Linear Algebra Take-home

Alekhya E (01FB15ECS032)

May 1, 2017

INDEX

Take Home-1

Linear Separability

PROBLEM STATEMENT: For the given set of samples,
1.S.T the data is linearly separable
b.S.T f1 = f2 is a decision boundary
c.Obtain the weight vector,w. S.T it is orthogonal to the decision boundary.
d.Suggest another decision boundary for the same data.
e.Compute the distance from the origin to the decision boundary
f.By transforming and normalizing the data, reconstruct the table and use perceptron learning algorithm to compute the new weight vector.

label no	f1	f2	class
1	0.5	3	X
2	1	3	X
3	0.5	2.5	X
4	1	2.5	X
5	1.5	2.5	X
6	4.5	1	O
7	5	1	O
8	4.5	0.5	O
9	5.5	0.5	O

Method used: I used perceptron algorithm and if the data converges in some finite amount of time(i.e in 1000 number of iterations) Then the data is said to be linearly separable.
Algorithm:

$Initializingthreshold_value \leftarrow 0$
$Initializinglearnrate \leftarrow 0.2i.eanynumberbetween0and1$
$Initializingierror \leftarrow -1$
$Initiallyinitializingtheweightarraywithrandomvaluesusingrandom.randint()withvaluesbetween0and1$
$Initializingtheconvergenceas0$ $count \leftarrow 0$
while $count \leq 1000andierror! = 0$ **do**
  $count \leftarrow count + 1$
  for $allthegivendataintrainingset$ **do**
    $output \leftarrow i[0] * wt[0] + i[1] * wt[1] + wt[2]$
    if $output > threshold_value$ **then**
      $output \leftarrow 1$
    else
      $output \leftarrow 0$
    end if
    $error \leftarrow expectedoutput - output$
    $ierror \leftarrow ierror + error * error$
    $weight1 \leftarrow weight1 + learnrate * error * f1_i$
    $weight2 \leftarrow weightt2 + learnrate * error * f2_i$
    $b \leftarrow b + learnrate * error$
  end for
  if $ierror = 0$ **then**
    $convergence \leftarrow 1$
    Exit while loop
  end if
end while
if $convergence = 1$ **then**
  The given data is linearly separable

else
    The given data is not linearly separable
end if

CODE

```
import math
import matplotlib.pyplot as plt
import random
import numpy as np
import statistics as stat

f=[[0.5,3],[1,3],[0.5,2.5],[1,2.5],[1.5,2.5],[4.5,1],[5,1],[4.5,0.5],[5.5,0.5]]
class1=[0,0,0,0,0,1,1,1,1]
threshold_value=0
learnrate=0.2
ierror=-1
count=0
weight=[]
weight.append(random.randint(1,1000)/1000)
weight.append(random.randint(1,1000)/1000)
weight.append(random.randint(1,1000)/1000)


convergence=0
while(count<=1000 and ierror!=0):
        count=count+1
        j=0
        ierror=0
        for i in f:
             op=i[0]*weight[0]+i[1]*weight[1]+weight[2]
             if(op>threshold_value):
                     return 1
             else:
                     return 0

                 error=class1[j]-op
                 j=j+1
                 weight[0]=weight[0]+learnrate*error*i[0]
                 weight[1]=weight[1]+learnrate*error*i[1]
                 weight[2]=weight[2]+learnrate*error
                 ierror=ierror+(error*error)
        if(ierror==0):
            convergence=1
if(convergence==1):
    print("The given data is linearly separable")
    print("The number of iterations are given by:",count)
else:
    print("data is not linearly separable")
```

Result

Output:

The given data is linearly separable The number of iterations are given by:4

part-b

To show that f1=f2 is decision boundary

Method Used:

