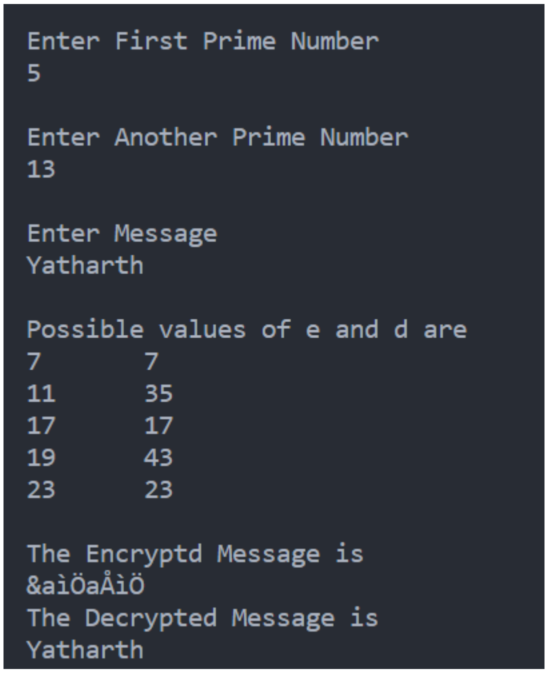
7. RSA algorithm is used by Salim to transfer session key to Anarkali. He suspects that Akbar is performing man in middle attack he chose to use 1024 bit prime numbers. Hint: you may choose to use big integer in java.

Code:

```
#include<iostream>
#include<math.h>
#include<string.h>
#include<stdlib.h>
using namespace std;
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();
int prime(long int pr)
{

  int i;
  j = sqrt(pr);
  for (i = 2; i <= j; i++)
  {

    if (pr % i == 0)
      return 0;
  }
  return 1;
}

int main()

  cout << "\nEnter First Prime Number\n";
  cin >> p;
  flag = prime(p);
  if (flag == 0)
  {
    cout << "\nWrong Input\n";
    exit(1);
  }

  cout << "\nEnter Another Prime Number\n";
  cin >> q;
  flag = prime(q);
  if (flag == 0 || p == q)
  {
    cout << "\nWrong Input\n";
    exit(1);
  }

  cout << "\nEnter Message\n";
  fflush(stdin);
  cin >> msg;
  for (i = 0; msg[i] != '\0'; i++)
    m[i] = msg[i];
  n = p * q;
  t = (p - 1) * (q - 1);
  ce();
  cout << "\nPossible values of e and d are\n";
  for (i = 0; i < j - 1; i++)
    cout << e[i] << "\t" << d[i] << "\n";
  encrypt();
  decrypt();
  return 0;
}
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;
  cout << "\nThe Encryptd 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;
  cout << "\nThe Decrypted Message is\n";
  for (i = 0; m[i] != -1; i++)
    printf("%c", m[i]);
   }
```

Output screenshot:



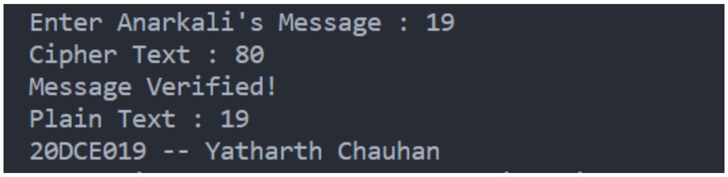
```
Enter First Prime Number
5

Enter Another Prime Number
13

Enter Message
Yatharth

Possible values of e and d are
7       7
11      35
17      17
19      43
23      23

The Encryptd Message is
&aìÖaÅìÖ
The Decrypted Message is
Yatharth
```

8. Salim received some doubtful message from Anarkali. He is doubtful that messages are tempered by some attacker during transmission so he asks Anarkali to digitally sign the messages. Implement system for transferring message with non-repudiation.

Code:

```
e1,d1,n1 = 5,29,91

#assuming the public & private key for anarkali

message_1 = int(input("Enter Anarkali's Message : "))

#encrypt
def encrypt(msg):
#encrypting with 1's private key
    return pow(msg, d1, n1)

#decrypt
def decrypt(cipher):
#decrypting with 1's public key
    return pow(cipher, e1, n1)
cipher = encrypt(message_1)
print(f"Cipher Text : {cipher}")
pt = decrypt(cipher)
if message_1 == pt:
    print("Message Verified!")
print(f"Plain Text : {pt}")
print("20DCE019 -- Yatharth Chauhan")
```

Output screenshot:



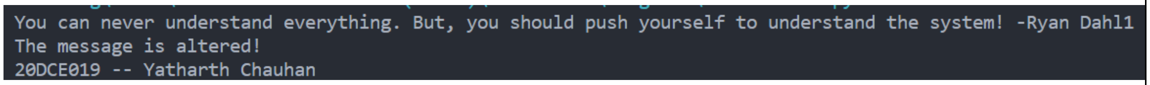
```
Enter Anarkali's Message : 19
Cipher Text : 80
Message Verified!
Plain Text : 19
20DCE019 -- Yatharth Chauhan
```

9. Viru sent important content in file along with hash of file content along with it. At the receivers end , how Raju can check integrity of file? Show that even slight in change in file content will bring significant change in hash value.

