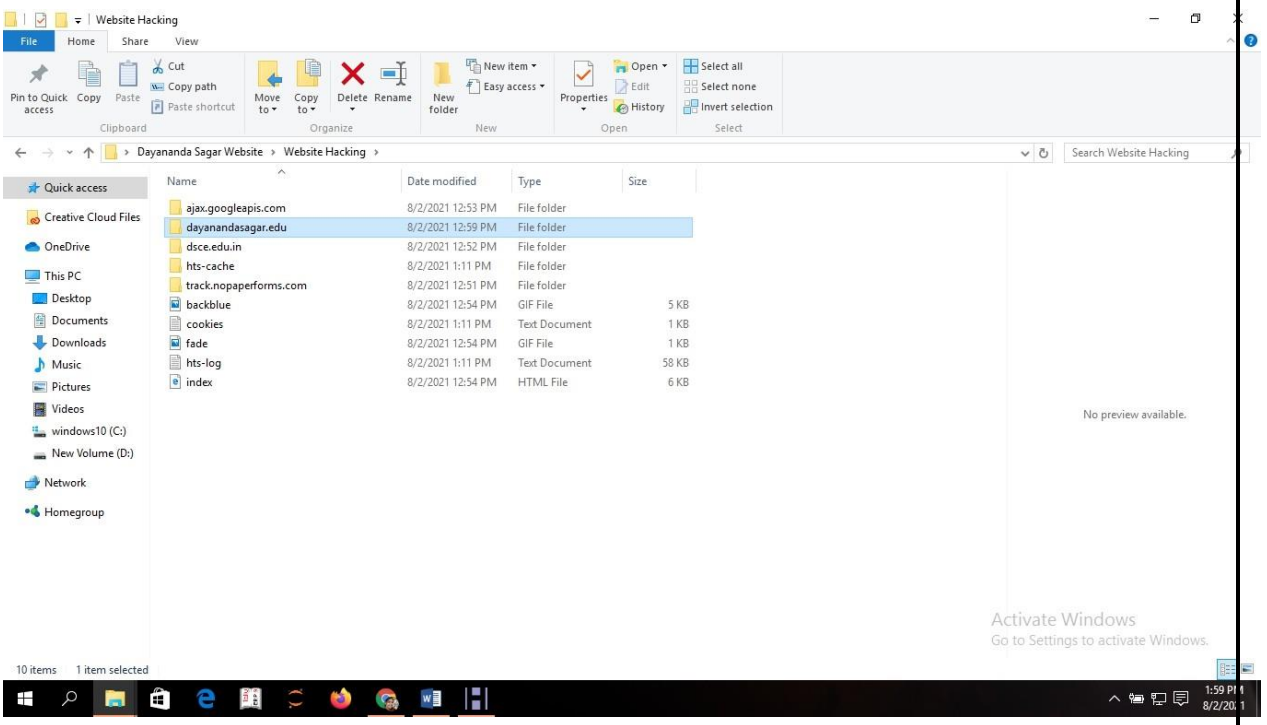
Step 10: mirroring process is get started



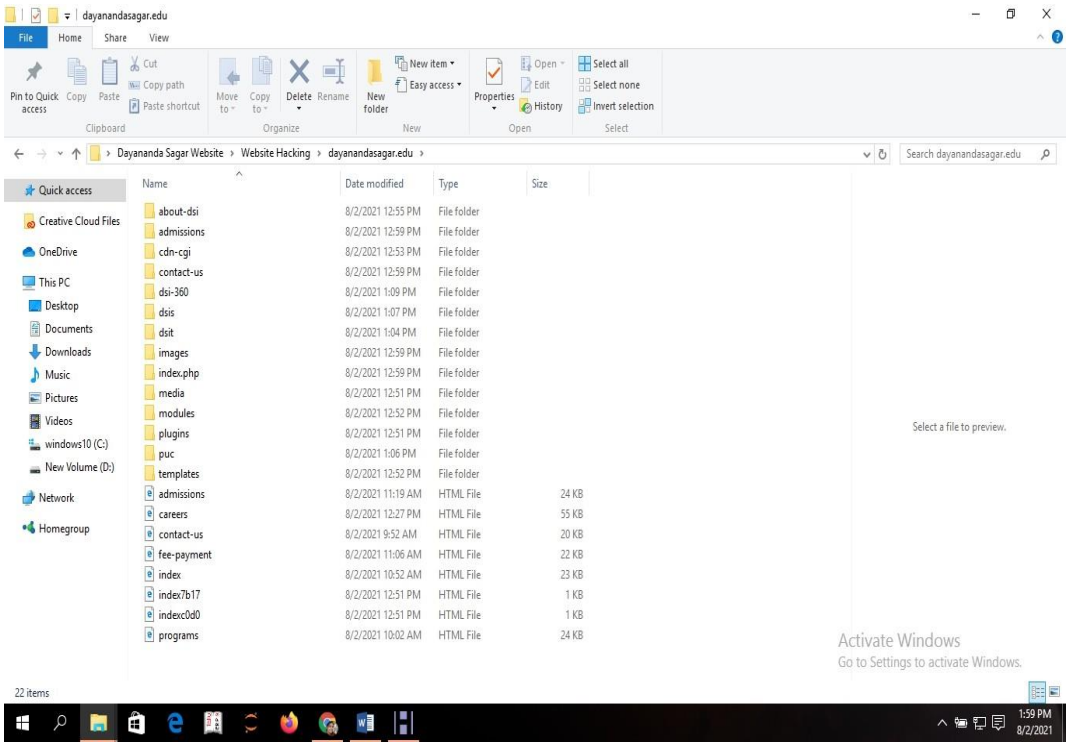
Step 11: The detail information about the URL will be fetched and saved in the folder “Dayananda Sagar Website”. You can now open the folder where you can see the the project name given as Website hacking as shown in Step 3.



Step 12: Click on Website hacking file, then the URL address dayanandasagar.edu file given in the Step 8 will be visible.

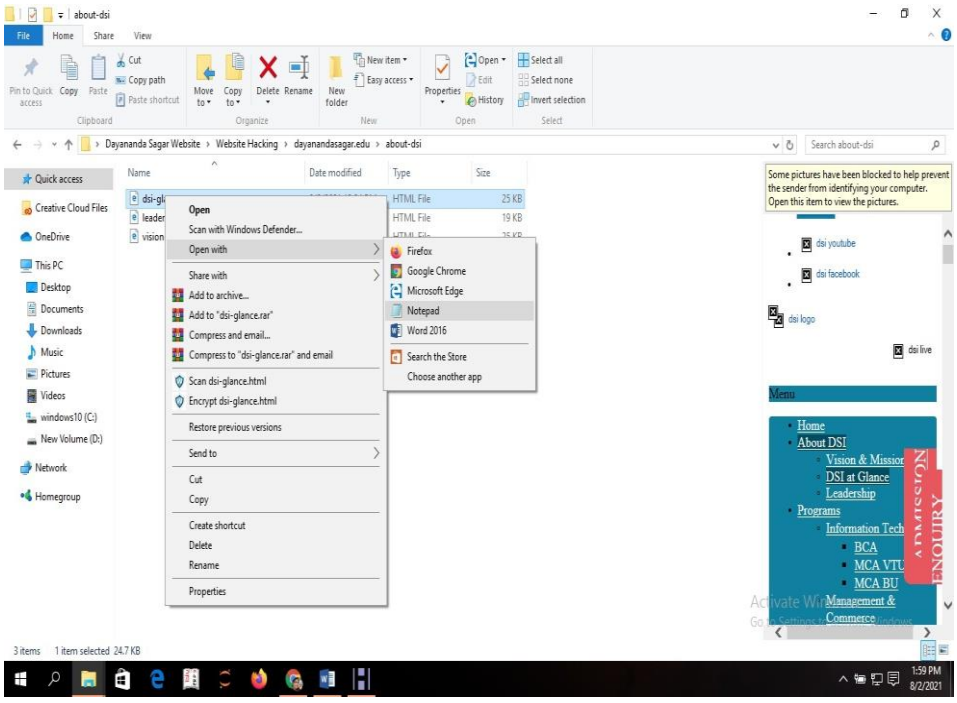


Step 13: Click on the file, dayanandasagar.edu. Now you can find all the files of the original page of the Website.

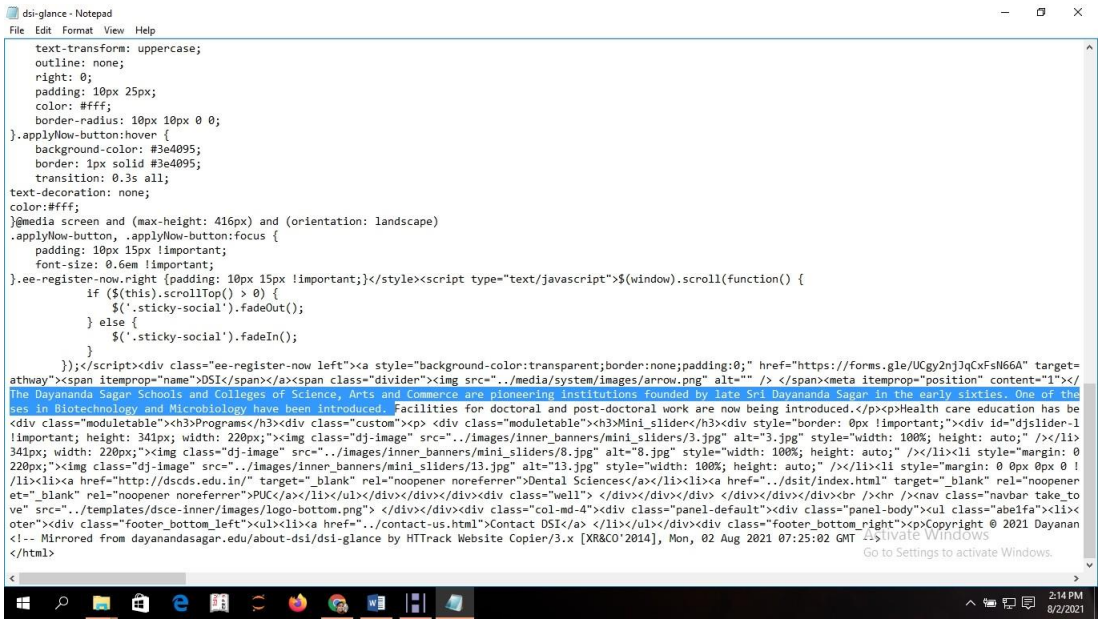


Step 14: Click on any file to alter the content : open with notepad.

Example: Click the file about-dsi. 3 Files are displayed. Any file can be opened in a notepad then changes can be done in the file.



Step 15: Contents of the pages is displayed. Now you can alter the Contents.



RESULT:

The main aim is to downloading a website using website copier tool (HTTtack) is completed successfully

EXPT.NO 3(B)	Implement Caesar Cipher in C Language.	DATE:

AIM:

To implement the simple substitution technique named Caesar cipher using C language.

DESCRIPTION:

To encrypt a message with a Caesar cipher, each letter in the message is changed using a simple rule: shift by three. Each letter is replaced by the letter three letters ahead in the alphabet. A becomes D, B becomes E, and so on. For the last letters, we can think of the alphabet as a circle and "wrap around". W becomes Z, X becomes A, Y beco mes B, and Z

becomes C. To change a message back, each letter is replaced by the one three before it.

ALGORITHM:

STEP-1: Read the plain text from the user.

STEP-2: Read the key value from the user.

STEP-3: If the key is positive then encrypt the text by adding the key with eachcharacter in the plain text.

STEP-4: Else subtract the key from the plain text.

