NRI INSTITUTE OF INFORMATION SCIENCE & TECHNOLOGY BHOPAL

DEPARTMENT OF MASTER OF COMPUTER APPLICATION

LAB MANUAL

LAB OF Elective

Advance Python, CCT , IS

MCA 404

MASTER OF COMPUTER APPLICATION (MCA)

NIIST BHOPAL	NRI INSTITUTE OF INFORMATION SCIENCE & TECHNOLOGY DEPARTMENT NAME:MASTER OF COMPUTER APPLICATION	FORM NO	NIIST /A/10
BRANCH: MCA	LIST OF EXPERIMENTS	REV. NO	0
SEM: IV		REV. DT	30/06/ 2011

SUBJECT/CODE : LAB OF Elective Advance Python, CCT , IS / MCA-404

1	Write a program to count the numbers of characters in the string and store them in a dictionary data structure Write a program to use split and join methods in the string and trace a birthday of a person with a dictionary data structure
2	Write a program to count frequency of characters in a given file. Can you use character frequency to tell whether the given file is a Python program file, C program file or a text file? Write a program to count frequency of characters in a given file. Can you use character frequency to tell whether the given file is a Python program file, C program file or a text file?
3	Write a program to print each line of a file in reverse order. Write a program to compute the number of characters, words and lines in a file.
4	Write function to compute gcd, lcm of two numbers.
5	Working of Google Drive to make spreadsheet and notes.
6	Installation and Configuration of Justcloud.
7	Working in Cloud9 to demonstrate different language.
8	Perform encryption, decryption using the following substitution techniques i. Ceaser cipher
9	Apply DES algorithm for practical applications.
10	Apply AES algorithm for practical applications

.

Experiment No. 1

AIM :-Write a program to count the numbers of characters in the string and store them in a dictionary data structure Write a program to use split and join methods in the string and trace a birthday of a person with a dictionary data structure

Code:-

```
str=input("enter string : ") f = {}
for i in str:
    if i in f:
        f[i] += 1
    else:
        f[i] = 1
print(f)
```

Write a program to use split and join methods in the string and trace a birthday of a person with a dictionary data structure

split Str.split() join
Str1.join(str2)

Algorithm

Step 1: Input a string.

Step 2: here we use split method for splitting and for joining use join function.

Example code

```
#split of string

str1=input("Enter first String with space :: ")

print(str1.split())                    #splits at space


str2=input("Enter second String with (,) :: ")

print(str2.split(','))                      #splits at ','


str3=input("Enter third String with (:) :: ")

print(str3.split(':'))                      #splits at ':'


str4=input("Enter fourth String with (;) :: ")

print(str4.split(';'))                      #splits at ';'


str5=input("Enter fifth String without space :: ")

print([str5[i:i+2]for i in range(0,len(str5),2)]) position 2                                    #splits at
```

Output

```
Enter first String with space :: python program ['python', 'program']

Enter second String with (,) :: python, program ['python', 'program']

Enter third String with (:) :: python: program ['python', 'program']

Enter fourth String with (;) :: python; program ['python', 'program']

Enter fifth String without space :: python program ['py', 'th', 'on', 'pr', 'og', 'ra', 'm']
```

Example Code

```
#string joining

str1=input("Enter first String                    :: ")

str2=input("Enter second String   :: ")

str=str2.join(str1) the #front          #each character of str1 is concatenated to
of str2

print("AFTER JOINING OF TWO STRING ::>",str)
```

Output

```
Enter first String                ::AAA
Enter second String   :: BBB
```

.

Experiment No. 2

AIM :- Write a program to count frequency of characters in a given file. Can you use character frequency to tell whether the given file is a Python program file, C program file or a text file? Write a program to count frequency of characters in a given file. Can you use character frequency to tell whether the given file is a Python program file, C program file or a text file?

```
str=input("enter string : ") f = {}

for i in str:

    if i in f:

        f[i] += 1

    else:

        f[i] = 1
print(f)
```

? Write a program to count frequency of characters in a given file.

Here is source code of the Python Program to count the occurrences of a letter in a text file. The program output is also shown below.

```
fname = input("Enter file name: ")
l=input("Enter letter to be searched:")
k = 0

with open(fname, 'r') as f:

    for line in f:

        words = line.split()
        for i in words:

            for letter in i:

                if(letter==l):
```

Program Explanation

1. User must enter a file name and the letter to be searched.
2. The file is opened using the open() function in the read mode.
3. A for loop is used to read through each line in the file.
4. Each line is split into a list of words using split().
5. A for loop is used to traverse through the words list and another for loop is used to traverse through the letters in the word.
6. If the letter provided by the user and the letter encountered over iteration are equal, the letter count is incremented.
7. The final count of occurrences of the letter is printed.