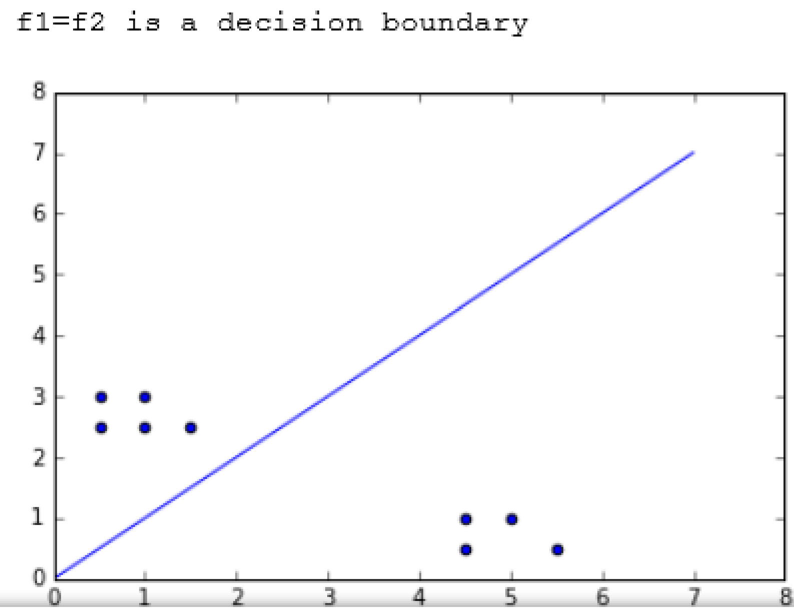
Basically for this problem the weight vector is 1x-1y=0 (f1-f2=0) with weight1=1 and weight2=-1 and with bias=0.For each data point in the given sample we find $.weigthvector^{transpose} sample$.If this value is greater than the threshold value then it belongs to class1 i.e (class 0 in this case) otherwise to class 0(i.e class X in this case)

Code:

```
%matplotlib inline
import math
import numpy as np
import statistics as stat
import matplotlib.pyplot as plt
import random
f=[[0.5,3],[1,3],[0.5,2.5],[1,2.5],[1.5,2.5],[4.5,1],[5,1],[4.5,0.5],[5.5,0.5]]
class1=[0,0,0,0,0,1,1,1,1]
learnrate=0.2
threshold_value=0
ierror=-1
weight=[]
weight.append(random.randint(1,1000)/1000)
weight.append(random.randint(1,1000)/1000)
weight.append(random.randint(1,1000)/1000)
count=0
bias=0
weight1=1
weight2=-1

 decision_boundary=1


j=0
for i in f:
     op=weight1*i[0]+weight2*i[1]+bias
          if(op>threshold_value):
                 return 1
          else:
                 return 0

          error=class1[j]-op
          j=j+1
          if(error!=0):
                  print("f1=f2 is not a decision boundary")
                  decision_boundary=0
                  break
if( decision_boundary==1):
    print("f1=f2 is a decision boundary")
    x1_p=np.linspace(0,7,7)
    y1_p=1*x1_p
    plt.plot(x1_p,y1_p)
    plt.scatter(f1,f2)
    plt.ylim(ymin=0)
    plt.xlim(xmin=0)
    plt.show()
```

RESULT: Output:

Graph



f1=f2 is a decision boundary.

PART-C

Show that the decision boundary is orthogonal to weight vector.

Method Used:

Basically considering two points p and q on the decision boundary f=$w^T * X + b$,and if $w^T(p-q) = 0$ i.e (the inner product of the weight vector with a vector in the direction of p and q)it implies that the weight vector is orthogonal to the decision boundary.

```
%matplotlib inline
import math
import numpy as np
import statistics as stat
import matplotlib.pyplot as plt
import random
f=[[0.5,3],[1,3],[0.5,2.5],[1,2.5],[1.5,2.5],[4.5,1],[5,1],[4.5,0.5],[5.5,0.5]]
class1=[0,0,0,0,0,1,1,1,1]
learnrate=0.2
threshold_value=0
ierror=-1
count=0
weight1=1
weight2=-1
weight=[]
weight.append(random.randint(1,1000)/1000)
weight.append(random.randint(1,1000)/1000)
weight.append(random.randint(1,1000)/1000)

bias=0
#when p=(1, 1) and q=(2,2) which lie on the decision boundary f1=f2
point1=[1,1]
point2=[2,2]
weight1=1
weight2=-1
```

```python
bias=0

point2_diff_point1=[1,1]
weight_difference=point2_diff_point1[0]*weight1+point2_diff_point1[1]*weight2
if(weight_difference==0):
    print("The weight  vector is orthogonal to above decision boundary")
else:
    print("The weight  vector is not orthogonal to the above decision boundary")
```