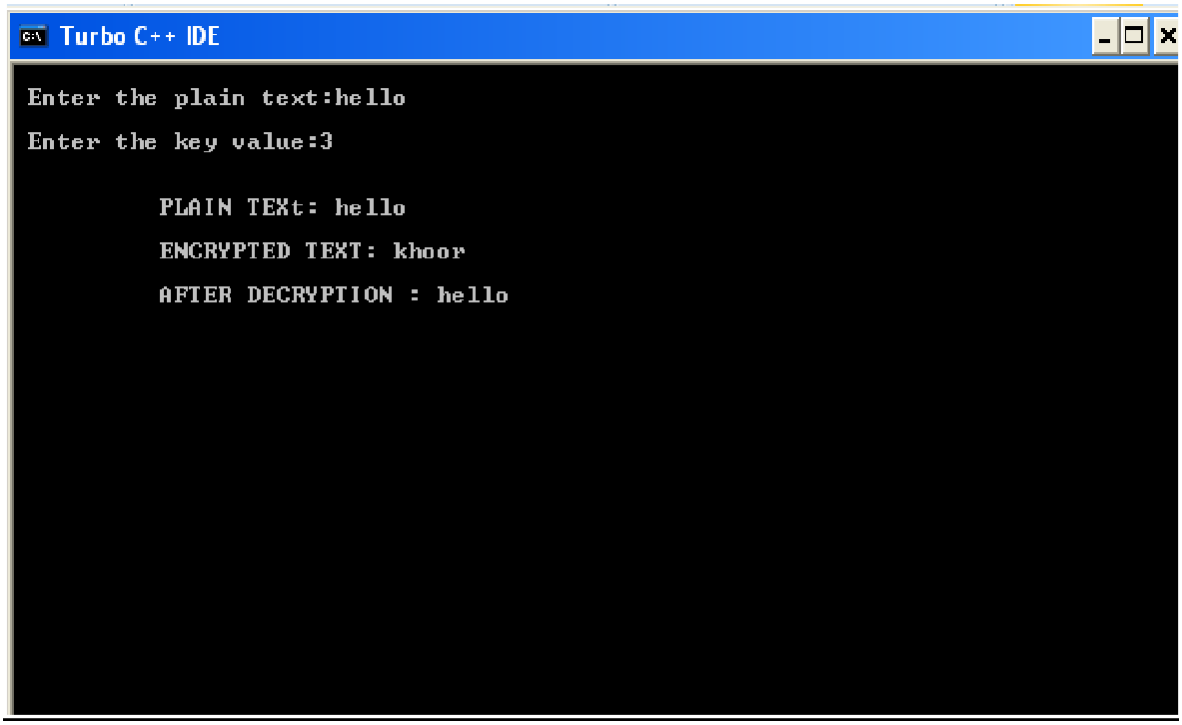
STEP-5: Display the cipher text obtained above.

PROGRAM: (Caesar Cipher)

```
#include <stdio.h>
#include <string.h>
#include<conio.h>
#include <ctype.h>
void main()


{
 char plain[10], cipher[10];
 int key,i,length;
 int result;
 clrscr();
 printf("\n Enter the plain text:");
 scanf("%s", plain);
 printf("\n Enter the key value:");
 scanf("%d", &key);
 printf("\n \n \t PLAIN TEXt: %s",plain);
 printf("\n \n \t ENCRYPTED TEXT: ");
```

```
 for(i = 0, length = strlen(plain); i < length; i++)
      {
      cipher[i]=plain[i] + key;
      if (isupper(plain[i]) && (cipher[i] > 'Z'))
      cipher[i] = cipher[i] - 26;
      if (islower(plain[i]) && (cipher[i] > 'z'))
      cipher[i] = cipher[i] - 26;
      printf("%c", cipher[i]);
      }
      printf("\n \n \t AFTER DECRYPTION : ");
 for(i=0;i<length;i++)
      {
       plain[i]=cipher[i]-key;
      if(isupper(cipher[i])&&(plain[i]<'A'))
      plain[i]=plain[i]+26;
      if(islower(cipher[i])&&(plain[i]<'a'))
      plain[i]=plain[i]+26;
      printf("%c",plain[i]);
 }
getch();
}
```

OUTPUT:



RESULT:

Thus the implementation of Caesar cipher had been executed successfully.

EXPT.NO 4	Implement DES algorithm in Java.	DATE:

AIM:

To write a C program to implement Data Encryption Standard (DES) using CLanguage.

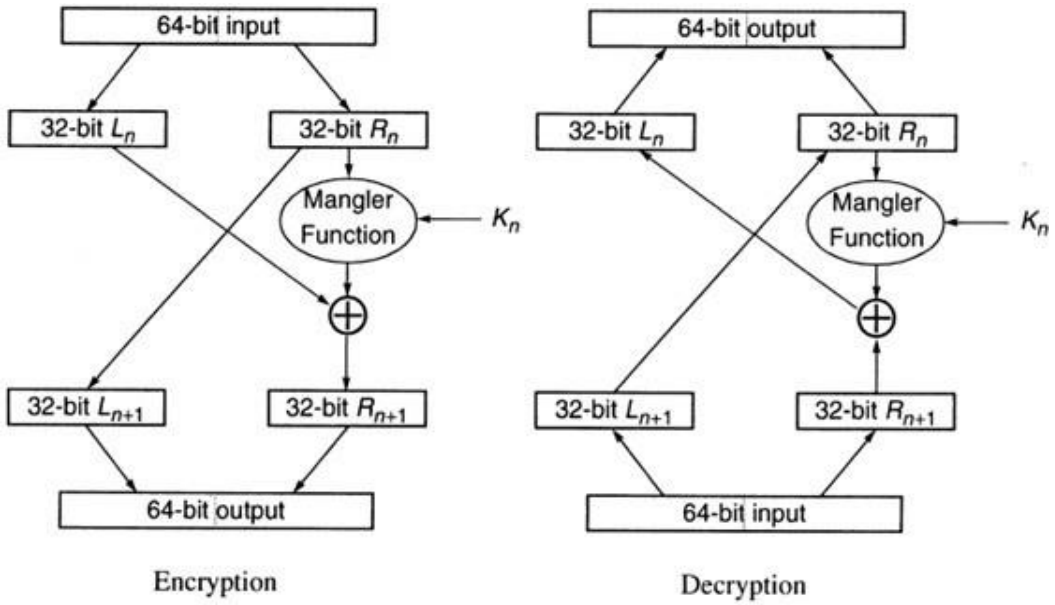
DESCRIPTION:

DES is a symmetric encryption system that uses 64-bit blocks, 8 bits of which areused for parity checks. The key therefore has a "useful" length of 56 bits, which means that only 56 bits are actually used in the algorithm. The algorithm involves carrying out combinations, substitutions and permutations between the text to be encrypted and the key, while making sure the operations can be performed in both directions. The key is ciphered on 64 bits and made of 16 blocks of 4 bits, generally denoted k_1 to k_{16}. Given that "only" 56 bits are actually used for encrypting, there can be 2^{56} different keys.

The main parts of the algorithm are as follows:

- Fractioning of the text into 64-bit blocks
- Initial permutation of blocks
- Breakdown of the blocks into two parts: left and right, named L and R
- Permutation and substitution steps repeated 16 times
- Re-joining of the left and right parts then inverse initial permutation

EXAMPLE:



ALGORITHM:

STEP-1: Read the 64-bit plain text.

STEP-2: Split it into two 32-bit blocks and store it in two different arrays.

STEP-3: Perform XOR operation between these two arrays.

STEP-4: The output obtained is stored as the second 32-bit sequence and the originalsecond 32-bit sequence forms the first part.

STEP-5: Thus the encrypted 64-bit cipher text is obtained in this way. Repeat the sameprocess for the remaining plain text characters.

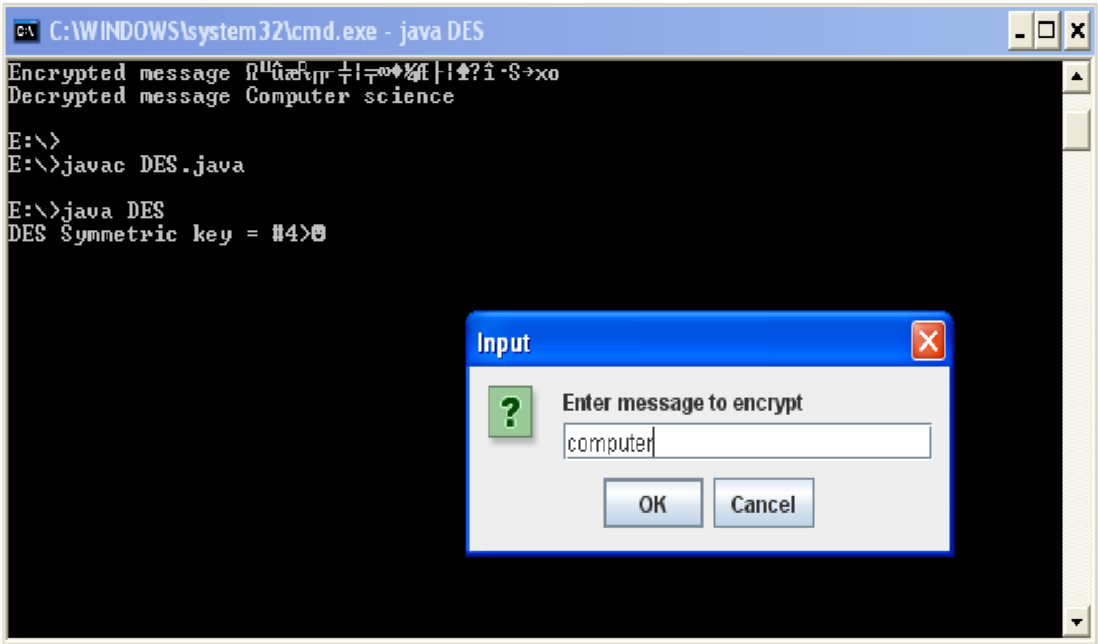
PROGRAM:

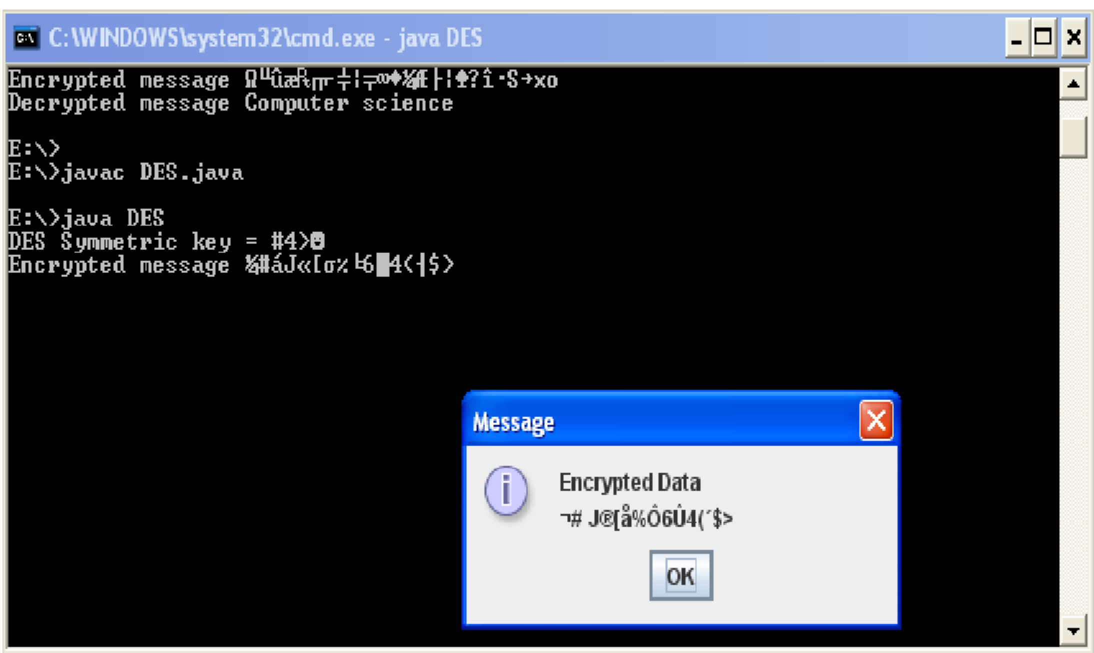
DES.java

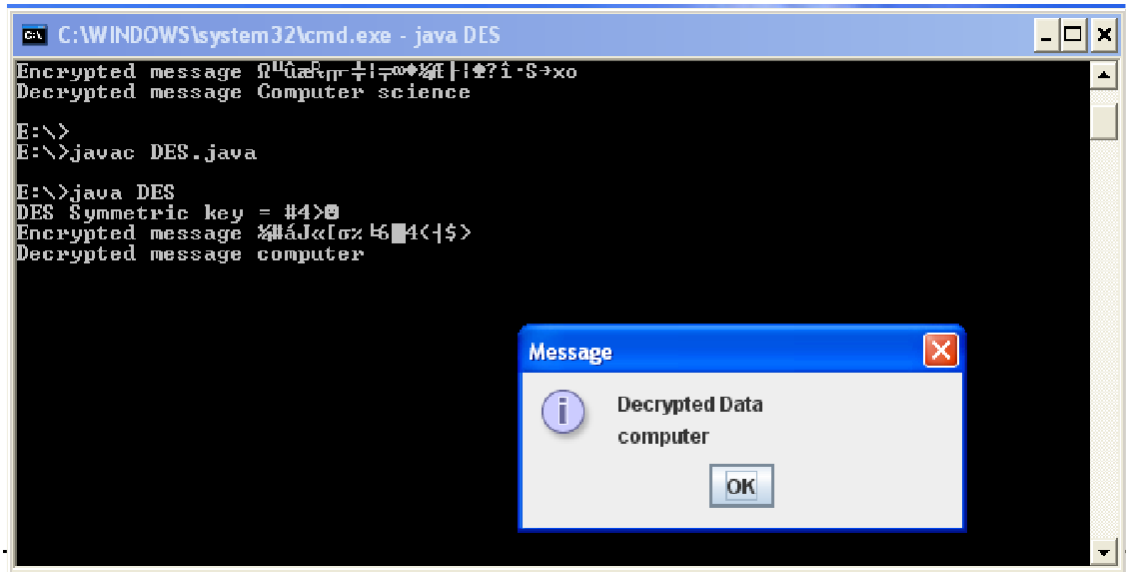
```
import javax.swing.*;
import  java.security.SecureRandom;
import javax.crypto.Cipher;
import  javax.crypto.KeyGenerator;
import javax.crypto.SecretKey;
import  javax.crypto.spec.SecretKeySpec;
import java.util.Random ;
class DES {
     byte[] skey = new byte[1000];
     String skeyString;
     static byte[] raw;
     String inputMessage,encryptedData,decryptedMessage;
public DES()
{
try
{
     generateSymmetricKey();
     inputMessage=JOptionPane.showInputDialog(null,"Enter
     message to encrypt");
     byte[] ibyte = inputMessage.getBytes();
     byte[] ebyte=encrypt(raw, ibyte);
     String encryptedData = new String(ebyte);
     System.out.println("Encrypted message "+encryptedData);
     JOptionPane.showMessageDialog(null,"Encrypted Data
     "+"\n"+encryptedData);
     byte[] dbyte= decrypt(raw,ebyte);
     String decryptedMessage = new String(dbyte);
     System.out.println("Decrypted message
     "+decryptedMessage);
     JOptionPane.showMessageDialog(null,"Decrypted Data
     "+"\n"+decryptedMessage);
}
catch(Exception e)
{
     System.out.println(e);
}
}
```

```
void generateSymmetricKey() {
try {
      Random r = new Random();
      int num = r.nextInt(10000);
      String knum = String.valueOf(num);
      byte[] knumb = knum.getBytes();
      skey=getRawKey(knumb);
      skeyString = new String(skey);
      System.out.println("DES Symmetric key = "+skeyString);
}
catch(Exception e)
{
      System.out.println(e);
}
}
private static byte[] getRawKey(byte[] seed) throws Exception
{
      KeyGenerator kgen = KeyGenerator.getInstance("DES");
      SecureRandom sr = SecureRandom.getInstance("SHA1PRNG");
      sr.setSeed(seed);
      kgen.init(56, sr);
      SecretKey skey = kgen.generateKey();
      raw = skey.getEncoded();
      return raw;
}
private static byte[] encrypt(byte[] raw, byte[] clear) throws
Exception {
            SecretKeySpec skeySpec = new SecretKeySpec(raw,
            "DES");
            Cipher cipher = Cipher.getInstance("DES");
            cipher.init(Cipher.ENCRYPT_MODE, skeySpec);
            byte[] encrypted = cipher.doFinal(clear);
            return encrypted;
}
private static byte[] decrypt(byte[] raw, byte[] encrypted)
throws Exception
{
            SecretKeySpec skeySpec = new SecretKeySpec(raw,
            "DES");
            Cipher cipher = Cipher.getInstance("DES");
            cipher.init(Cipher.DECRYPT_MODE, skeySpec);
            byte[] decrypted = cipher.doFinal(encrypted);
            return decrypted;
}
public static void main(String args[]) {
      DES des = new DES();
}
}
```

OUTPUT:


C:\WINDOWS\system32\cmd.exe - java DES
Encrypted message
Decrypted message Computer science
E:\>
E:\>javac DES.java
E:\>java DES
DES Symmetric key =
Input
Enter message to encrypt
computer
OK
Cancel


C:\WINDOWS\system32\cmd.exe - java DES
Encrypted message
Decrypted message Computer science
E:\>
E:\>javac DES.java
E:\>java DES
DES Symmetric key =
Encrypted message
Message
Encrypted Data
OK



RESULT:

Thus the data encryption standard algorithm had been implemented successfullyusing C language.

EXPT.NO 5(A)	Write a program to illustrate Buffer overflow attack	DATE:

AIM:

The main aim is to write a program to illustrate buffer overflow attack.

PROCEDURE:

A buffer overflow (or buffer overrun) occurs when the volume of data exceeds the storage capacity of the memory buffer..... If the transaction overwrites executable code, it can cause the program to behave unpredictably and generate incorrect results, memory access errors, or crashes.

```
#include <stdio.h>
#include <string.h>

int main(void)

{

   char buff[15];

   int pass = 0;

   printf("\n Enter the password : \n");

   gets(buff);

   if(strcmp(buff, "thegeekstuff"))

   {

      printf ("\n Wrong Password \n");

   }

   else

   {

      printf ("\n Correct Password \n");

      pass = 1;

   }

   if(pass)

   {

     /* Now Give root or admin rights to user*/

      printf ("\n Root privileges given to the user \n");

   }
```

```
    return 0;
}
```

The program above simulates scenario where a program expects a password from user and if the password is correct then it grants root privileges to the user.

Let's the run the program with correct password ie 'thegeekstuff' :

OUTPUT

RUN1

Enter the password :

thegeekstuff

Correct Password

Root privileges given to the user

This works as expected. The passwords match and root privileges are given.But do you know that there is a possibility of buffer overflow in this program. The gets() function does not check the array bounds and can even write string of length greater than the size of the buffer to which the string is written. Now, can you even imagine what can an attacker do with this kind of a loophole?

Here is an example :

RUN 2

Enter the password :

hhhhhhhhhhhhhhhhhhhh

Wrong Password

Root privileges given to the user

RESULT:

The main aim is to write a program to illustrate buffer overflow attack is completed successfully

EXPT.NO 5(B)	Explore Compare It Tool to Compare of two files for Forensic Investigation.	DATE:

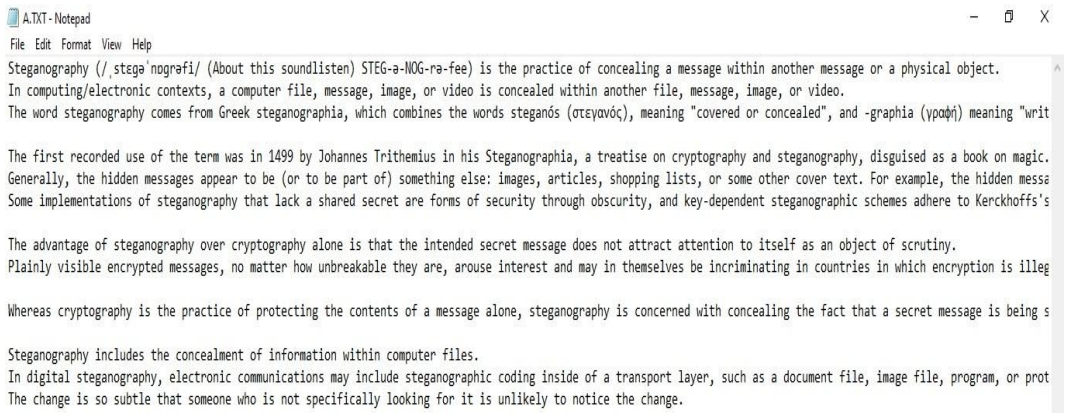
AIM:

The main aim is to comparison of two files for forensics investigation by COMPARE IT tool.

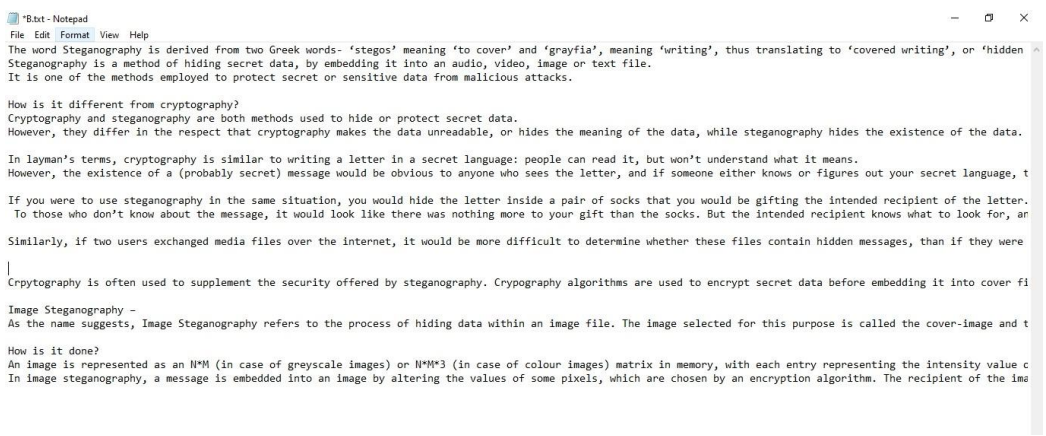
PROCEDURE:

- COMPARE IT is software that displays 2 files side by side, with colored differences sections to simplify analyzing. You can move changes between files with a single mouse click or keystroke, and of course, you have the ability to edit files directly in comparison window.
- It can make colored printout of differences report, exactly as it's on the screen. First of all, install the Compare It from the Link given below. http://www.grigsoft.com/wincmp3.htm it is a 1.7 Mb Software package Click on Compare It Tool, It will show a window to select the files to be compared.
- First, select the first file and click on open and then select the second file and click on open.