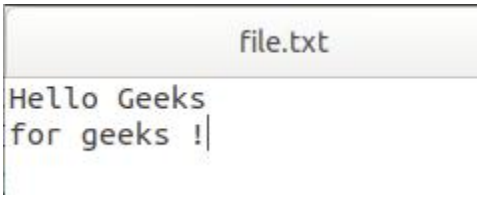
Experiment No. 3

AIM :- Write a program to print each line of a file in reverse order. Write a program to compute the number of characters, words and lines in a file.

Given a text file. The task is to reverse as well as stores the content from an input file to an output file.

This reversing can be performed in two types.

- **Full reversing:** In this type of reversing all the content get reversed.
- **Word to word reversing:** In this kind of reversing the last word comes first and the first word goes to the last position.
  - **ext file:**



- filter_none
- brightness_4

```
# Open the file in write mode f1 =
open("output1.txt", "w")

# Open the input file and get
# the content into a variable data with open("file.txt",
"r") as myfile:
        data =myfile.read()

# For Full Reversing we will store the
# value of data into new variable data_1
# in a reverse order using [start: end: step], # where step when passed
-1 will reverse
# the string data_1=data[::-
1]

# Now we will write the fully reverse # data in the
output1 file using
# following command
f1.write(data_1)

f1.close()
```

- **Output:**

output1.txt

```
! skeeg rof
skeeG olleH
```

Write a program to compute the number of characters, words and lines in a file.

```
import sys


fname = sys.argv[1]
lines = 0
words = 0
letters = 0

for line in open(fname):
    lines += 1
    letters += len(line)

    pos = 'out'
    for letter in line:

        if letter != ' ' and pos == 'out':
            words += 1
            pos = 'in'
        elif letter == ' ':
            pos = 'out'

print("Lines:", lines)

print("Words:", words)

print("Letters:", letters)
```

Experiment No. 4

AIM :-Write function to compute gcd, lcm of two numbers.

```
Python program to find LCM of two numbers # Python 3
program to find
# LCM of 2 numbers without # using
GCD
import sys

# Function to return # LCM of
two numbers def findLCM(a,
b):

    lar = max(a, b) small =
    min(a, b) i = lar
    while(1) :
        if (i % small == 0): return i
        i += lar

# Driver Code a = 5
b = 7
print("LCM of " , a , " and ",
                        b , " is " ,
      findLCM(a, b), sep = "")
```

```
LCM of 15 and 20 is 60
```

```
Recursive function to return gcd of a and b def gcd(a,b):

    # Everything divides 0 if (a == 0):
        return b if (b
    == 0):
        return a

    # base case if (a
    == b):
        return a

    # a is greater if (a >
    b):
        return gcd(a-b, b) return
    gcd(a, b-a)

# Driver program to test above function
```

```
a = 98
b = 56
if(gcd(a, b)):
        print('GCD of', a, 'and', b, 'is', gcd(a, b)) else:
        print('not found')
```

Output:

```
GCD of 98 and 56 is 14
```

Experiment No. 5

Objective: Working of Goggle Drive to make spreadsheet and notes.

Requirement: Google account, Internet Connection.

THEORY:

Google Docs is a free cloud-based suite of tools for creating documents, spreadsheets, presentations, and more. This tutorial will cover the **Spreadsheets** application in Google Docs, in addition to showing you how to access and store your Docs from **Google Drive**.

Google Docs, Sheets, and Slides are productivity apps that let you create different kinds of online documents, work on them in real time with other people, and store them in your Google Drive online — all for free. You can access the documents, spreadsheets, and presentations you create from any computer, anywhere in the world. (There's even some work you can do without an Internet connection!) This guide will give you a quick overview of the many things that you can do with Google Docs, Sheets, and Slides.

Google Docs

Google Docs is an online word processor that lets you create and format text documents and collaborate with other people in real time. Here's what you can do with Google Docs:

- Upload a Word document and convert it to a Google document
- Add flair and formatting to your documents by adjusting margins, spacing, fonts, and colors — all that fun stuff
- Invite other people to collaborate on a document with you, giving them edit, comment or view access
- Collaborate online in real time and chat with other collaborators — right from inside the document
- View your document's revision history and roll back to any previous version
- Download a Google document to your desktop as a Word, OpenOffice, RTF, PDF, HTML or zip file
- Translate a document to a different language
- Email your documents to other people as attachments

Google Sheets

Google Sheets is an online spreadsheet app that lets you create and format spreadsheets and simultaneously work with other people. Here's what you can do with Google Sheets:

- Import and convert Excel, .csv, .txt and .ods formatted data to a Google spreadsheet
- Export Excel, .csv, .txt and .ods formatted data, as well as PDF and HTML files
- Use formula editing to perform calculations on your data, and use formatting make it look the way you'd like
- Chat in real time with others who are editing your spreadsheet
- Create charts with your data
- Embed a spreadsheet — or individual sheets of your spreadsheet — on your blog or website

Google Slides

Google Slides is an online presentations app that allows you to show off your work in a visual way. Here's what you can do with Google Slides:

- Create and edit presentations
- Edit a presentation with friends or coworkers, and share it with others effortlessly
- Import .pptx and .pps files and convert them to Google presentations
- Download your presentations as a PDF, a PPT, or a .txt file
- Insert images and videos into your presentation
- Publish and embed your presentations in a website

Create, name or delete a Google document

Create a Google document

To create a new document, go to your Drive, click the **Create** button, and select **Document**.

A window with a new Google document will open, and you'll be able to edit the document, share it with other people, and collaborate on it in real-time. Google Docs saves your document automatically, and you can always access it from your Drive.

Name a document

When you create a new document, Google Docs will name it **Untitled** by default.

To choose a name other than **Untitled**, click the **File** menu, and select **Rename**. From here you can choose and confirm your document's title. You can also edit the name by clicking the title displayed at the top of the page, and making your changes in the dialog that appears. Titles can be up to 255 characters long.

Delete a document

Delete an item that you own from your Drive

1. From your Drive, select the item(s) you want to delete.
2. From the **More** menu, choose **Move to trash**.
3. If you're deleting a shared document that you own, you'll see an option to change the ownership of the document.
4. The item will be moved to the **Trash**.
5. To purge individual items from Trash, select them and choose **Delete forever**. To purge all your items click **Empty Trash** in the upper left.

Create and save a document

There are different ways of getting started using Google documents: you can create a new online document, you can upload an existing one, or you can use a template from our templates gallery.

To create a new document, go to your Drive, click the red **Create** button, and select **Document** from the drop-down menu.

As soon as you name the document or start typing, Google Docs will automatically save your work every few seconds. At the top of the document, you'll see text that indicates when your document was last saved. You can access your document at any time by opening your Drive at http://drive.google.com.

To save a copy of a document to your computer, you can download it. In your document, go to the **File** menu and point your mouse to the **Download as** option. Select one of the following file types: HTML (zipped), RTF, Word, Open Office, PDF, and plain text. Your document will download to your computer.

Upload a document

You can upload existing documents to Google documents at any time. When you're uploading, you can either keep your document in its original file type or convert it to Google Docs format. Converting your document to Google Docs format allows you to edit and collaborate online from any computer.

Note: When uploaded, images within a document are left as images (rather than being converted to text by Optical Character Recognition technology).

You can upload the following file types:

- .html
- .txt
- .odt
- .rtf
- .doc and .docx
- .pdf

Follow these steps to upload a document:

1. Click the **Upload** icon in the top left of your Documents List.
2. Click **Files...**, and select the document you'd like to upload.
3. Click **Open**.
4. Check the box next to 'Convert documents, presentations, spreadsheets, and drawings to the corresponding Google Docs format' if you'd like to be able to edit and collaborate on the document online. Uploaded document files that are converted to Google documents format can't be larger than 1 MB.
5. Click **Start upload**. The uploaded file will appear in your Documents List.

Experiment No. 6

Objective: Installation and Configuration of Justcloud.

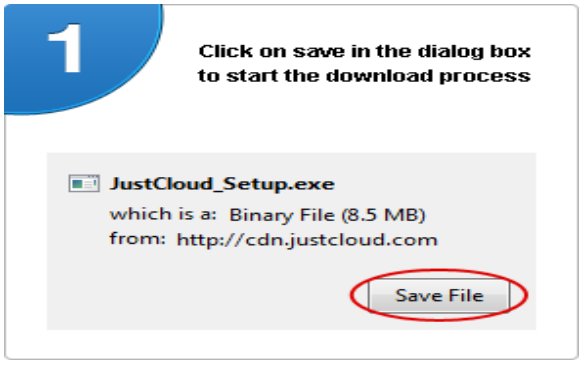
Requirement: Justcloud exe File

THEORY:

Professional Cloud Storage from JustCloud is Simple, Fast and Secure. Just Cloud will automatically backup the documents, photos, music and videos stored on your computer, to the cloud so you are never without files again.