Output:

The weight vector is orthogonal to above decision boundary

part-d

Suggest another decision boundary for the same data.

Method Used:

Actually used the perceptron algorithm to find another decision boundary for the given data points. Algorithm:

$Initializingthreshold_value \leftarrow 0$
$Initializinglearnrate \leftarrow 0.2i.eanynumberbetween0and1$
$Initializingierror \leftarrow -1$
$Initiallyinitializingtheweightarraywithrandomvaluesusingrandom.randint()withvaluesbetween0and1$
$Initializingtheconvergenceas0$ $count \leftarrow 0$
while $count \leq 1000andierror! = 0$ **do**
  $count \leftarrow count + 1$
  for $allthegivendataintrainingset$ **do**
    $output \leftarrow i[0] * weight[0] + i[1] * weight[1] + weight[2]$
    if $output > threshold_value$ **then**
      $output \leftarrow 1$
    else
      $output \leftarrow 0$
    end if
    $error \leftarrow expectedoutput - output$
    $ierror \leftarrow ierror + error * error$
    $weight1 \leftarrow weight1 + learnrate * error * f1_i$
    $weight2 \leftarrow weightt2 + learnrate * error * f2_i$
    $b \leftarrow b + learnrate * error$
  end for
  if $ierror = 0$ **then**
    $convergence \leftarrow 1$
    Exit while loop
  end if
end while
if $convergence = 1$ **then**
  The given data is linearly separable
else
  The given data is not linearly separable
end if

Code :

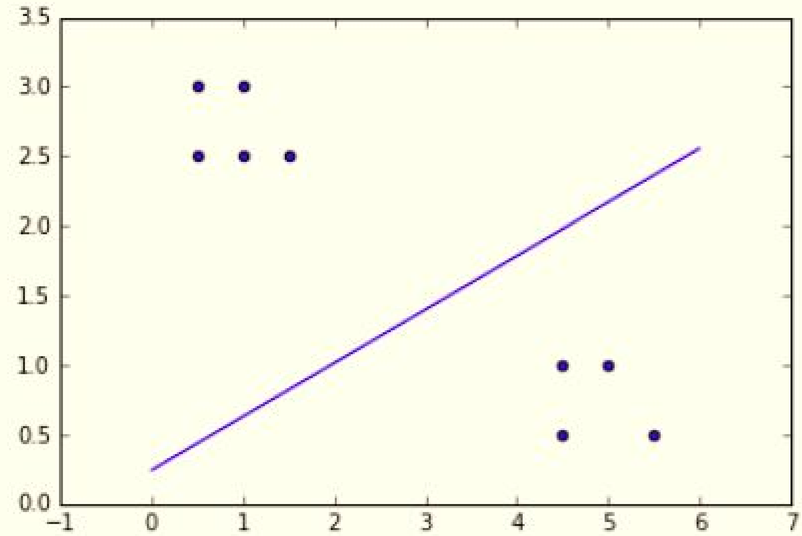
```python
import math
import numpy as np
import statistics as stat
```

```
import matplotlib.pyplot as plt
import random
f=[[0.5,3],[1,3],[0.5,2.5],[1,2.5],[1.5,2.5],[4.5,1],[5,1],[4.5,0.5],[5.5,0.5]]
class1=[0,0,0,0,0,1,1,1,1]
learnrate=0.2
threshold_value=0
ierror=-1
weight=[]
weight.append(random.randint(1,1000)/1000)
weight.append(random.randint(1,1000)/1000)
weight.append(random.randint(1,1000)/1000)
count=0


convergence=0
while(count<=10000 and ierror!=0):
        count=count+1
        j=0
        ierror=0
        for i in f:
            op=i[0]*weight[0]+i[1]*weight[1]+weight[2]
            if(op>threshold_value):
                   return 1
              else:
                  return 0

                error=class1[j]-op
                j=j+1
                weight[0]=weight[0]+learnrate*error*i[0]
                weight[1]=weight[1]+learnrate*error*i[1]
                weight[2]=weight[2]+learnrate*error
                ierror=ierror+(error*error)
        if(ierror==0):
            convergence=1

if(convergence==1):
    print("data is linearlly separable")
    print("decision boundary eqn is:" ,weight[0] ,"*x", weight[1], "*y","+",weight[2],"=0")
    f1=[i[0] for i in f]
    f2=[i[1] for i in f]
    plt.scatter(f1,f2)
    x_p=np.linspace(0,6,10)
    y_p=(-weight[2]-weight[0]*x_p)/weight[1]
    plt.plot(x_p,y_p)
    plt.show()
else:
    print("data is not linearlly separable")
```