Code:

```
import hashlib
import random
# Writing a file
with open("hash.txt", 'w') as f:
    f.write("You can never understand everything. But, you should push yourself
to understand the system! -Ryan Dahl")
with open("hash.txt", 'r') as f:
    msg_1 = f.read()
    hash = hashlib.sha1(msg_1.encode())
    # print(hash.__hash__())
def alter_msg(msg):
    r = random.randint(0, 1)
    msg_1, hash = msg
    if r:
        msg_1 += str(r)  # alter the string by 1 char at random
    print(msg_1)
    return (msg_1, hash)
def check_msg(message):
    x, y = message
    a = hashlib.sha1(str(x).encode()).hexdigest()
    b = y.hexdigest()
    if a == b:
        print("The message is not altered!")
    else:
        print("The message is altered!")
msg_o = (msg_1, hash)  # a tuple of msg along with it's hash
message = alter_msg(msg_o)  # a random function to alter the message
check_msg(message) # to check is msg is altered or not
print("20DCE019 -- Yatharth Chauhan")
```

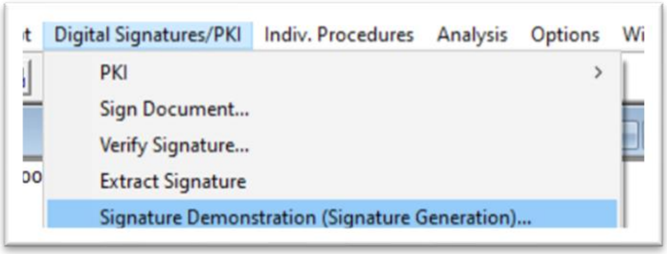
Output Screenshot:

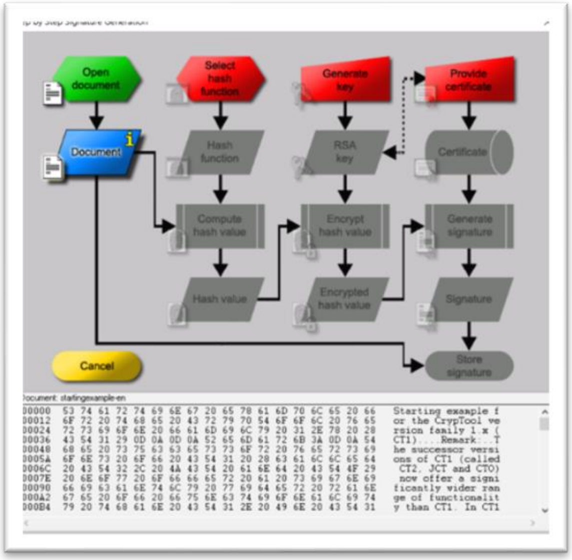


```
You can never understand everything. But, you should push yourself to understand the system! -Ryan Dahl1
The message is altered!
20DCE019 -- Yatharth Chauhan
```

10. Demonstrate working of Digital Signature using Cryptool.

Output Screenshot:

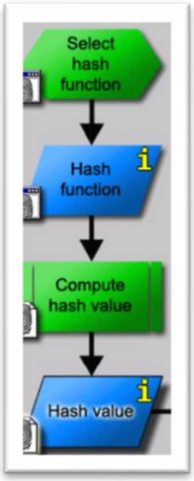




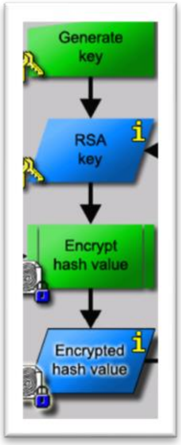
Document: startingexample-en

```
00000  53 74 61 72 74 69 6E 67 20 65 78 61 6D 70 6C 65 20 66   Starting example f
00012  6F 72 20 74 68 65 20 43 72 79 70 54 6F 6F 6C 20 76 65   or the CrypTool ve
00024  72 73 69 6F 6E 20 66 61 6D 69 6C 79 20 31 2E 78 20 28   rsion family 1.x (
00036  43 54 31 29 0D 0A 0D 0A 52 65 6D 61 72 6B 3A 0D 0A 54   CT1)....Remark:..T
00048  68 65 20 73 75 63 63 65 73 73 6F 72 20 76 65 72 73 69   he successor versi
0005A  6F 6E 73 20 6F 66 20 43 54 31 20 28 63 61 6C 6C 65 64   ons of CT1 (called
0006C  20 43 54 32 2C 20 4A 43 54 20 61 6E 64 20 43 54 4F 29    CT2, JCT and CTO)
0007E  20 6E 6F 77 20 6F 66 66 65 72 20 61 20 73 69 67 6E 69    now offer a signi
00090  66 69 63 61 6E 74 6C 79 20 77 69 64 65 72 20 72 61 6E   ficantly wider ran
000A2  67 65 20 6F 66 20 66 75 6E 63 74 69 6F 6E 61 6C 69 74   ge of functionalit
000B4  79 20 74 68 61 6E 20 43 54 31 2E 20 49 6E 20 43 54 31   y than CT1. In CT1
```