Installation :

1. Download Software this link http://www.justcloud.com/download/



2. By following these steps you will download and install the JustCloud software application on this computer. This software will automatically start backing up files from your computer and saving them securely in an online cloud user account. Your free account gives you 15MB storage space or 50 files for 14 days. Once installed a sync folder will be added to your desktop for you to easily drag and drop files you wish to backup.

Experiment No. 7

Objective: Working in Cloud9 to demonstrate different

language. **Requirement:** Login account in Cloud9

THEORY:

Cloud9 IDE is an online development environment for JavaScript and Node.js applications as well as HTML, CSS, PHP, Java, Ruby and 23 other languages. Anyone looking for a modern and secure IDE. With your code online and accessible from anywhere, you can work more efficiently than before.

Creating a new account

Creating an account for the Cloud 9 IDE can be done in a few simple steps:

1. First, sign up for an account on the Cloud9 homepage, by filling in your desired username and email address and pressing the **Sign me up** button:

You will then see a message indicating that we have sent you an email to the address you provided with activation instructions:

2. Check your email now. You will receive an email from us with a link to activate the account. Click on the link. You will now be asked to set a password for your new Cloud9 account:

3. Click on **Activate**.

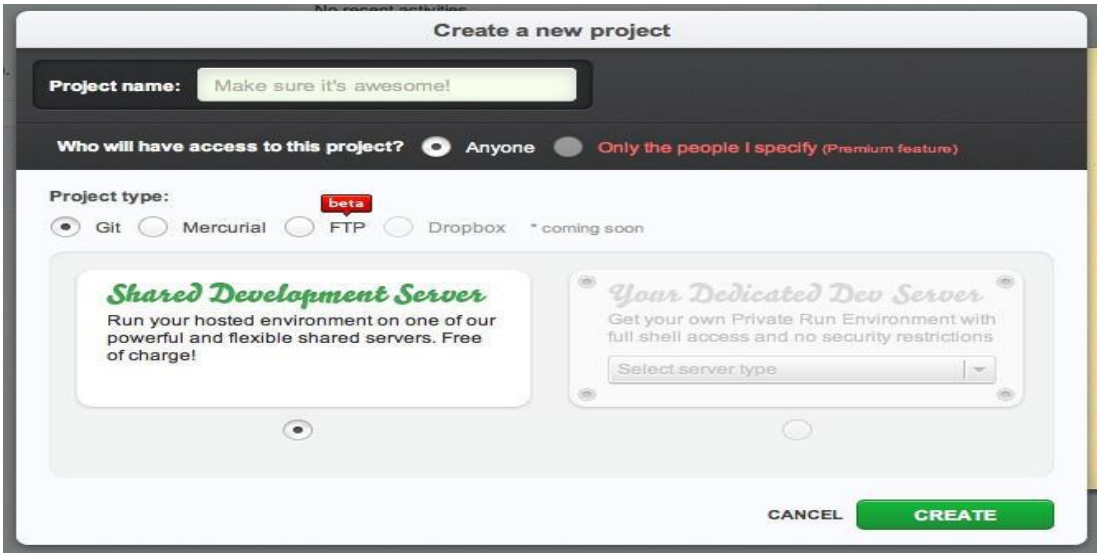
Congratulations! You are now the proud owner of a Cloud9 account. Now, go ahead and create your first project.

Create a new project

The first step for creating a new project is to click on the "+" next to **My Projects** in the Projects tab:

At this point, you will encounter two choices: **Create a new project** and **Clone from url**. We will explore both paths.

After clicking on **Create a new project**, you will be presented with the screen shown below:



Enter a project name. You will now have three choices for the type of project you wish to create:

- Git project: will allow you to run *git* commands from the console and push your changes to Github
- Mercurial: will allow you to run *hg* commands form the console and push your changes to Bitbucket.
- FTP: will allow you to upload your files directly to an FTP server you have access to.

Make a choice for the type of project and press **Create**. That is all! You will now see your new project in the dashboard:

Now, just click **Start Editing** to get started!

Let's start with a simple "Hello Cloud9" example.

1. From your dashboard, click 'create new workspace' and then select 'create new workspace'.
2. Enter a catchy workspace name, visibility: open (proud to share your creations), hosting: hosted and choose a 'custom' workspace type. Click 'create'.
3. The workspace is being prepared and when done, select the project in the dashboard and click 'start editing'.
4. The workspace is opened, right click the project tree and select 'new file'. Name it 'helloCloud9.cc'.
5. Open the file by double clicking it in the file tree. Copy / paste the following code in the file:

```
int main() {


  cout << "Hello world\n";


}
```

6. Compile the code using:

```
g++ helloCloud9.cc -o helloCloud9
```

7. Run the file by typing:

```
./helloWorld
```

Experiment No. 8

CAESAR CIPHER

AIM:

To implement a program for encrypting a plain text and decrypting a cipher text using Caesar Cipher (shift cipher) substitution technique

PRELAB DISCUSSION:

The Caesar cipher is one of the earliest known and simplest ciphers. It is a type of substitution cipher in which each letter in the plaintext is 'shifted' a certain number of places down the alphabet.