Output:

Graph

```
w1= 0.791 0.945 0.784
decision boundary eqn is: 0.29100000000000004 *x -0.7550000000000002 *y + 0.18
```



part-e

Find the distance from origin to the separating hyperplane.

Method:

If line is given by the equation ax + by + c = 0, where a, b and c are real constants , the distance from the line to a point (x0,y0) is given by the formula Used the formula of finding distance of a point from a line. If a point is (x_0, y_0) then distance from a line Ax +By + C = 0 is (Ax_0 + By_0 + C)/($\sqrt{A^2 + B^2}$)
Here (x_0, y_0) is (0,0) and the hyperplane equation is same as the output of previous question.

Method used: So to find the distance of origin from decision boundary The distance d=weight[2]/math.sqrt(weight[0]**2+weig

Output:
The distance is given by 0.38534267477732

part-f

Problem Statement: By transforming and normalising the data, reconstruct the table and use perceptron learning algorithm to find the decision boundary.

Method:

Basically normalizing the data using the formula x-mean/standard deviation where x is the data point mean is the mean of the feature to which the datapoint belongs to and standard deviation is also the standard deviation of the feature to which the datapoint belongs to.
After normalizing the given data points applying perceptron to find the decision boundary.

Algorithm:

$Initializingthreshold_value \leftarrow 0$
$Initializinglearnrate \leftarrow 0.2i.eanynumberbetween0and1$
$Initializingierror \leftarrow -1$
$Initiallyinitializingtheweightarraywithrandomvaluesusingrandom.randint()withvaluesbetween0and1$
$Initializingtheconvergenceas0$ $count \leftarrow 0$
while $count \leq 1000andierror! = 0$ **do**
  $count \leftarrow count + 1$
  for $allthegivendataintrainingset$ **do**
    $output \leftarrow i[0] * wt[0] + i[1] * wt[1] + wt[2]$
    if $output > threshold_value$ **then**
      $output \leftarrow 1$
    else
      $output \leftarrow 0$
    end if
    $error \leftarrow expectedoutput - output$
    $ierror \leftarrow ierror + error * error$
    $weight1 \leftarrow weight1 + learnrate * error * f1_i$
    $weight2 \leftarrow weightt2 + learnrate * error * f2_i$
    $b \leftarrow b + learnrate * error$
  end for
  if $ierror = 0$ **then**

convergence $\leftarrow 1$
Exit while loop
end if
end while
if *convergence* $= 1$ **then**
The given data is linearly separable
else
The given data is not linearly separable
end if