STEP 1: open the notepad and create a first text file with the extension .txt and save with a file name

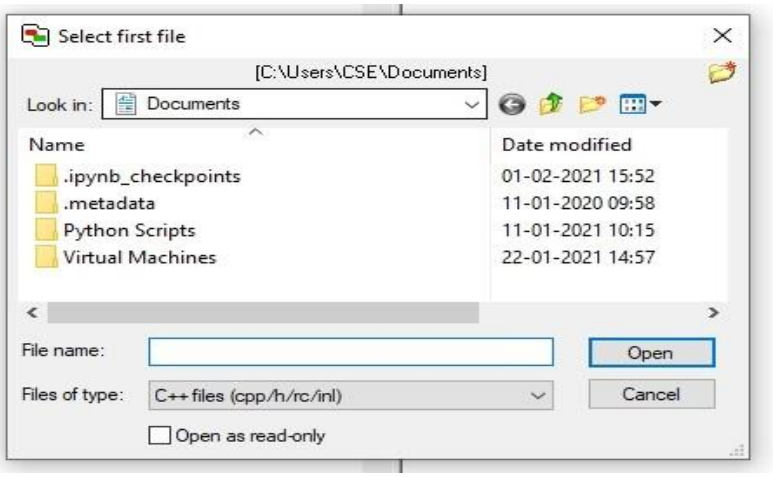


Step 2: create a second text file with the extension .txt

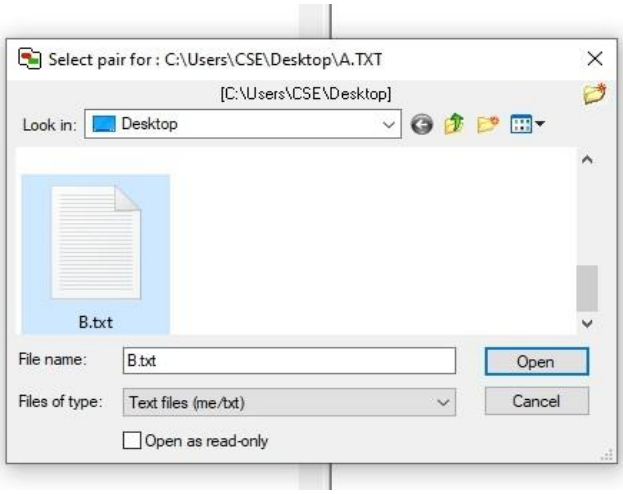


Step 4: Download the compare it tool install the Compare It from the Link given below. http://www.grigsoft.com/wincmp3.htm it is a 1.7 Mb Software package Click on Compare It Tool, It will show a window to select the files to be compared.

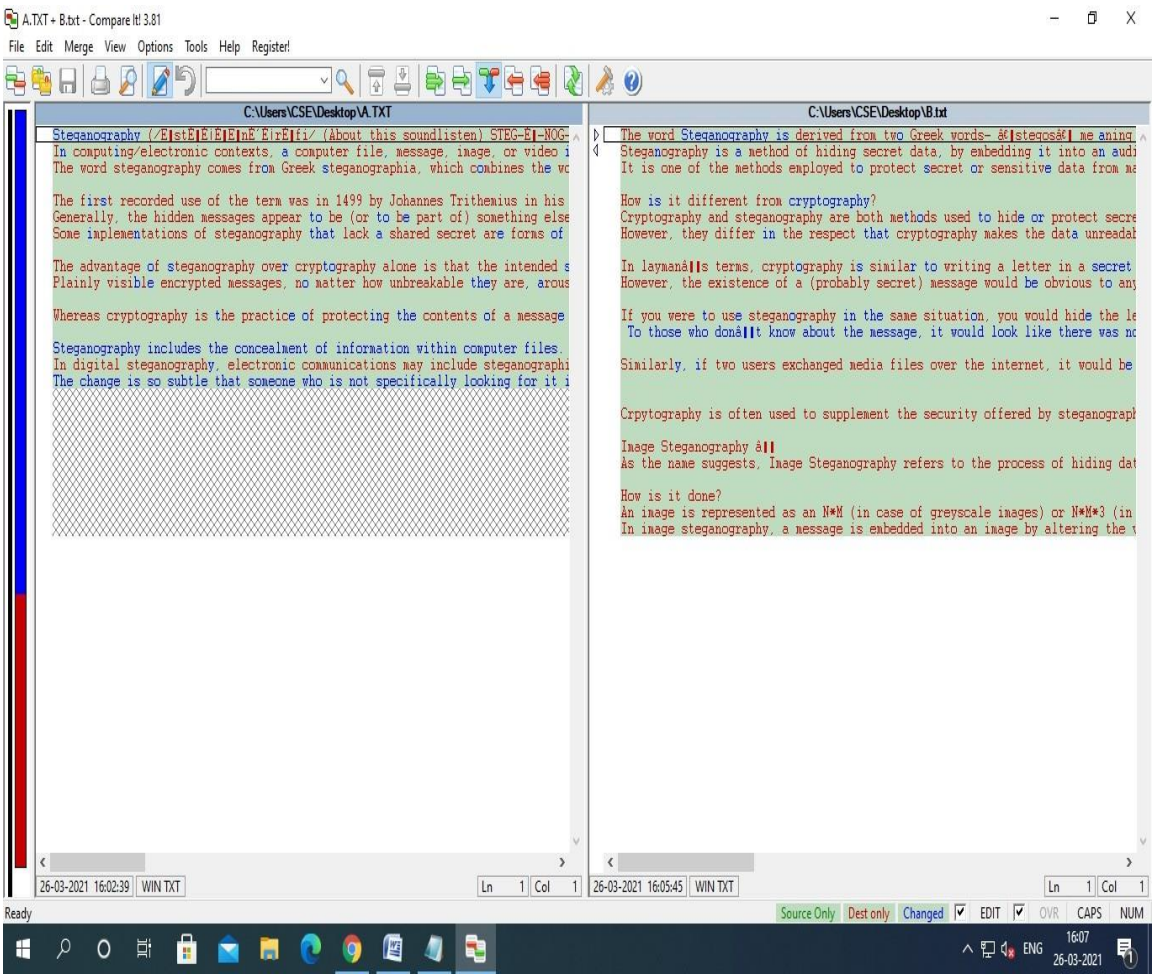
Step 5: Upload the first file to the compare it tool



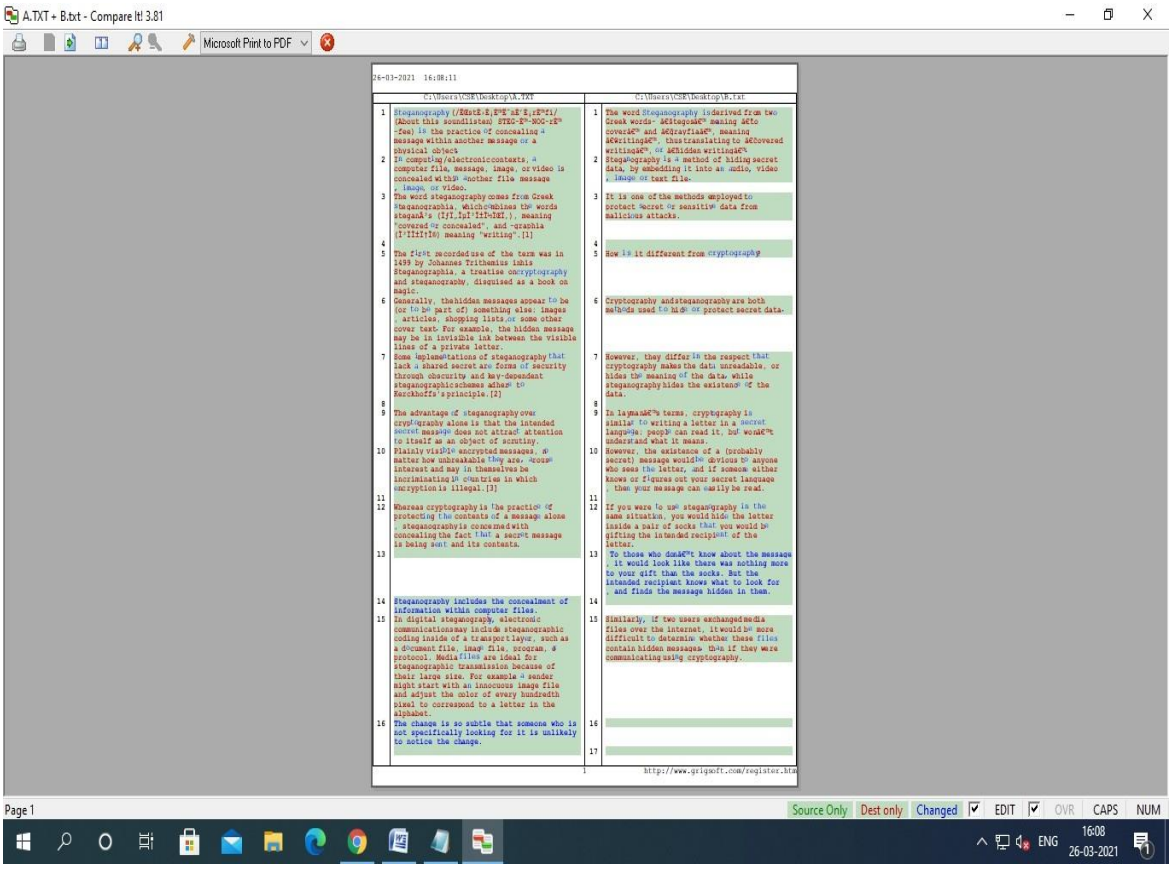
Step 6: upload the second file to the compare it tool



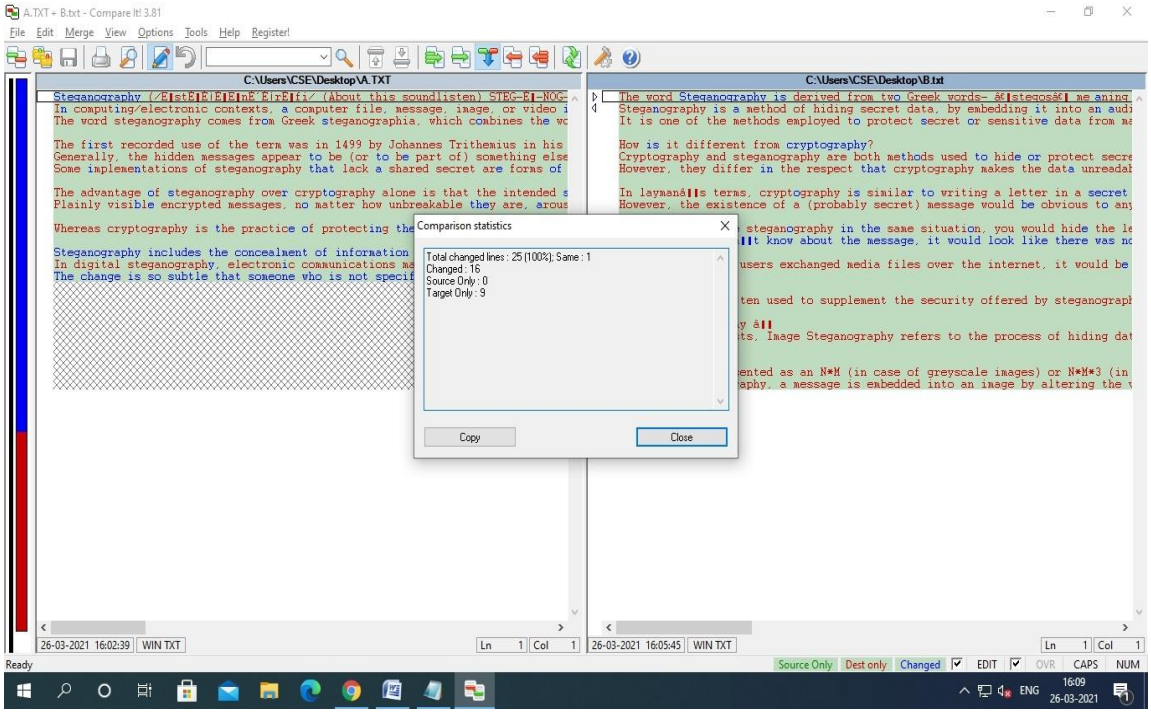
Step 7: Displays 2 files side by side, with colored differences sections to simplify analyzing. You can move changes between files with a single mouse click or keystroke



STEP 8: It also gives you Print report of the difference in the file as follows



STEP9: the comparison result is get display.