For example, with a shift of 1, A would be replaced by B, B would become C, and so on. The method is named after Julius Caesar, who apparently used it to communicate with his generals. More complex encryption schemes such as the Vigenere employ the Caesar cipher as one element of the encryption process. The widely known ROT13 'encryption' is simply a Caesar cipher with an offset of 13. The Caesar cipher offers essentially no communication security, and it will be shown that it can be easily broken even by hand.

To pass an encrypted message from one person to another, it is first necessary that both parties have the 'key' for the cipher, so that the sender may encrypt it and the receiver may decrypt it. For the caesar cipher, the key is the number of characters to shift the cipher alphabet.

First we translate all of our characters to numbers, 'a'=0, 'b'=1, 'c'=2, ... , 'z'=25. We can now represent the caesar cipher encryption function, e(x), where x is the character we are encrypting, as:

Where k is the key (the shift) applied to each letter. After applying this function the result is a number which must then be translated back into a letter. The decryption function is :

ALGORITHM:

1. Create and initialize a string ALPHABET that holds the alphabet characters. The index position of the string represents the numeric representation for the corresponding characters in the string ALPHABET.
2. Read the input plain text to be encrypted and also the Caeser cipher key an integer between 0 and 25.
3. Encrypt the plain text using the Caeser cipher key and the ALPHABET string.
    a. For every character in the plain text
        i. Search the ALPHABET string for the character and assign the numeric representation of the character (plainnumeric) as the index position of the character in the ALPHABET string.
        ii. Perform encryption using
            ciphernumeric = (plainnumeric + Caeser cipher key) mod 26
        iii. Use ciphernumeric as the index position and get the corresponding character from the ALPHABET string as the equivalent cipher text character for the plain text character
    b. Print the equivalent cipher text
4. Decrypt the cipher text using the Caeser cipher key and the ALPHABET string.
    a. For every character in the cipher text
        i. Search the ALPHABET string for the character and assign the numeric representation of the character (ciphernumeric) as the index position of the character in the ALPHABET

string.

ii. Perform decryption using
Plainnumeric = (ciphernumeric - Caeser cipher key) mod 26, if plainnumeric < 0 , plainnumeric = plainnumeric + 26

iii. Use plainnumeric as the index position and get the corresponding character from the ALPHABET string as the equivalent plain text character for the cipher text character
Print the equivalent plain text

1. Stop

PROGRAM:

```
import java.util.*;
import java.io.*;
public class Caesercipher
{
   public static final String ALPHABET =
   "abcdefghijklmnopqrstuvwxyz"; public static String encrypt(String
   ptext, int cserkey)
   {
      String ctext = "";
      for (int i = 0; i < ptext.length(); i++)
      {
         int plainnumeric =
         ALPHABET.indexOf(ptext.charAt(i)); int
         ciphernumeric = (plainnumeric+cserkey) %
         26;
         char cipherchar =
         ALPHABET.charAt(ciphernumeric); ctext +=
         cipherchar;
      }
      return ctext;
   }
   public static String decrypt(String ctext, int cserkey)
   {
      String ptext = "";
      for (int i = 0; i < ctext.length(); i++)
      {
         int ciphernumeric =
         ALPHABET.indexOf(ctext.charAt(i)); int
         plainnumeric= (ciphernumeric-cserkey) %
         26;
         if (plainnumeric < 0)
         {
            plainnumeric = ALPHABET.length() + plainnumeric;
         }
         char plainchar =
```

```
            ALPHABET.charAt(plainnumeric);
            ptext += plainchar;
        }

        return ptext;
    }

    public static void
     main(String[] args)
     throws IOException
     {

        BufferedReader br =  new BufferedReader(new InputStreamReader(System.in));
        System.out.println("Enter the PLAIN TEXT for Encryption: ");
        String plaintext =
        new String(); String
        ciphertext = new
        String(); String key;
        int cserkey;
        plaintext = br.readLine();
        System.out.println("Enter the CAESERKEY between 0
        and 25:"); key = br.readLine();
        cserkey = Integer.parseInt(key);
        System.out.println("EN
        CRYPTION");
        ciphertext =
        encrypt(plaintext,cserke
        y);
        System.out.println("CIPHER  TEXT  :"+  ciphertext);
        System.out.println("DE
        CRYPTION"); plaintext
        =
        decrypt(ciphertext,cserk
        ey);
        System.out.println("PLAIN TEXT :" + plaintext);
    }

}
```