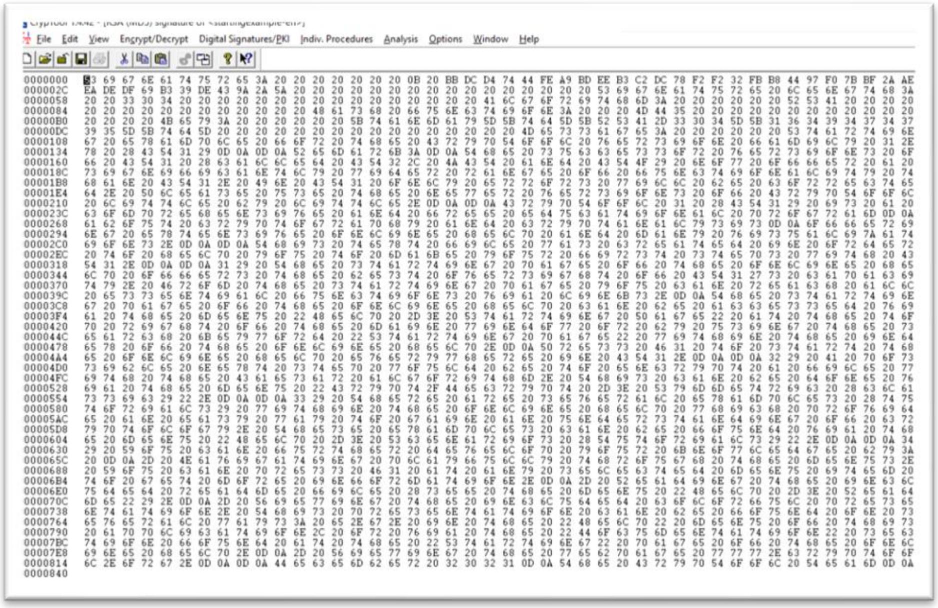
- Now Click on hash function & generate hash



- Now click on generate key & generate key



- Now click on provide certificate, generate certificate & store the signature and thus our required digital signature is generated.


CrypTool 1.4.42 - [RSA (MD5) signature of <startingexample-en>]
File Edit View Encrypt/Decrypt Digital Signatures/PKI Indiv. Procedures Analysis Options Window Help