EXP NO:6

IMPLEMENTATION OF HILL CIPHER

Aim: To write a C program to implement the hill cipher substitution techniques.

ALGORITHM:

STEP-1: Read the plain text and key from the user.

STEP-2: Split the plain text into groups of length three.

STEP-3: Arrange the keyword in a 3*3 matrix.

STEP-4: Multiply the two matrices to obtain the cipher text of length three.

STEP-5: Combine all these groups to get the complete cipher text.

PROGRAM: (Hill Cipher)

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
int main(){
     unsigned int a[3][3]={{6,24,1},{13,16,10},{20,17,15}};
     unsigned int b[3][3]={{8,5,10},{21,8,21},{21,12,8}};
     int i,j, t=0;
     unsigned int c[20],d[20];
     char msg[20];
     clrscr();
     printf("Enter plain text\n ");
     scanf("%s",msg);
     for(i=0;i<strlen(msg);i++)
     {   c[i]=msg[i]-65;
```

```
            printf("%d ",c[i]);
      }
      for(i=0;i<3;i++)
      {    t=0;
      for(j=0;j<3;j++)
      {
            t=t+(a[i][j]*c[j]);
      }
            d[i]=t%26;
      }
printf("\nEncrypted Cipher Text :");
for(i=0;i<3;i++)
printf(" %c",d[i]+65);
for(i=0;i<3;i++)
{
      t=0;
      for(j=0;j<3;j++)
      {
            t=t+(b[i][j]*d[j]);
      }
      c[i]=t%26;
}
printf("\nDecrypted Cipher Text :");
for(i=0;i<3;i++)
printf(" %c",c[i]+65);
getch();
return 0;
}
```

OUTPUT:

RESULT:

Thus the hill cipher substitution technique had been implemented successfully in C.

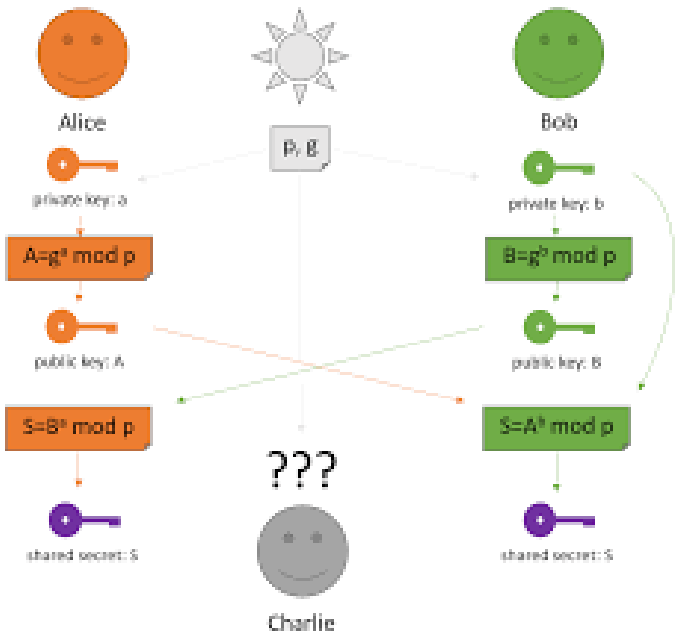
EXPT.NO 7(A)	Implement the Diffie-Hellman Key Exchange algorithm using C language.	DATE:

AIM: To implement the Diffie-Hellman Key Exchange algorithm using C language.

DESCRIPTION:

Diffie–Hellman Key Exchange establishes a shared secret between two parties that can be used for secret communication for exchanging data over a public network. It is primarily used as a method of exchanging cryptography keys for use in symmetric encryption algorithms like AES. The algorithm in itself is very simple. The process begins by having the two parties, Alice and Bob. Let's assume that Alice wants to establish a shared secret with Bob.

EXAMPLE:



ALGORITHM:

STEP-1: Both Alice and Bob shares the same public keys g and p.

STEP-2: Alice selects a random public key a.

STEP-3: Alice computes his secret key A as g^a mod p.

STEP-4: Then Alice sends A to Bob.

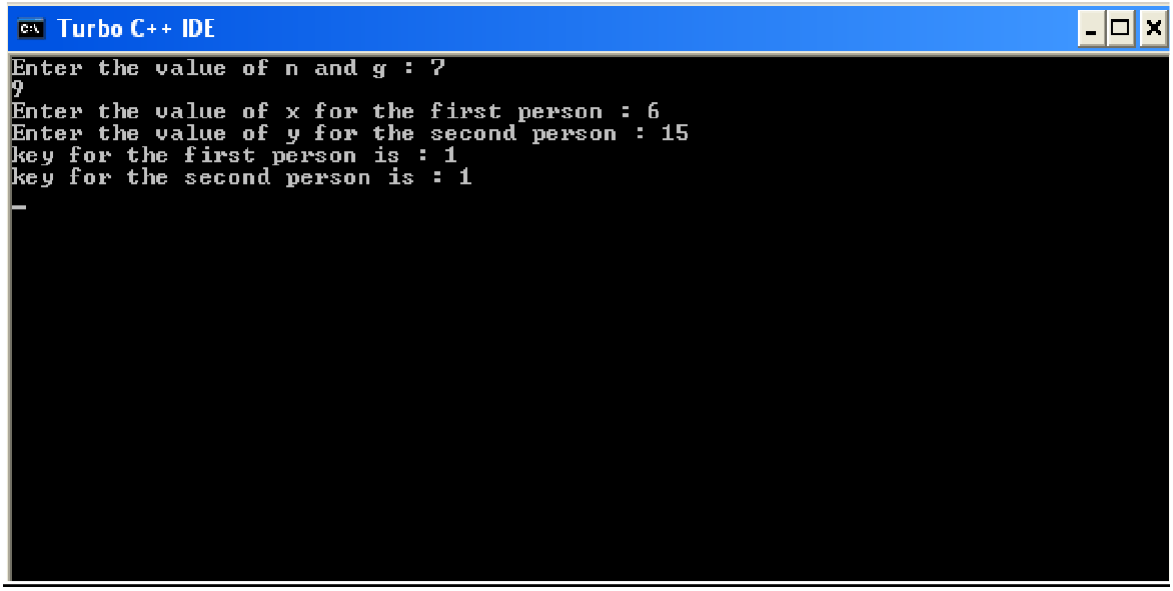
STEP-5: Similarly Bob also selects a public key b and computes his secret key as Band sends the same back to Alice.

STEP-6: Now both of them compute their common secret key as the other one's secretkey power of a mod p.

PROGRAM: (Diffie Hellman Key Exchange)