OUTPUT:

```
C:\Program  Files\Java\jdk1.8.0_71\bin>javac  Caesercipher.java
C:\Program Files\Java\jdk1.8.0_71\bin>java Caesercipher


Enter the PLAIN TEXT for Encryption:
information
Enter the CAESERKEY
between 0 and 25: 7
ENCRYPTION
CIPHER TEXT :pumvythapvu
DECRYPTION
PLAIN TEXT :information
```

VIVA QUESTIONS (PRELAB and POSTLAB):

1. Crack the following plaintext TRVJRI TZGYVIJ RIV HLZKV VRJP KF TIRTB
2. What encryption key was used?

3. Make you own cipher text using the Caesar cipher.
4. Can you crack other people's ciphertexts?
5. What key do we need to make "CAESAR" become "MKOCKB"?
6. What key do we need to make "CIPHER" become "SYFXUH"?
7. Use the Caesar cipher to encrypt your first name
8. How can we find the decryption key from the encryption key?

RESULT:

Thus the program to implement caeser cipher encryption technique was developed and executed.

Experiment No. 9

RAIL FENCE CIPHER

AIM:

To develop a program for implementing encryption and decryption using rail fence transposition technique.

PRELAB DISCUSSION:

The rail fence is the simplest example of a class of transposition ciphers, known as route ciphers. In general, the elements of the plaintext (usually single letters) are written in a prearranged order (route) into a geometric array (matrix)—typically a rectangle—agreed upon in advance by the transmitter and receiver and then read off by following another prescribed route through the matrix to produce the cipher. The key in a route cipher consists of keeping secret the geometric array, the starting point, and the routes. Clearly both the matrix and the routes can be much more complex than in this example; but even so, they provide little security. One form of transposition (permutation) that was widely used depends on an easily remembered key word for identifying the route in which the columns of a rectangular matrix are to be read. For example, using the key word AUTHOR and ordering the columns by the lexicographic order of the letters in the key word.

In decrypting a route cipher, the receiver enters the cipher text symbols into the agreed-upon matrix according to the encryption route and then reads the plaintext according to the original order of entry. A significant improvement in crypto security can be achieved by reencrypting the cipher obtained from one transposition with another transposition. Because the result (product) of two transpositions is also a transposition, the effect of multiple transpositions is to define a complex route in the matrix, which in itself would be difficult to describe by any simple mnemonic.

ALGORITHM DESCRIPTION:

In the rail fence cipher, the plaintext is written downwards and diagonally on successive "rails" of an imaginary fence, then moving up when we reach the bottom rail. When we reach the top rail, the message is written downwards again until the whole plaintext is written out. The message is then read off in rows.

1. Generate numerical key from the word key by the characters of the word in alphabetical order.
2. Encryption
    i. The plain text is written in the matrix form, where the column of the matrix is number of characters in the word key and row of the matrix is to accommodate the characters of the plain text and the space left after the plain text characters in the last row is filled with any character. (eg. x or z)
    ii. The cipher text is generated by reading the characters column by column in the order specified in the numerical key.
3. Decryption
    i. The characters in the cipher text are filled in the matrix of same order used for encryption, but in the order specified in the key. The characters from cipher text equal to the number of rows in matrix are taken and filled in the matrix column based on the order specified in the key
    ii. The plain text is generated from cipher text by reading the characters from the matrix row by row.
4. Stop