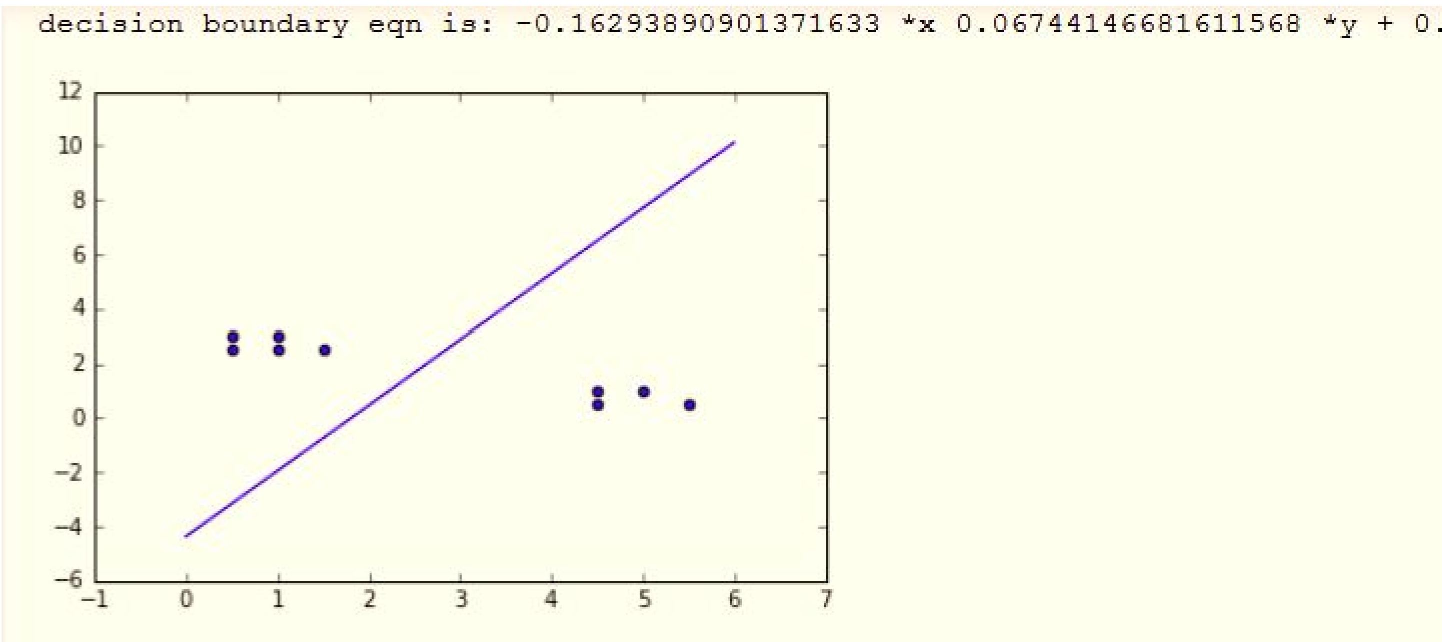
Code :

```
import math
import numpy as np
import statistics as stat
import matplotlib.pyplot as plt
import random
f=[[0.5,3],[1,3],[0.5,2.5],[1,2.5],[1.5,2.5],[4.5,1],[5,1],[4.5,0.5],[5.5,0.5]]
class1=[0,0,0,0,0,1,1,1,1]
f1=[i[0] for i in f]
f2=[i[1] for i in f]
stddev_of_f1=stat.stdev(f1)
stddev_of_f2=stat.stdev(f2)
mean_of_f1=stat.mean(f1)
mean_of_f2=stat.mean(f2)

nf=[]
for i in range(0,len(f1)):
    x=[]
    j=(f1[i]-mean_of_f1)/stddev_of_f1
    y=(f2[i]-mean_of_f2)/stddev_of_f2
    x.append(j)
    x.append(y)
    nf.append(x)
print(nf)

learnrate=0.2
threshold_value=0
ierror=-1
weight=[]
weight.append(random.randint(1,1000)/1000)
weight.append(random.randint(1,1000)/1000)
weight.append(random.randint(1,1000)/1000)
count=0


convergence=0
while(count<=10000 and ierror!=0):
        count=count+1
        j=0
        ierror=0
        for i in f:
                output=i[0]*weight[0]+i[1]*weight[1]+weight[2]

            if(output>threshold_value):
                    return 1
              else:
                     return 0
                error=class1[j]-output
                j=j+1
                weight[0]=weight[0]+learnrate*error*i[0]
                weight[1]=weight[1]+learnrate*error*i[1]
                weight[2]=weight[2]+learnrate*error
                ierror=ierror+(error*error)
        if(ierror==0):
```

```
            convergence=1
if(convergence==1):
    print("data is linearlly separable")
    print("decision boundary eqn is:" ,weight[0] ,"*x", weight[1], "*y","+",weight[2],"=0")
    f1=[i[0] for i in f]
    f2=[i[1] for i in f]
    plt.scatter(f1,f2)
    x_p=np.linspace(0,6,10)
    y_p=(-weight[2]-weight[0]*x_p)/weight[1]
    plt.plot(x_p,y_p)
    plt.show()
else:
    print("data is not linearlly separable")
```