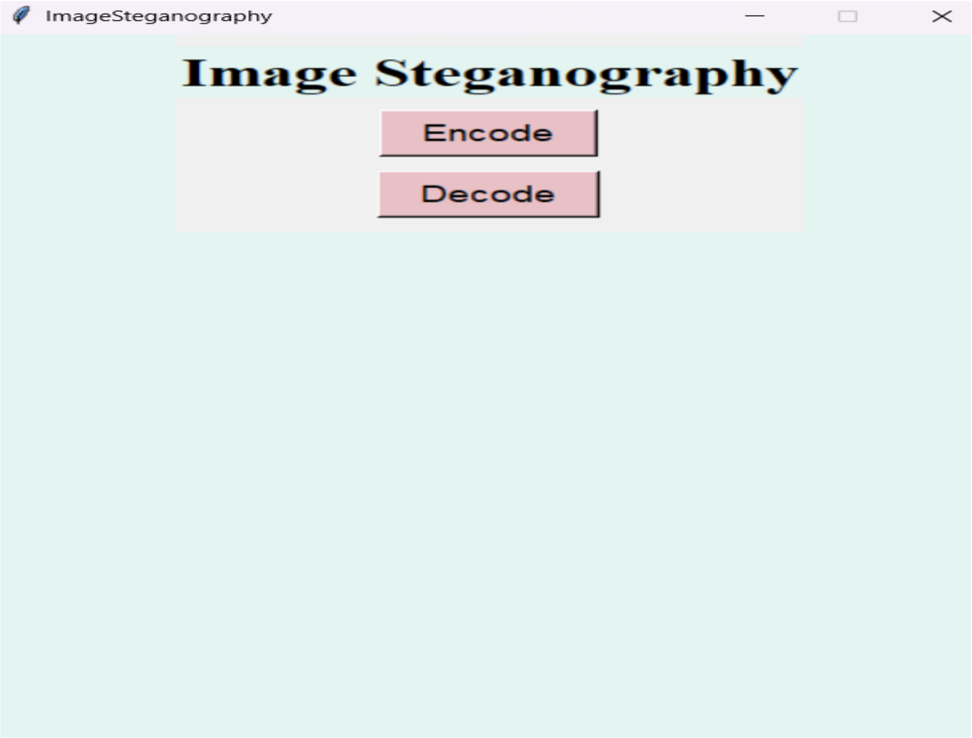
```
0000000  53 69 67 6E 61 74 75 72 65 3A 20 20 20 20 20 20 20 0B 20 BB DC D4 74 44 FE A9 BD EE B3 C2 DC 78 F2 F2 32 FB B8 44 97 F0 7B BF 2A AE
000002C  EA DE DF 69 B3 39 DE 43 9A 2A 5A 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 53 69 67 6E 61 74 75 72 65 20 6C 65 6E 67 74 68 3A
0000058  20 20 33 30 34 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 41 6C 67 6F 72 69 74 68 6D 3A 20 20 20 20 20 20 52 53 41 20 20 20 20
0000084  20 20 20 20 20 20 20 20 20 20 20 20 48 61 73 68 20 66 75 6E 63 74 69 6F 6E 3A 20 20 20 4D 44 35 20 20 20 20 20 20 20 20 20 20 20 20
00000B0  20 20 20 20 4B 65 79 3A 20 20 20 20 20 20 5B 74 61 6E 6D 61 79 5D 5B 74 64 5D 5B 52 53 41 2D 33 30 34 5D 5B 31 36 34 39 34 37 34 37
00000DC  39 35 5D 5B 74 64 5D 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 4D 65 73 73 61 67 65 3A 20 20 20 20 20 20 53 74 61 72 74 69 6E
0000108  67 20 65 78 61 6D 70 6C 65 20 66 6F 72 20 74 68 65 20 43 72 79 70 54 6F 6F 6C 20 76 65 72 73 69 6F 6E 20 66 61 6D 69 6C 79 20 31 2E
0000134  78 20 28 43 54 31 29 0D 0A 0D 0A 52 65 6D 61 72 6B 3A 0D 0A 54 68 65 20 73 75 63 63 65 73 73 6F 72 20 76 65 72 73 69 6F 6E 73 20 6F
0000160  66 20 43 54 31 20 28 63 61 6C 6C 65 64 20 43 54 32 2C 20 4A 43 54 20 61 6E 64 20 43 54 4F 29 20 6E 6F 77 20 6F 66 66 65 72 20 61 20
000018C  73 69 67 6E 69 66 69 63 61 6E 74 6C 79 20 77 69 64 65 72 20 72 61 6E 67 65 20 6F 66 20 66 75 6E 63 74 69 6F 6E 61 6C 69 74 79 20 74
00001B8  68 61 6E 20 43 54 31 2E 20 49 6E 20 43 54 31 20 6F 6E 6C 79 20 65 72 72 6F 72 73 20 77 69 6C 6C 20 62 65 20 63 6F 72 72 65 63 74 65
00001E4  64 2E 20 50 6C 65 61 73 65 20 75 73 65 20 74 68 65 20 6E 65 77 65 72 20 76 65 72 73 69 6F 6E 73 20 6F 66 20 43 72 79 70 54 6F 6F 6C
0000210  20 6C 69 74 74 6C 65 20 62 79 20 6C 69 74 74 6C 65 2E 0D 0A 0D 0A 43 72 79 70 54 6F 6F 6C 20 31 20 28 43 54 31 29 20 69 73 20 61 20
000023C  63 6F 6D 70 72 65 68 65 6E 73 69 76 65 20 61 6E 64 20 66 72 65 65 20 65 64 75 63 61 74 69 6F 6E 61 6C 20 70 72 6F 67 72 61 6D 0D 0A
0000268  61 62 6F 75 74 20 63 72 79 70 74 6F 67 72 61 70 68 79 20 61 6E 64 20 63 72 79 70 74 61 6E 61 6C 79 73 69 73 0D 0A 6F 66 66 65 72 69
0000294  6E 67 20 65 78 74 65 6E 73 69 76 65 20 6F 6E 6C 69 6E 65 20 68 65 6C 70 20 61 6E 64 20 6D 61 6E 79 20 76 69 73 75 61 6C 69 7A 61 74
00002C0  69 6F 6E 73 2E 0D 0A 0D 0A 54 68 69 73 20 74 65 78 74 20 66 69 6C 65 20 77 61 73 20 63 72 65 61 74 65 64 20 69 6E 20 6F 72 64 65 72
00002EC  20 74 6F 20 68 65 6C 70 20 79 6F 75 20 74 6F 20 6D 61 6B 65 20 79 6F 75 72 20 66 69 72 73 74 20 73 74 65 70 73 20 77 69 74 68 20 43
0000318  54 31 2E 0D 0A 0D 0A 31 29 20 54 68 65 20 73 74 61 72 74 69 6E 67 20 70 61 67 65 20 6F 66 20 74 68 65 20 6F 6E 6C 69 6E 65 20 68 65
0000344  6C 70 20 6F 66 66 65 72 73 20 74 68 65 20 62 65 73 74 20 6F 76 65 72 73 69 67 68 74 20 6F 66 20 43 54 31 27 73 20 63 61 70 61 63 69
0000370  74 79 2E 20 46 72 6F 6D 20 74 68 65 20 73 74 61 72 74 69 6E 67 20 70 61 67 65 20 79 6F 75 20 63 61 6E 20 72 65 61 63 68 20 61 6C 6C
000039C  20 65 73 73 65 6E 74 69 61 6C 20 66 75 6E 63 74 69 6F 6E 73 20 76 69 61 20 6C 69 6E 6B 73 2E 0D 0A 54 68 65 20 73 74 61 72 74 69 6E
00003C8  67 20 70 61 67 65 20 6F 66 20 74 68 65 20 6F 6E 6C 69 6E 65 20 68 65 6C 70 20 63 61 6E 20 62 65 20 61 63 63 65 73 73 65 64 20 76 69
00003F4  61 20 74 68 65 20 6D 65 6E 75 20 22 48 65 6C 70 20 2D 3E 20 53 74 61 72 74 69 6E 67 20 50 61 67 65 22 20 61 74 20 74 68 65 20 74 6F