PROGRAM:

```
import java.util.*;
import java.io.*;
public class Railfence
{
        public static int key[] = new int[8];
        public char mat[][] = new char[10][8];
        public char pmat[][] = new char[10][8];
        public char cmat[][] = new char[10][8];
        String plain="";
        String cipher="";
        int rows=0, col;
        public String rfencryption(String text1)
        {
            int i,j,len,ch,k,p=0;
            String enctxt="";
            String text = "";
            len = text1.length();
            for(i=0;i<len;i++)
                text += text1.charAt(i);
            if (( len % 7 ) != 0)
            {
                rows = ( len / 7 ) + 1;
                ch = len % 7;
                for  (i=0;i<(7-ch);i++)
                    text += 'X';
            }
          else
                rows = len / 7;
                k=0;
                for(i=1;i<=rows;i++)
                {
                  for(j=1;j<=7;j++)
                      mat[i][j] = text.charAt(k++);
                }
            for(i=1;i<=rows;i++)
            {
                  for(j=1;j<=7;j++)
                      System.out.print(mat[i][j] + " ");
                  System.out.println();
                }
            k = 1;
            j = 1;
            while ( k <= 7 )
            {
                  for(p=0;p<7;p++)
                  {
```

```
                    if ( k == key[p] )
                    {
                                j=p+1;
                                k++;
                                break;
                    }
          }
          for(i=1;i<=rows;i++)
                    enctxt+=mat[i][j];
                    }
    System.out.println(enctxt); return enctxt;
    }
public String rfdecryption(String txt,int plength)
{
    int i,j=1,len,k=1,p,q=0;
    String dectxt="";
    String ptext="";
      while (k<=7)
      {
          for(p=0;p<7;p++)
          {
                    if (key[p] == k)
                    {
                        j = p+1;
                        k++;
                    break;
               }
     }
          for(i=1;i<=rows;i++)
                    cmat[i][j] = txt.charAt(q++);
                        }
      for(i=1;i<=rows;i++)
      {
          for(j=1;j<=7;j++)
                    System.out.print(
          cmat[i][j] + " ");
          System.out.println();
      }
      for(i=1;i<=rows;i++)
```

```
        {
                for(j=1;j<=7;j++)
                        dectxt += cmat[i][j];
        }
        len = dectxt.length();
        if (plength < len)
        {
                for(i=0;i<plength;i++)
                        ptext += dectxt.charAt(i);
        }
        return ptext;
}
public static void
main(String[] args) throws
IOException
{
int i=0;
int k;
String c;
        BufferedReader br =  new BufferedReader(new
        InputStreamReader(System.in));  Railfence rf = new Railfence();
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter
        key"); for(i=0;i<7;i++)
        {
                c = br.readLine();
                key[i] = Integer.parseInt(c);
        }
        for(i=0;i<7;i++)
                System.out.print(key[i] + " ");
        String plain = new String();
        System.out.println("Enter
        PLAIN TEXT"); plain =
        sc.next();
        k=plain.length();
                System.out.println(plain);
                String ctext = new String();
                ctext = rf.rfencryption(plain);
                System.out.println();
        System.out.println("CIPHER TEXT :" + ctext);
        System.out.println();
                String plaintext = new String();
```

```
                    plaintext = rf.rfdecryption(ctext,k);

                    System.out.println();

                System.out.println("PLAIN TEXT :" + plaintext);
                sc.close();
            }

        }
```

OUTPUT:

```
C:\Program Files\Java\jdk1.8.0_71\bin>javac
Railfence.java C:\Program
Files\Java\jdk1.8.0_71\bin>java Railfence
Enter key
 AUTHOR
A U T H O R A H O R T U

 KEY : NUMERICAL
 REPRESENTATION  0 3 4 5
 2 1
                Enter PLAIN TEXT ATTACKPOSTPONEDUNTILTWOAM
                ATTACKPOSTPONEDUNTILTWOAM

A T T A C K P O S T P O N E D U N T I L T W O A M X X X X X

 APNIMKOTAXCPNOXTOELXTSDTXATUWX
 CIPHER TEXT :APNIMKOTAXCPNOXTOELXTSDTXATUWX
A T T A C K P O S T P O N E D U N T I L T W O A M X X X X X

 PLAIN TEXT :ATTACKPOSTPONEDUNTILTWOAM
C:\Program  Files\Java\jdk1.8.0_71\bin>
```

VIVA QUESTIONS (PRELAB and POSTLAB):

1. Where do you apply PGP?
2. List out the basic tasks in Public Key Encryption in key distribution.
3. Give an example for Simple Hash Function.
4. List out the two methods of operations in Authentication Header (AH) and Encapsulating Security Payload (ESP).
5. Enumerate the functions provided by S/MIME.
6. List out the two ways in which password can be protected.
7. Which attack is related to integrity?
8. Which public key cryptosystem can be used for digital signature?
9. Expand: S/MIME.
10. What is the use of trusted system?

RESULT:

Thus the program for Railfence cipher was executed and verified successfully.

Experiment No. 10

DATA ENCRYPTION STANDARD (DES)

AIM:

To develop a program to implement Data Encryption Standard for encryption and decryption.