Output:

Graph



Refrences:

`https://www.youtube.com/watch?v=1XkjVl-j8MM&t=9s`

`https://www.quora.com/How-can-I-know-whether-my-data-is-linearly-separable#!n=18`

`http://mathworld.wolfram.com/Point-PlaneDistance.html`

Take Home-2

Diagonalization Theorem

PROBLEM STATEMENT:

Prove the Diagonalization Theorem $P^{-1}AP = D$.Show that columns of P are eigen vectors of A.

ASSUMPTIONS:

$$1.A \in R^n$$

$$\lambda_1, \lambda_2......\lambda_n \in R$$

$$v_1, v_2, ...v_n \in R^n$$

The theorem holds good only if the matrix A is a square matrix.

SUMMARY:

Suppose A is a n by n matrix.For $P^{-1}APisadiagonalmatrixD$ D is a diagonal matrix where the diagonal elements are eigen values of A.P is a matrix containing eigen vectors of A.

Proof:

Basically from eigen-vector and eigen value relation we know that

$$Av_1 = \lambda_1 v_1, Av_2 = \lambda_2 v_2, Av_3 = \lambda_3 v_3, ...Av_n = \lambda_n v_n,$$

Putting eigen vectors in the columns of the matrix P.
AP=A*

$$\begin{bmatrix} v_1 & v_2 & v_3 & \dots & v_n \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}$$

$$\begin{bmatrix} Av_1 & Av_2 & Av_3 & \dots & Av_n \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}$$

$$\begin{bmatrix} \lambda_1 v_1 & \lambda_2 v_2 & \lambda_3 v_3 & \dots & \lambda_N v_n \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}$$

Now if we observe the above matrix it is just a scalar multiple of the matrix P. So $AP =$

$$\begin{bmatrix} \lambda_1 v_1 & \lambda_2 v_2 & \lambda_3 v_3 & \dots & \lambda_N v_n \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}$$

So now AP=PD

AP=P x

$$\begin{bmatrix} lambda_1 & & \\ & \ddots & \\ & & lambda_n \end{bmatrix}$$

$P^{-1}AP = D$

The above given matrix a Diagonal matrix with the eigen values of matrix A on the diagonal. Hence proved that $P^{-1}AP = D$.Columns of P are eigen vectors of A.

Refrences:

Linear Algebra and its Applications by Gilbert Strang

Take Home-3

Problem Statement: $||A||_0 = \sum_{i=1}^{n} \sum_{j=1}^{n} |a_{ji}| isamatrixnorm$

Proof:

To prove that $||A||_o$ is a matrix norm , the following properties has to be satisfied:

1. Non - negativity of matrix norm i.e matrix norm is always positive.

2.$||\alpha A||_o = \alpha ||A||_o$

3.$||A||_o = 0$, if and only if all the elements in the matrix are 0.