0000420  70 20 72 69 67 68 74 20 6F 66 20 74 68 65 20 6D 61 69 6E 20 77 69 6E 64 6F 77 20 6F 72 20 62 79 20 75 73 69 6E 67 20 74 68 65 20 73
000044C  65 61 72 63 68 20 6B 65 79 77 6F 72 64 20 22 53 74 61 72 74 69 6E 67 20 70 61 67 65 22 20 77 69 74 68 69 6E 20 74 68 65 20 69 6E 64
0000478  65 78 20 6F 66 20 74 68 65 20 6F 6E 6C 69 6E 65 20 68 65 6C 70 2E 0D 0A 50 72 65 73 73 20 46 31 20 74 6F 20 73 74 61 72 74 20 74 68
00004A4  65 20 6F 6E 6C 69 6E 65 20 68 65 6C 70 20 65 76 65 72 79 77 68 65 72 65 20 69 6E 20 43 54 31 2E 0D 0A 0D 0A 32 29 20 41 20 70 6F 73
00004D0  73 69 62 6C 65 20 6E 65 78 74 20 73 74 65 70 20 77 6F 75 6C 64 20 62 65 20 74 6F 20 65 6E 63 72 79 70 74 20 61 20 66 69 6C 65 20 77
00004FC  69 74 68 20 74 68 65 20 43 61 65 73 61 72 20 61 6C 67 6F 72 69 74 68 6D 2E 20 54 68 69 73 20 63 61 6E 20 62 65 20 64 6F 6E 65 20 76
0000528  69 61 20 74 68 65 20 6D 65 6E 75 20 22 43 72 79 70 74 2F 44 65 63 72 79 70 74 20 2D 3E 20 53 79 6D 6D 65 74 72 69 63 20 28 63 6C 61
0000554  73 73 69 63 29 22 2E 0D 0A 0D 0A 33 29 20 54 68 65 72 65 20 61 72 65 20 73 65 76 65 72 61 6C 20 65 78 61 6D 70 6C 65 73 20 28 74 75
0000580  74 6F 72 69 61 6C 73 29 20 77 69 74 68 69 6E 20 74 68 65 20 6F 6E 6C 69 6E 65 20 68 65 6C 70 20 77 68 69 63 68 20 70 72 6F 76 69 64
00005AC  65 20 61 6E 20 65 61 73 79 20 77 61 79 20 74 6F 20 67 61 69 6E 20 61 6E 20 75 6E 64 65 72 73 74 61 6E 64 69 6E 67 20 6F 66 20 63 72
00005D8  79 70 74 6F 6C 6F 67 79 2E 20 54 68 65 73 65 20 65 78 61 6D 70 6C 65 73 20 63 61 6E 20 62 65 20 66 6F 75 6E 64 20 76 69 61 20 74 68
0000604  65 20 6D 65 6E 75 20 22 48 65 6C 70 20 2D 3E 20 53 63 65 6E 61 72 69 6F 73 20 28 54 75 74 6F 72 69 61 6C 73 29 22 2E 0D 0A 0D 0A 34
0000630  29 20 59 6F 75 20 63 61 6E 20 66 75 72 74 68 65 72 20 64 65 76 65 6C 6F 70 20 79 6F 75 72 20 6B 6E 6F 77 6C 65 64 67 65 20 62 79 3A
000065C  20 0D 0A 2D 20 4E 61 76 69 67 61 74 69 6E 67 20 70 6C 61 79 66 75 6C 6C 79 20 74 68 72 6F 75 67 68 20 74 68 65 20 6D 65 6E 75 73 2E
0000688  20 59 6F 75 20 63 61 6E 20 70 72 65 73 73 20 46 31 20 61 74 20 61 6E 79 20 73 65 6C 65 63 74 65 64 20 6D 65 6E 75 20 69 74 65 6D 20
00006B4  74 6F 20 67 65 74 20 6D 6F 72 65 20 69 6E 66 6F 72 6D 61 74 69 6F 6E 2E 0D 0A 2D 20 52 65 61 64 69 6E 67 20 74 68 65 20 69 6E 63 6C
00006E0  75 64 65 64 20 72 65 61 64 6D 65 20 66 69 6C 65 20 28 73 65 65 20 74 68 65 20 6D 65 6E 75 20 22 48 65 6C 70 20 2D 3E 20 52 65 61 64
000070C  6D 65 22 29 2E 0D 0A 2D 20 56 69 65 77 69 6E 67 20 74 68 65 20 69 6E 63 6C 75 64 65 64 20 63 6F 6C 6F 72 66 75 6C 20 70 72 65 73 65
0000738  6E 74 61 74 69 6F 6E 2E 20 54 68 69 73 20 70 72 65 73 65 6E 74 61 74 69 6F 6E 20 63 61 6E 20 62 65 20 66 6F 75 6E 64 20 6F 6E 20 73
0000764  65 76 65 72 61 6C 20 77 61 79 73 3A 20 65 2E 67 2E 20 69 6E 20 74 68 65 20 22 48 65 6C 70 22 20 6D 65 6E 75 20 6F 66 20 74 68 69 73
0000790  20 61 70 70 6C 69 63 61 74 69 6F 6E 2C 20 6F 72 20 76 69 61 20 74 68 65 20 22 44 6F 63 75 6D 65 6E 74 61 74 69 6F 6E 22 20 73 65 63
00007BC  74 69 6F 6E 20 66 6F 75 6E 64 20 61 74 20 74 68 65 20 22 53 74 61 72 74 69 6E 67 22 20 70 61 67 65 20 6F 66 20 74 68 65 20 6F 6E 6C
00007E8  69 6E 65 20 68 65 6C 70 2E 0D 0A 2D 20 56 69 65 77 69 6E 67 20 74 68 65 20 77 65 62 70 61 67 65 20 77 77 77 2E 63 72 79 70 74 6F 6F
0000814  6C 2E 6F 72 67 2E 0D 0A 0D 0A 44 65 63 65 6D 62 65 72 20 32 30 32 31 0D 0A 54 68 65 20 43 72 79 70 54 6F 6F 6C 20 54 65 61 6D 0D 0A
0000840
```