PRELAB DISCUSSION:

- The Data Encryption Standard (DES) is a symmetric-key block cipher published by the National Institute of Standards and Technology (NIST).
- DES is an implementation of a Feistel Cipher. It uses 16 round Feistel structure. The block size is 64-bit.
- Though, key length is 64-bit, DES has an effective key length of 56 bits, since 8 of the 64 bits of the key are not used by the encryption algorithm (function as check bits only).
- General Structure of DES is depicted in the following illustration

ALGORITHM:

1. Process the key.
   i. Get a 64-bit key from the user.
   ii. Calculate the key schedule.
      1. Perform the following permutation on the 64-bit key. The parity bits are discarded, reducing the key to 56 bits. Bit 1 of the permuted block is bit 57 of the original key, bit 2 is bit 49, and so on with bit 56 being bit 4 of the original key.
      2. Split the permuted key into two halves. The first 28 bits are called C[0] and the last 28 bits are called D[0].
      3. Calculate the 16 subkeys. Start with i = 1.
         1. Perform one or two circular left shifts on both C[i-1] and D[i-1] to get C[i] and D[i], respectively. The number of shifts per iteration are given in the table below.

            Iteration # 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16

            Left Shifts 1 1 2 2 2 2 2 2 1 2 2 2 2 2 2 1

         2. Permute the concatenation C[i]D[i] as indicated below. This will yield K[i], which is 48 bits long.
         3. Loop back to 1.ii.c.1 until K[16]

has been calculated. 2 Process a 64-bit data block.

   i. Get a 64-bit data block. If the block is shorter than 64 bits, it should be padded as appropriate for the application.
   ii. Perform the initial permutation on the data block.
   iii. Split the block into two halves. The first 32 bits are called L[0], and the last 32 bits are called R[0].
   iv. Apply the 16 subkeys to the data block. Start with i = 1.
      a. Expand the 32-bit R[i-1] into 48 bits according to the bit-selection function Expansion (E)

b. Exclusive-or E(R[i-1]) with K[i].

c. Break E(R[i-1]) xor K[i] into eight 6-bit blocks. Bits 1-6 are B[1], bits 7-12 are B[2], and so on with bits 43-48 being B[8].

d. Substitute the values found in the S-boxes for all B[j]. Start with j = 1. All values in the S-boxes should be considered 4 bits wide.

i. Take the 1st and 6th bits of B[j] together as a 2-bit value (call it m) indicating the row in S[j] to look in for the substitution.

ii. Take the 2nd through 5th bits of B[j] together as a 4-bit value(call it n) indicating the column in S[j] to find the substitution.

iii. Replace B[j] with S[j][m][n].

iv. Loop back to 2.iv.d.i until all 8 blocks have been replaced.

e. Permute the concatenation of B[1] through B[8]

f. Exclusive-or the resulting value with L[i-1]. Thus, all together, your R[i] = L[i-1] xor P(S[1](B[1])...S[8](B[8])), where B[j] is a 6-bit block of E(R[i-1]) xor K[i]. (The function for R[i] is written as, R[i] = L[i-1] xor f(R[i-1], K[i]).) g. L[i] = R[i-1].

h. Loop back to 2.iv.a until K[16] has been applied.

v. Perform the final permutation on the block R[16]L[16].

3. Decryption : Use the keys K[i] in reverse order. That is, instead of applying K[1] for the first iteration, apply K[16], and then K[15] for the second, on down to K[1]

PROGRAM:

DES :-

```
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();
intnum = 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 { KeyGeneratorkgen =
KeyGenerator.getInstance("DES"); SecureRandomsr
= SecureRandom.getInstance("SHA1PRNG");
sr.setSeed(seed);
kgen.init(56, sr);
SecretKeyskey = kgen.generateKey();
raw = skey.getEncoded();
return raw;
}
private static byte[] encrypt(byte[] raw, byte[] clear) throws Exception
{
SecretKeySpecskeySpec = 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

        {

        SecretKeySpecskeySpec = new SecretKeySpec(raw, "DES");

Cipher cipher =
Cipher.getInstance("DES");
cipher.init(Cipher.DECRYPT_MOD
E, skeySpec); byte[] decrypted =
cipher.doFinal(encrypted); return
decrypted;
}

public static void main(String args[])

{

DES des = new DES();

 }

 }
```

OUTPUT:

VIVA QUESTIONS (PRELAB and POSTLAB):

1. DES follows which basic stream cipher?
2. The DES Algorithm Cipher System consists of how many rounds (iterations) each with a round key?
3. What is the key length of the DES algorithm?
4. In the DES algorithm, although the key size is 64 bits only 48bits are used for the encryption procedure, the rest are parity bits. Is it true or false?
5. In the DES algorithm, what is the size of the round key and the Round Input?
6. In the DES algorithm how the Round Input is expanded to 48 ?
7. What is size of the Initial Permutation table/matrix
8. How many unique substitution boxes are in DES after the 48 bit XOR operation?
9. In the DES algorithm the 64 bit key input is shortened to 56 bits by ignoring every 4th bit. Is it true or false?

RESULT:

Thus the program to implement DES encryption technique was developed and executed successfully