```
#include<stdio.h>
#include<conio.h>
long long int power(int a, int b, int mod)
{
     long long int t;
     if(b==1)
     return a;
     t=power(a,b/2,mod);
     if(b%2==0)
     return (t*t)%mod;
     else
     return (((t*t)%mod)*a)%mod;
}
long int calculateKey(int a, int x, int n)
{
     return power(a,x,n);
}
void main()
{
     int n,g,x,a,y,b;
     clrscr();
     printf("Enter the value of n and g : ");
     scanf("%d%d",&n,&g);
     printf("Enter the value of x for the first person : ");
     scanf("%d",&x);
     a=power(g,x,n);
     printf("Enter the value of y for the second person : ");
     scanf("%d",&y);
     b=power(g,y,n);
     printf("key for the first person is :
     %lld\n",power(b,x,n));
     printf("key for the second person is :
     %lld\n",power(a,y,n));
     getch();
}
```

OUTPUT:



RESULT:

Thus the Diffie-Hellman key exchange algorithm had been successfully implementedusing C.

EXPT.NO 7(B)	Write the step by step procedure for Hiding and extracting any Text file behind an image file using Command Prompt	DATE

AIM:

The main aim is to hide and extract any text file behind an image file using Command Prompt.

PROCEDURE:

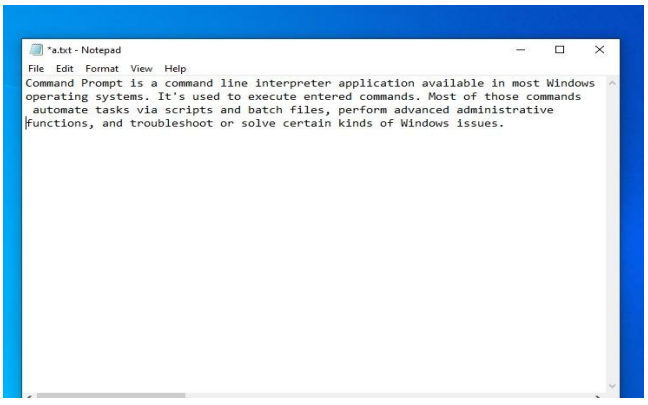
1. Any file like .rar .jpg .txt or any file can be merged inside another file. In a simple way, we shall learn how to hide a text file inside an image file using the Command Prompt.

2. Suppose you have to hide a text file “A.txt” with the image file “B.jpg” and combine them in a new file as “C.jpg”. Where “C.jpg” is our output file which contains the text hidden in the image file**.**

Step1:Create a text document with the file name and .txt as an extension

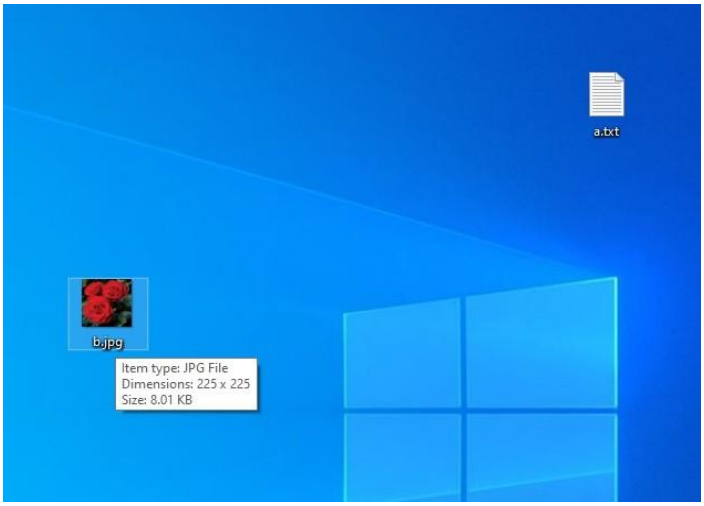
Example: a.txt is created

Step2: Type the content which you need to hide in the image and save it

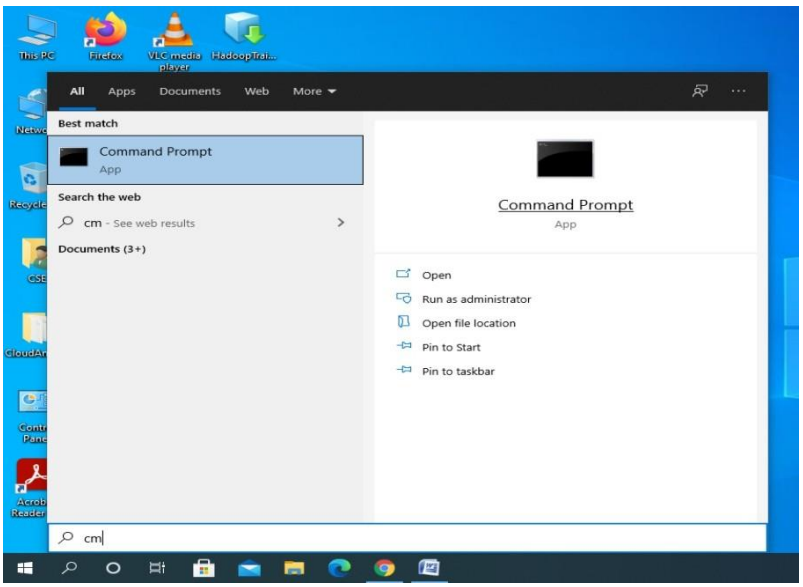


Step 4: Create an image file and save it with the extension .jpg

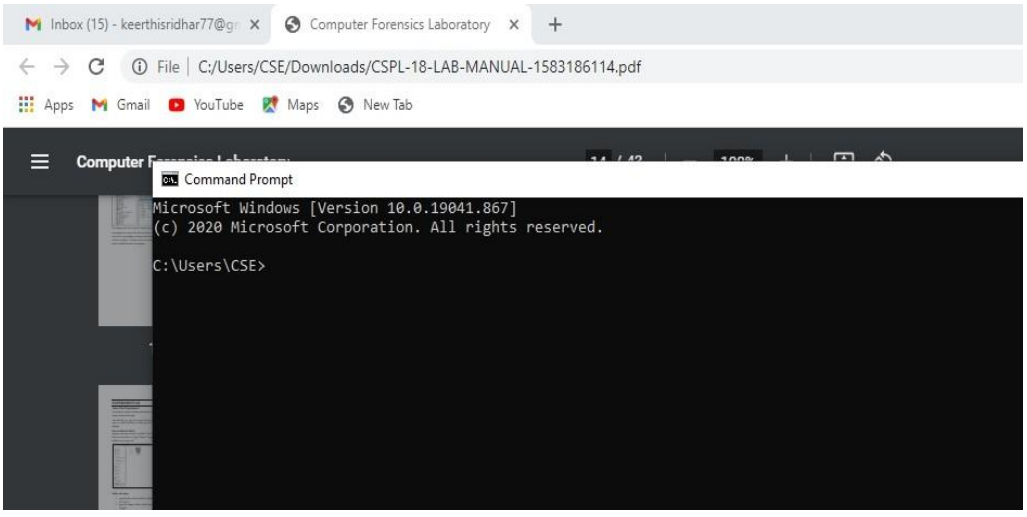
Example: b.jpg is created



Step 5: Open command prompt by selecting start icon in the task bar

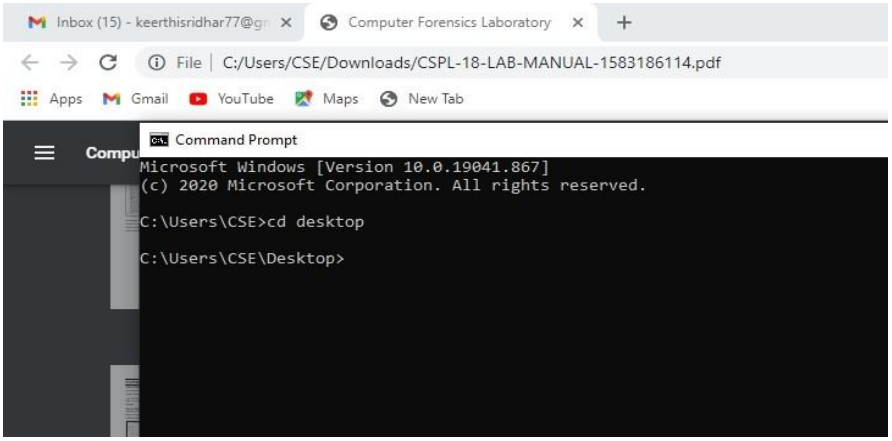


Step 6: Open the command prompt a black working place will be available (or) press ctrl+r and type cmd and hit enter.

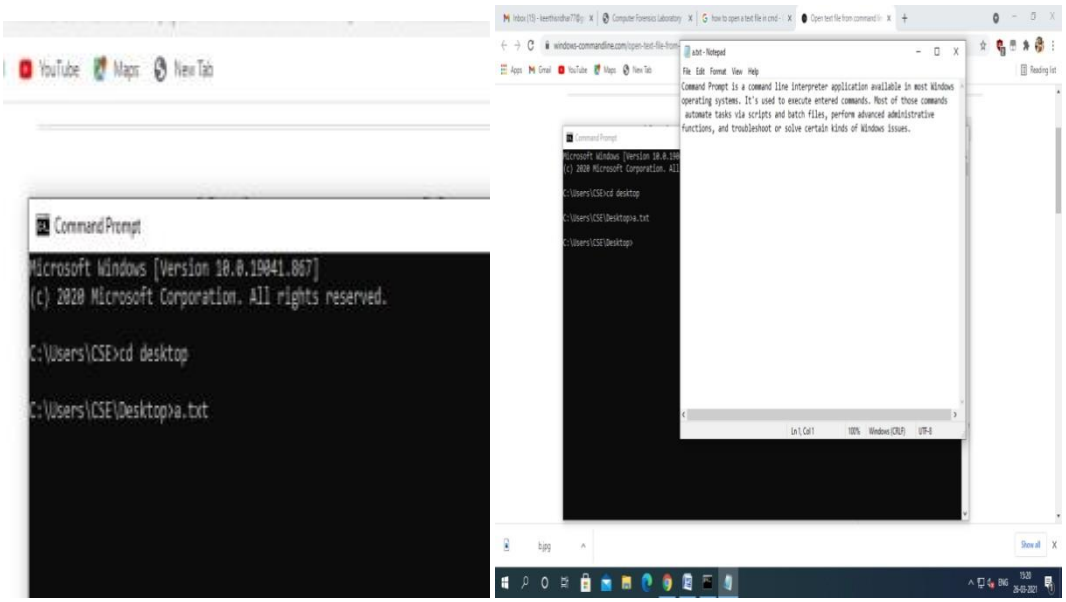


Step 7: Move to the folder where the two are located the CD command is used to enter in to the folder.

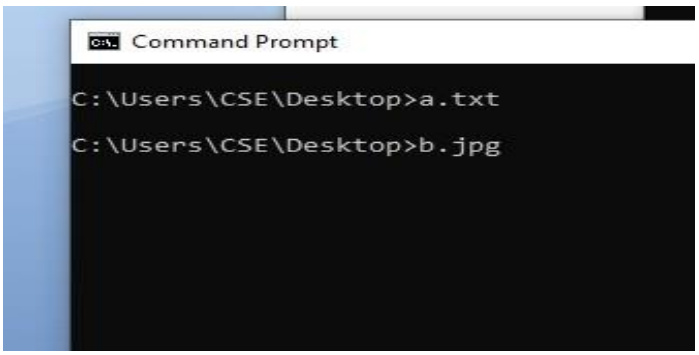
>>cd desktop

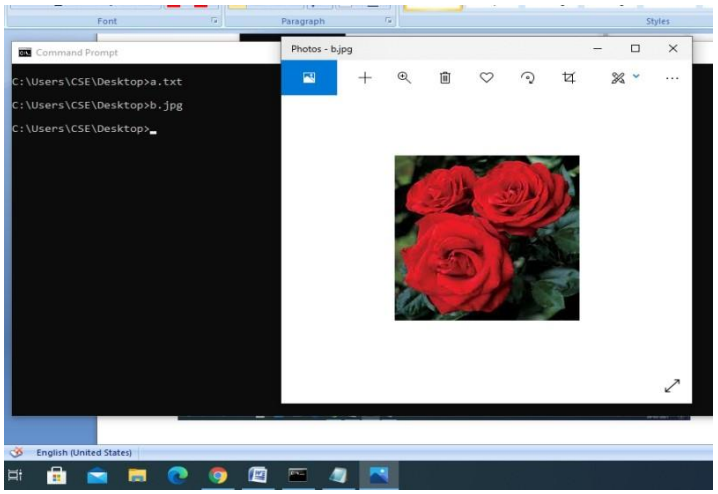


Step 8: Open the text file by its file name Example a.txt then txt file will get open



Step 9: Open the .jpg file by its file name Example b.jpg then the image file will get open

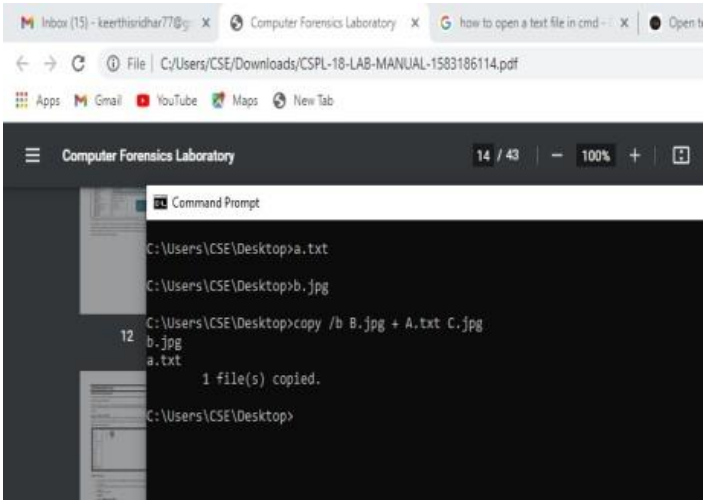




Step 10: Now type the following

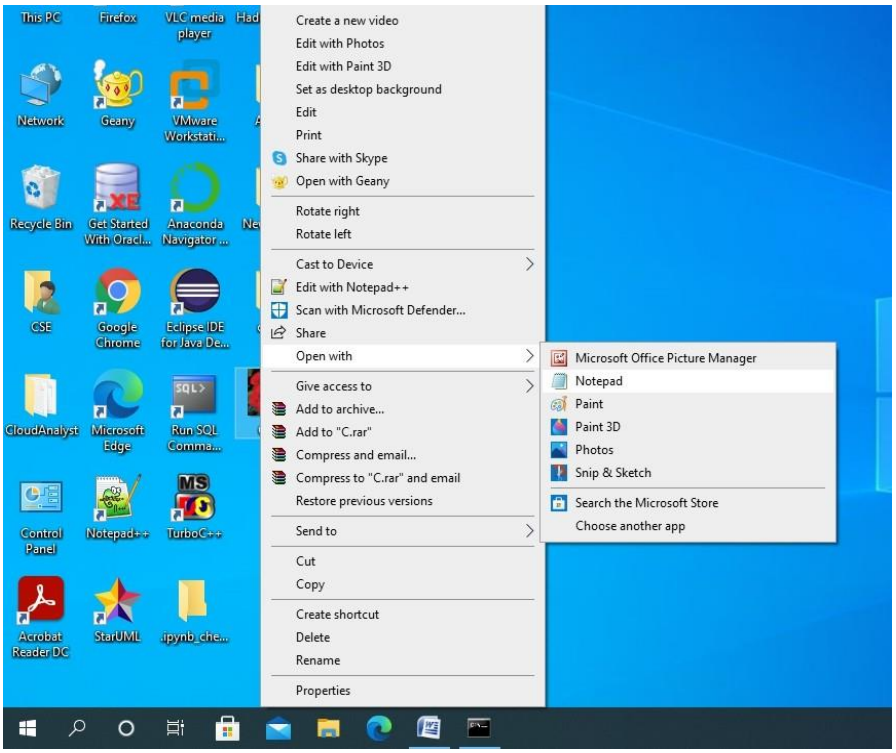
Syntax: copy /b Name-of-file-containing-text-you-want-to-hide.txt + Name-of-initialimage.jpg Resulting-image-name.jpg

Code: > copy /b B.jpg + A.txt C.jpg

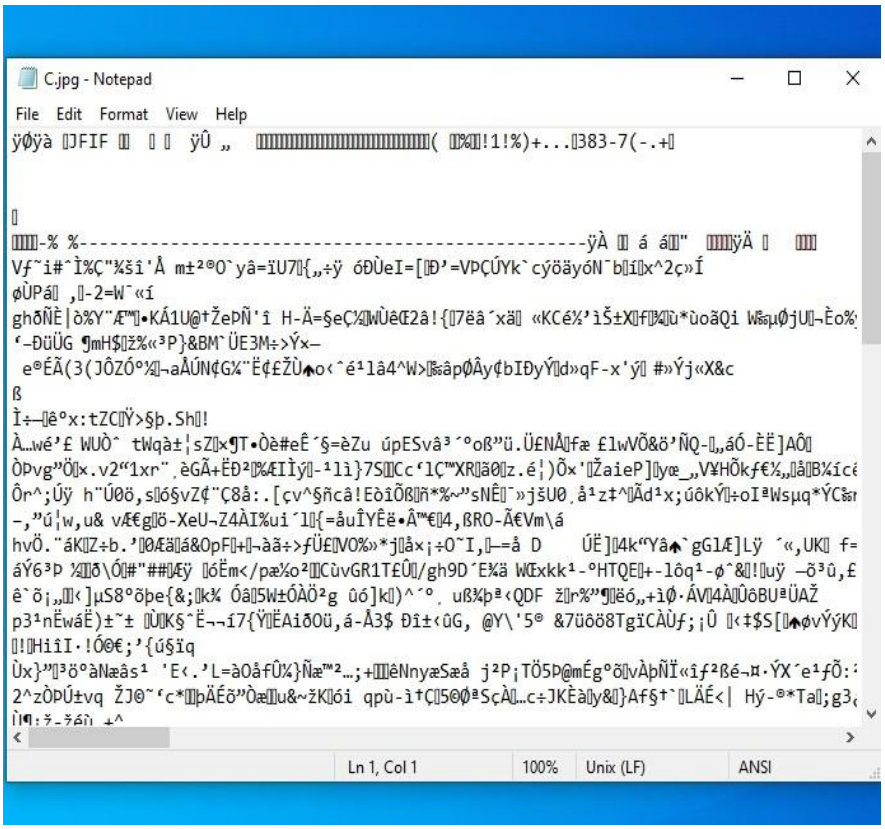


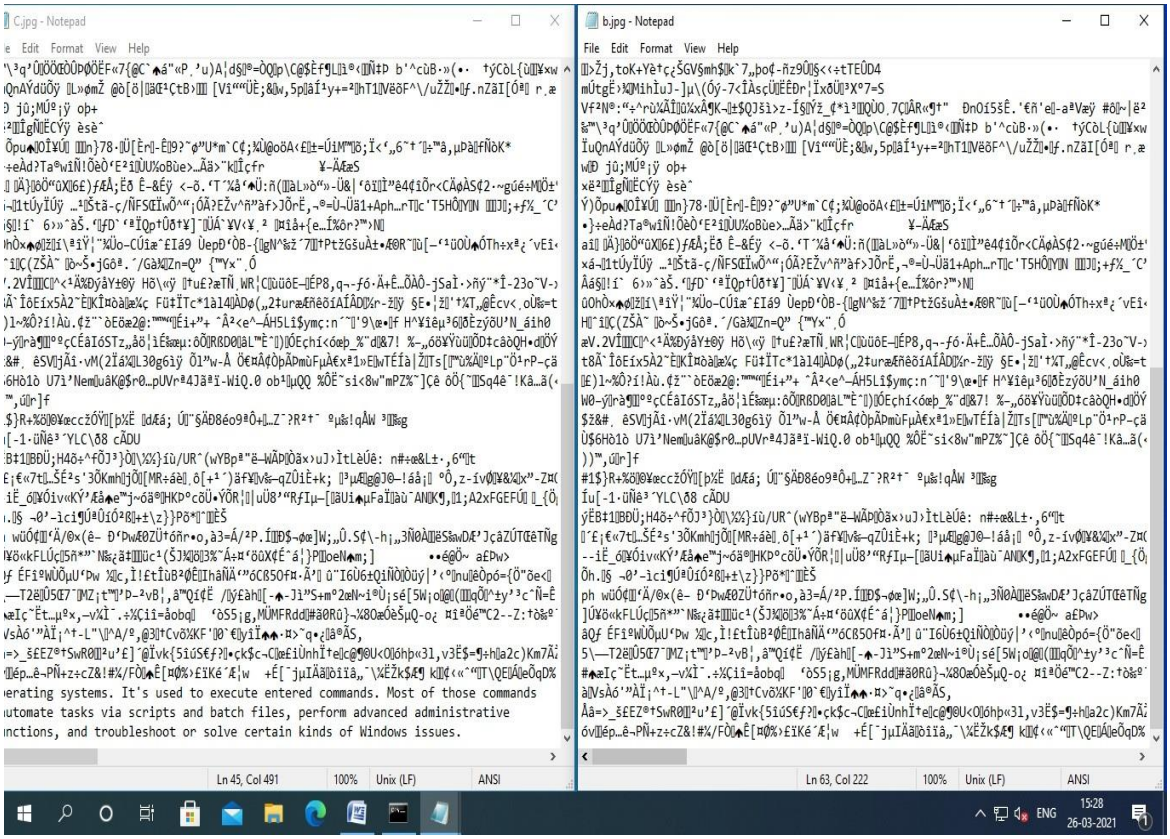
Step 11: locate C.jpg file from where you want to retrieve text data

Step 12: Right-click and open with notepad



Done! Successfully opened! In the last of the notepad, you'll find the content of the text file