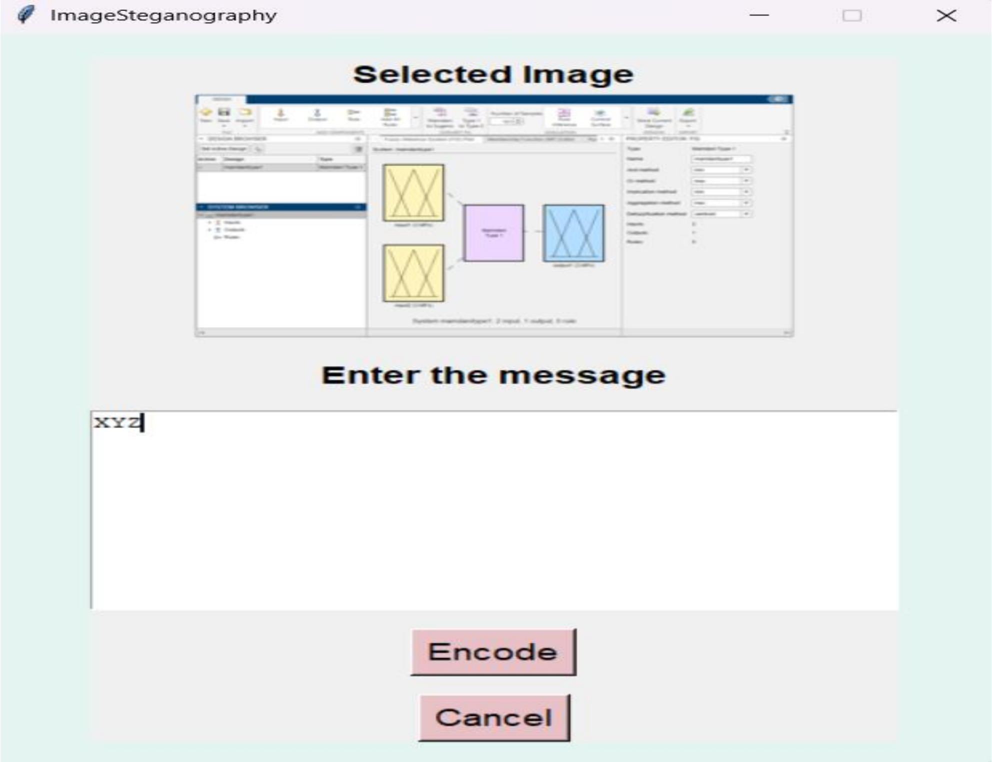
11. Demonstrate image steganography using any programming language of your choice.