4.$||A + B||_o <= ||A||_o + ||B||_o$

5.$||A.B||_o <= ||A||_o.||B||_o$

Property number 1: Since the matrix norm is sum of all the elements row-wise and we are considering the absolute value of this.So the matrix norm is always positive.

Property 2:
$||\alpha A_O|| = \sum_{i=1}^{n} \sum_{j=1}^{n} |\alpha a_{ij}|$

Since α is just a scalar we can take it out. $||\alpha A_O|| = |\alpha|(\sum_{i=1}^{n} \sum_{j=1}^{n} |a_{ij}|)$

Property 3:
Since the matrix norm is sum of all the elements row-wise and we are considering the absolute value of this.So the matrix norm is always positive.But it will be 0 if all the elements of the matrix A are 0.

Property 4:

We use Cauchy Schwartz inequality to prove this property.

$$||A + B||_o = \sum_{i=1}^{n} \sum_{j=1}^{n} |a_{ij} + b_{ij}|$$
$$\leqslant \sum_{i=1}^{n} \sum_{j=1}^{n} |a_{ji}| + \sum_{i=1}^{n} \sum_{j=1}^{n} |b_{ji}|$$

$<= ||A|| + ||B||$
property-5
$||A.B||_o <= ||A||_o.||B||_o$

$$= \sum_{i=1}^{n} \sum_{j=1}^{n} \left[\sum_{k=1}^{n} |a_{jk} b_{ki}| \right]$$
$$\leqslant \sum_{i=1}^{n} \sum_{j=1}^{n} \left[\sum_{k=1}^{n} |a_{jk}| [\sum_{k=1}^{n} |b_{ki}| \right]$$
$$\leqslant \sum_{i=1}^{n} \sum_{k=1}^{n} [a_{jk}| * \sum_{i=1}^{n} \sum_{k=1}^{n} [b_{ki}|$$
$$\leqslant \sum_{k=1}^{n} \sum_{j=1}^{n} [a_{jk}| * \sum_{i=1}^{n} \sum_{k=1}^{n} b_{ki}|$$
$$\leqslant ||A|| * ||B||$$

Hence proved that $||A||_0 = \sum_{i=1}^{n} \sum_{j=1}^{n} |a_{ji}|$ is a matrix norm since it satisfies all the properties of the matrix norm.

REFRENCES:

Numerical Analysis R L Burden and J D Faires

Take Home-4

Probability problem

Problem Statement

Suppose that an object can be at any one of the (n+1) equally spaced points $x_0, x_1, ..., x_n$. When an object is at location x_i , it is equally likely to move to either x_{i-1} or x_{i+1} and cant directly move to any other location. Consider the probabilities ${p_i}_{i=0}^{n}$ that an object starting at location x_i will reach the left end point x_0 before reaching the right end point. Clearly p_o=1 and p_n=0. Also, $p_i = 0.5p_{i-1} + 0.5p_{i+1}$ for all i=1,2,...n-1.

Show that:
1.
$$\begin{bmatrix} 1 & -0.5 & 0 & 0 & 0 & \dots & 0 \\ -0.5 & 1 & -0.5 & 0 & 0 & \dots & 0 \\ 0 & -0.5 & 1 & -0.5 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & -0.5 & 1 \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ \vdots \\ p_{n-1} \end{bmatrix} = \begin{bmatrix} 0.5 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

2.Solve the system for n=100, and n=1000

3.Change the probabilities α and $1 - \alpha$ for movement to left or right. Derive the linear system similar to one in (1).