RESULT:

The main aim is to hide and extract any text file behind an image file using Command Prompt is completed successfully.

EXPT.NO 8(A)	Setup a honey pot and monitor the honeypot on network (KF Sensor)	DATE:

AIM:

Honey Pot is a device placed on Computer Network specifically designed to capture malicious network traffic. KF Sensor is the tool to setup as honeypot when KF Sensor is running it places a siren icon in the windows system tray in the bottom right of the screen. If there are no alerts then green icon is displayed.

INTRODUCTION:

HONEY POT:

A honeypot is a computer system that is set up to act as a decoy to lure cyber attackers, and to detect, deflect or study attempts to gain unauthorized access to information systems. Generally, it consists of a computer, applications, and data that simulate the behavior of a real system that appears to be part of a network but is actually isolated and closely monitored. All communications with a honeypot are considered hostile, as there's no reason for legitimate users to access a honeypot. Viewing and logging this activity can provide an insight into the level and types of threat a network infrastructure faces while distracting attackers away from assets of real value. Honeypots can be classified based on their deployment (use/action) and based on their level of involvement.

Based on deployment, honeypots may be classified as:

1. Production honeypots
2. Research honeypots

Production honeypots are easy to use, capture only limited information, and are used primarily by companies or corporations. Production honeypots are placed inside the production network with other production servers by an organization to improve their overall state of security. Normally, production honeypots are low-interaction honeypots, which are easier to deploy. They give less information about the attacks or attackers than researchhoneypots.

Research honeypots are run to gather information about the motives and tactics of the Black hat community targeting different networks. These honeypots do not add direct value to a specific organization; instead, they are used to research the threats that organizations face andto learn how to better protect against those threats.

KF SENSOR:

KFSensor is a Windows based honeypot Intrusion Detection System (IDS). It acts as a honeypot to attract and detect hackers and worms by simulating vulnerable system services and trojans. By acting as a decoy server it can divert attacks from critical systems and provide a higher level of information than can be achieved by using firewalls and NIDS alone. KFSensor is a system installed in a network in order to divert and study an attacker's behavior. This is a new technique that is very effective in detecting attacks.