Code:

```
 from tkinter import *
 import tkinter.filedialog
 from tkinter import messagebox
 from PIL import ImageTk
 from PIL import Image
 from io import BytesIO
 import  os

    class IMG_Stegno:
    output_image_size = 0

    # main frame or start page

    def main(self, root):
        root.title('ImageSteganography')
        root.geometry('500x600')
        root.resizable(width =False, height=False)
        root.config(bg = '#e3f4f1')
        frame = Frame(root)
        frame.grid()

        title = Label(frame,text='Image Steganography')
        title.config(font=('Times new roman',25, 'bold'))
        title.grid(pady=10)
        title.config(bg = '#e3f4f1')
        title.grid(row=1)

        encode = Button(frame,text="Encode",command= lambda
 :self.encode_frame1(frame), padx=14,bg = '#e3f4f1' )
        encode.config(font=('Helvetica',14), bg='#e8c1c7')
        encode.grid(row=2)
        decode = Button(frame, text="Decode",command=lambda
 :self.decode_frame1(frame), padx=14,bg = '#e3f4f1')
        decode.config(font=('Helvetica',14), bg='#e8c1c7')
        decode.grid(pady = 12)
        decode.grid(row=3)

        root.grid_rowconfigure(1, weight=1)
        root.grid_columnconfigure(0, weight=1)



    # back function to loop back to main screen
    def back(self,frame):
        frame.destroy()
        self.main(root)
    #frame for encode page
    def encode_frame1(self,F):
        F.destroy()
        F2 = Frame(root)
        label1= Label(F2,text='Select the Image in which \nyou want to hide text :')
        label1.config(font=('Times new roman',25, 'bold'),bg = '#e3f4f1')
        label1.grid()

        button_bws = Button(F2,text='Select',command=lambda : self.encode_frame2(F2))
        button_bws.config(font=('Helvetica',18), bg='#e8c1c7')
        button_bws.grid()
        button_back = Button(F2, text='Cancel', command=lambda :
IMG_Stegno.back(self,F2))
        button_back.config(font=('Helvetica',18),bg='#e8c1c7')
        button_back.grid(pady=15)
        button_back.grid()
        F2.grid()

    # frame for decode page
    def decode_frame1(self,F):
        F.destroy()
        d_f2 = Frame(root)
        label1 = Label(d_f2, text='Select Image with Hidden text:')
        label1.config(font=('Times new roman',25,'bold'),bg = '#e3f4f1')
        label1.grid()
        label1.config(bg = '#e3f4f1')
        button_bws = Button(d_f2, text='Select', command=lambda
:self.decode_frame2(d_f2))
        button_bws.config(font=('Helvetica',18), bg='#e8c1c7')
        button_bws.grid()
        button_back = Button(d_f2, text='Cancel', command=lambda :
IMG_Stegno.back(self,d_f2))
        button_back.config(font=('Helvetica',18), bg='#e8c1c7')
        button_back.grid(pady=15)
        button_back.grid()
        d_f2.grid()


    # function to encode image
    def encode_frame2(self,e_F2):
        e_pg= Frame(root)
        myfile = tkinter.filedialog.askopenfilename(filetypes = ([('png', '*.png'),('jpeg',
'*.jpeg'),('jpg', '*.jpg'),('All Files', '*.*')]))
        if not myfile:
            messagebox.showerror("Error","You have selected nothing !")
        else:
            my_img = Image.open(myfile)
            new_image = my_img.resize((300,200))
            img = ImageTk.PhotoImage(new_image)
            label3= Label(e_pg,text='Selected Image')
            label3.config(font=('Helvetica',14,'bold'))
            label3.grid()
            board = Label(e_pg, image=img)
            board.image = img
            self.output_image_size = os.stat(myfile)
            self.o_image_w, self.o_image_h = my_img.size
            board.grid()
            label2 = Label(e_pg, text='Enter the message')
            label2.config(font=('Helvetica',14,'bold'))
            label2.grid(pady=15)
            text_a = Text(e_pg, width=50, height=10)
            text_a.grid()
            encode_button = Button(e_pg, text='Cancel', command=lambda :
IMG_Stegno.back(self,e_pg))
            encode_button.config(font=('Helvetica',14), bg='#e8c1c7')
            data = text_a.get("1.0", "end-1c")
            button_back = Button(e_pg, text='Encode', command=lambda :
[self.enc_fun(text_a,my_img),IMG_Stegno.back(self,e_pg)])
            button_back.config(font=('Helvetica',14), bg='#e8c1c7')
            button_back.grid(pady=15)
            encode_button.grid()
            e_pg.grid(row=1)
            e_F2.destroy()





    # function to decode image
    def decode_frame2(self,d_F2):
        d_F3 = Frame(root)
    myfiles = tkinter.filedialog.askopenfilename(filetypes = ([('png', '*.png'),('jpeg',
    '*.jpeg'),('jpg', '*.jpg'),('All Files', '*.*')]))
        if not myfiles:
            messagebox.showerror("Error","You have selected nothing! ")
        else:
            my_img = Image.open(myfiles, 'r')
            my_image = my_img.resize((300, 200))
            img = ImageTk.PhotoImage(my_image)
            label4= Label(d_F3,text='Selected Image :')
            label4.config(font=('Helvetica',14,'bold'))
            label4.grid()
            board = Label(d_F3, image=img)
            board.image = img
            board.grid()
            hidden_data = self.decode(my_img)
            label2 = Label(d_F3, text='Hidden data is :')
            label2.config(font=('Helvetica',14,'bold'))
            label2.grid(pady=10)
            text_a = Text(d_F3, width=50, height=10)
            text_a.insert(INSERT, hidden_data)
            text_a.configure(state='disabled')
            text_a.grid()
            button_back = Button(d_F3, text='Cancel', command= lambda
:self.frame_3(d_F3))
            button_back.config(font=('Helvetica',14),bg='#e8c1c7')
            button_back.grid(pady=15)
            button_back.grid()
            d_F3.grid(row=1)
            d_F2.destroy()



    # function to decode data
    def decode(self, image):
        image_data = iter(image.getdata())
        data = ''

        while (True):
            pixels = [value for value in image_data.__next__()[:3] +
                      image_data.__next__()[:3] +
                      image_data.__next__()[:3]]
            binary_str = ''
            for i in pixels[:8]:
                if i % 2 == 0:
                    binary_str += '0'
                else:
                    binary_str += '1'

            data += chr(int(binary_str, 2))
            if pixels[-1] % 2 != 0:
                return data

    # function to generate data
    def generate_Data(self,data):
        new_data = []

        for i in data:
            new_data.append(format(ord(i), '08b'))
        return new_data



    #DataFlair- function to modify the pixels of image
    def modify_Pix(self,pix, data):
        dataList = self.generate_Data(data)
        dataLen = len(dataList)
        imgData = iter(pix)
        for i in range(dataLen):
            # Extracting 3 pixels at a time
            pix = [value for value in imgData.__next__()[:3] +
                   imgData.__next__()[:3] +
                   imgData.__next__()[:3]]

            for j in range(0, 8):
                if (dataList[i][j] == '0') and (pix[j] % 2 != 0):
                    if (pix[j] % 2 != 0):
                        pix[j] -= 1

                elif (dataList[i][j] == '1') and (pix[j] % 2 == 0):
                    pix[j] -= 1

            if (i == dataLen - 1):
                if (pix[-1] % 2 == 0):
                    pix[-1] -= 1
            else:
                if (pix[-1] % 2 != 0):
                    pix[-1] -= 1

            pix = tuple(pix)
            yield pix[0:3]
            yield pix[3:6]
            yield pix[6:9]


    # function to enter the data pixels in image
    def encode_enc(self,newImg, data):
        w = newImg.size[0]
        (x, y) = (0, 0)

        for pixel in self.modify_Pix(newImg.getdata(), data):

            # Putting modified pixels in the new image
            newImg.putpixel((x, y), pixel)
            if (x == w - 1):
                x = 0
                y += 1
            else:
                x += 1


    # function to enter hidden text
    def enc_fun(self,text_a,myImg):
        data = text_a.get("1.0", "end-1c")
        if (len(data) == 0):
            messagebox.showinfo("Alert","Kindly enter text in TextBox")
        else:
            newImg = myImg.copy()
            self.encode_enc(newImg, data)
            my_file = BytesIO()
            temp=os.path.splitext(os.path.basename(myImg.filename))[0]
            newImg.save(tkinter.filedialog.asksaveasfilename(initialfile=temp,filetypes =
([('png', '*.png')]),defaultextension=".png"))
            self.d_image_size = my_file.tell()
            self.d_image_w,self.d_image_h = newImg.size
            messagebox.showinfo("Success","Encoding Successful\nFile is saved as
Image_with_hiddentext.png in the same directory")

    def frame_3(self,frame):
        frame.destroy()
        self.main(root)



       #GUI loop
        root = Tk()
        o = IMG_Stegno()
        o.main(root)
        root.mainloop()
```

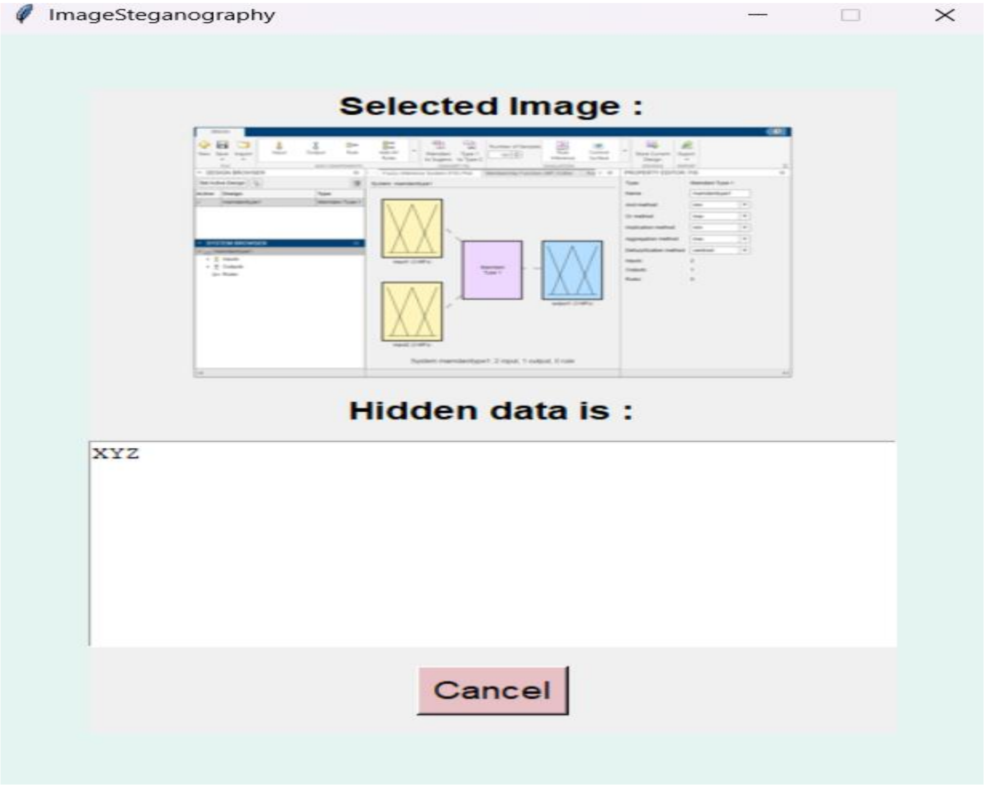
Output screenshot:





XYZ

Encodin



XYZ

Decoding