The main feature of KFSensor is that every connection it receives is a suspect hence it results in very few false alerts. At the heart of KFSensor sits a powerful internet daemon service that is built to handle multiple ports and IP addresses. It is written to resist denial of service and buffer overflow attacks. Building on this flexibility KFSensor can respond to connections in a variety of ways, from simple port listening and basic services (such as echo), to complex simulations of standard system services. For the HTTP protocol KFSensor accurately simulates the way Microsoft's web server (IIS) responds to both valid and invalid requests. As well as being able to host a website it also handles complexities such as range requests and client side cache negotiations. This makes it extremely difficult for an attacker to fingerprint, or identify KFSensor as a honeypot.

PROCEDURE:

STEP-1: Download KF Sensor Evaluation Setup File from KF Sensor Website.

STEP-2: Install with License Agreement and appropriate directory path.

STEP-3: Reboot the Computer now. The KF Sensor automatically starts during windowsboot.

STEP-4: Click Next to setup wizard.

STEP-5: Select all port classes to include and Click Next.

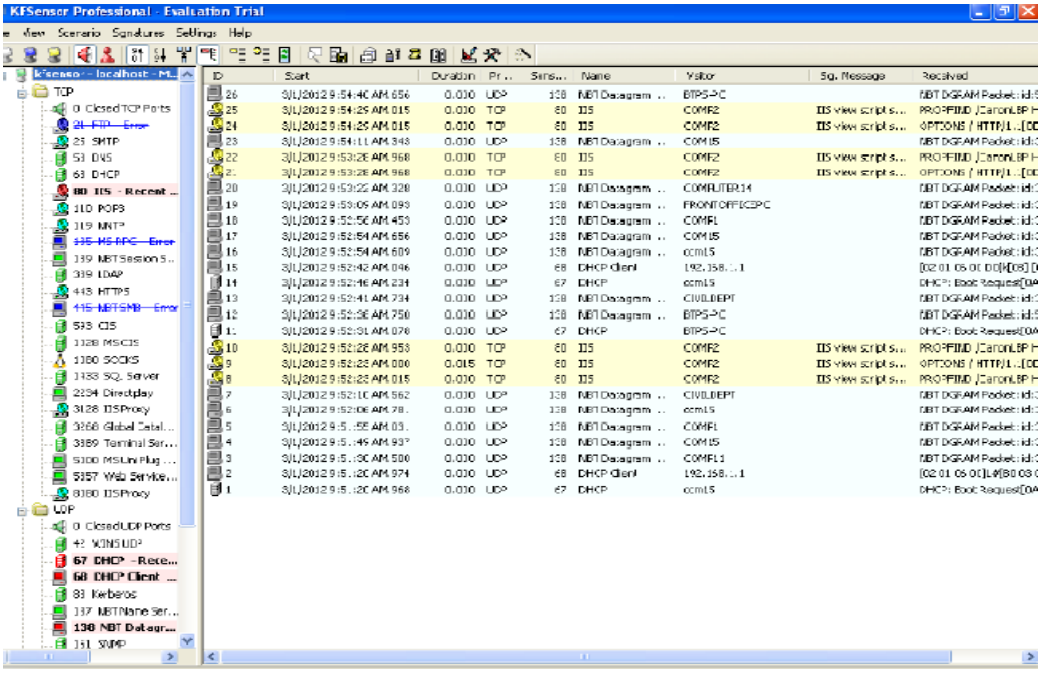
STEP-6: "Send the email and Send from email", enter the ID and Click Next.

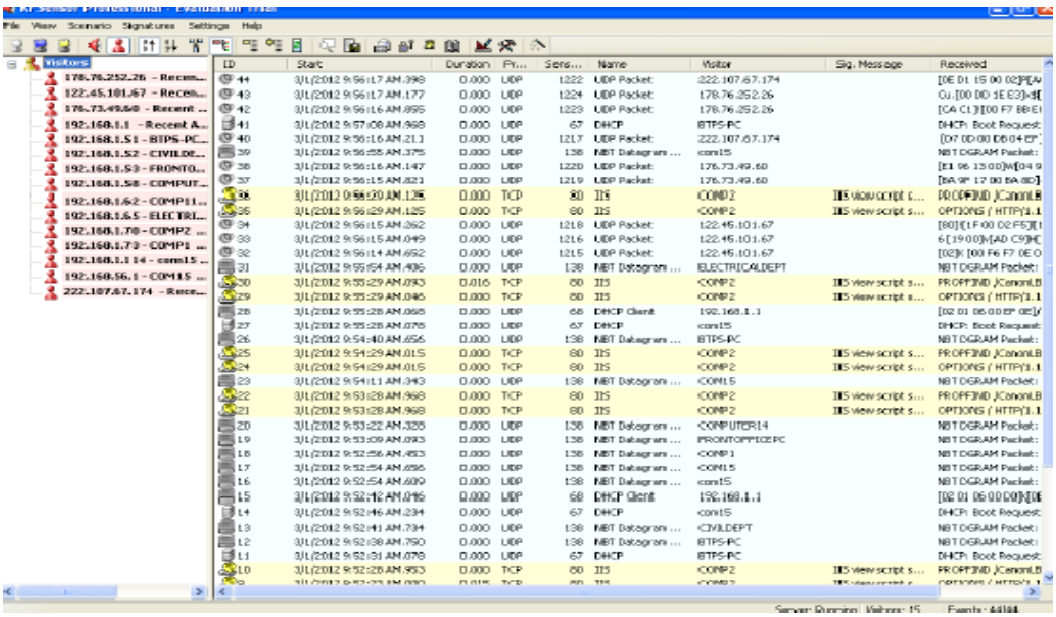
STEP-7: Select the options such as Denial of Service[DOS], Port Activity, Proxy Emulsion,Network Port Analyzer, Click Next.

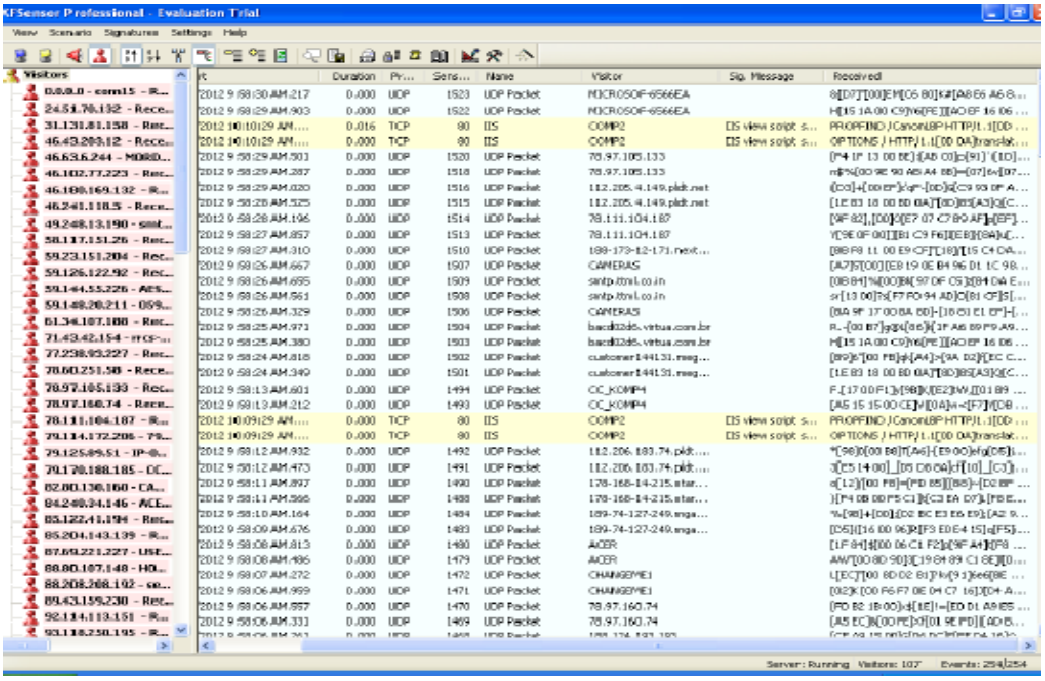
STEP-8: Select Install as System service and Click Next.

STEP-9: Click finish.

SCREENSHOTS:







RESULT:

Thus the study of setup a hotspot and monitor the hotspot on network has been developed successfully.

EXPT.NO 8(B)	Explore the Snow Tool for hiding the information in Text File	DATE:

AIM:

The main aim is to hide the information in the Text File Using SNOW TOOL- Text Stenography

PROCEDURE:

1) Create a text File with some data in the same directory where SNOW Tool is installed.
2) In our Experiment Snow tool is installed in Desktop.

Figure: Text File

3) Go to the Command Prompt, Change the directory to run snow Tool

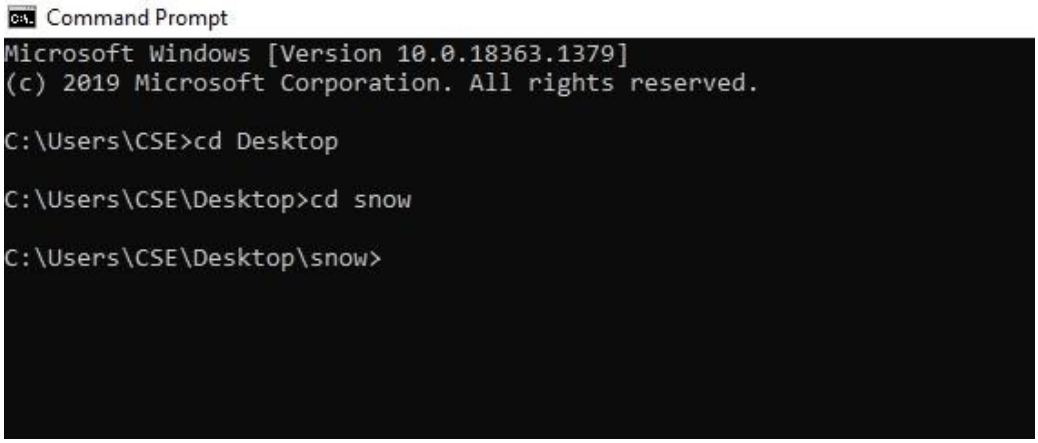


Figure: Changing the Directory

4) Type the Command:

snow -C -m “text to be hidden ” –p “password” <Source File><Destination File>

5) Example:

Snow -C -m “My Account number 1234567” –p “password123” Sample.txt Test.txt

The Source file is a Sample.txt file as shown above. Destination file will be created automatically and exact copy of source file containing hidden information.

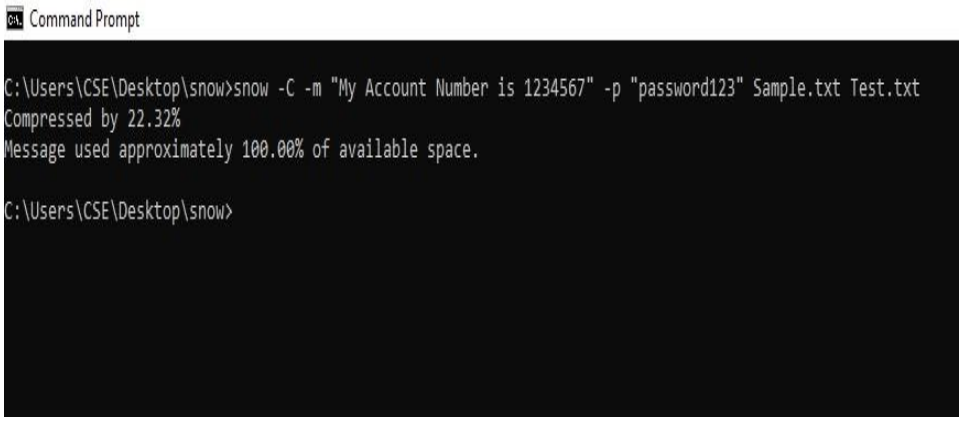


Figure: White Space Steganography using Snow Tool

6) **Go to the Directory:** You will find a new File by name Test.txt. Open the file

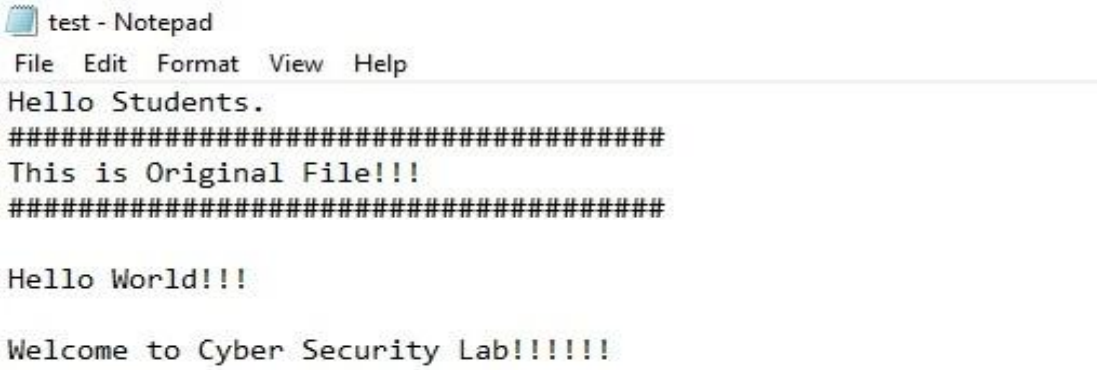


Figure: File Containing Hidden Encrypted Information

7) New file has the same text as an Original file (Sample.txt) without any hidden information. This file can be sent to the target.
8) Recovering the Hidden Information :

   On the Destination, the receiver can reveal information by using the command snow –C –p “password” <Destination File>

   snow –C –p “password123” test.txt

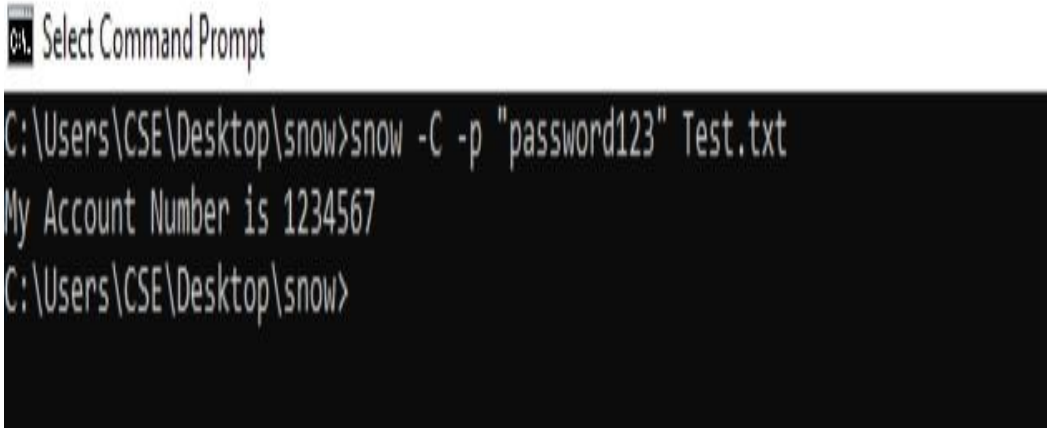


Figure: Decrypting File

As shown in the above figure, file decrypted, showing hidden information encrypted in the previous section

RESULT:

The main aim is to hide the information in the Text File Using SNOW TOOL- Text Stegnography is completed successfully.