4.Repeat (2) with α =1/3.

a]Part-1:

Multplying both the matrices we get,

$$\begin{bmatrix} P_1 - 1/2P_2 \\ -1/2P_1 + P_2 - P_3 \\ -1/2P_2 + P_3 \\ -1/2P_{n-1} \\ . \\ . \\ . \\ . \\ . \\ -1/2P_{n-2} + P_{n-1} \end{bmatrix}$$

Now from the given equation $P_i = 1/2P_{i-1} + 1/2P_{i+1}$ Now by solving these simultaneous linear equations we ger,

So for i=1 $P_1 = 1/2P_0 + 1/2P_2$

$P_1 - 1/2P_2 = 1/2P_0$

Since the value of P_0 is 1

So $P_1 - 1/2P_2 = 1/2$

Then for i=2

$P_2 = 1/2P_1 + 1/2P_3$

$P_2 - 1/2P_1 - 1/2P_3 = 0$

Similarly for i=3,4.....n

It is 0 So the matrix is

$$\begin{bmatrix} 1/2 \\ 0 \\ 0 \\ 0 \\ . \\ . \\ . \\ . \\ 0 \end{bmatrix}$$

Hence proved.

part-b:

Assumptions:
Here the input is taken dynamically from the user.So the code below holds good for all the values of n i.e for n=100 and n=1000.

Algorithm:

Used Gaussian elimination to find the to solve the given system.

Basically the programm itself generates the given above matrix depending on the input given by the user (i.e n=100 or 1000 in this case)

Gaussian Elimination.

for j=0,...,n **do**
  for i=0,...,n **do**
    if $i > j$ **then**
      $c = a[i][j]/a[j][j]$
      Dividing each element by diagonal to generate a upper triangular matrix.
      for k=0,...,n **do**
        $a[i][k] = a[i][k] - c * a[j][k]$
      end for
    end if
  end for
end for
$x[n-1] = a[n-1][n]/a[n-1][n-1]$
for i=n-1,...,0 **do**
  $Initializesum \leftarrow 0.$
  for j=i+1,...,n-1 **do**
    $sum = sum + a[i][j] * x[j]$
  end for
  $x[i] = (a[i][n] - sum)/a[i][i]$
end for

CODE

```
#include<stdio.h>
#include<stdlib.h>
int input(int n)
{
  int i;
  int j;

  double **a=(double **)malloc((n+1)*sizeof(double *));
  for(i=0;i<=n;i++)
  {
    a[i]=(double *)malloc(n*sizeof(double));
  }
    //For generating the input matrix
  for(i=0;i<=n-1;i++)
  {
    for(j=0;j<=n;j++)
    {
      if(i==j)
      {
        a[i][j]=1;
      }
      else if((j==i+1 || j==i-1) && j!=n)
      {
        a[i][j]=-0.5;
      }
      else if(j!=n)
      {
        a[i][j]=0;
      }
      else if(j==n && i==0)
        a[i][j]=0.5;
      else if (j==n && i!=0)
        a[i][j]=0;
    }
  }
  int k;
```

```
  float c;
    //Gaussian elimination-for generating upper triangular matrix.
       for(j=0; j<=n; j++)
        {
        for(i=0; i<=n; i++)
        {
            if(i>j)
            {
                c=a[i][j]/a[j][j];
                for(k=0; k<=n+1; k++)
                {
                    a[i][k]=a[i][k]-c*a[j][k];
                }
            }
        }
        }
  for(i=0;i<=n-1;i++)
  {
    for(j=0;j<=n;j++)
    {
      printf("%2f  ",a[i][j]);
    }
    printf("\n");
  }

  double *x=(double *)malloc((n)*sizeof(double));

    x[n-1]=a[n-1][n]/a[n-1][n-1]; int sum;
  //Back-substitution :to find the solution i.e to get the values of probabilities in this
    case.
    for(i=n-1; i>=0; i--)
    {
        sum=0;
        for(j=i+1; j<=n-1; j++)
        {
            sum=sum+a[i][j]*x[j];
        }
        x[i]=(a[i][n]-sum)/a[i][i];
    }
    printf("\nThe solution is: \n");
    for(i=0; i<n; i++)

        printf("\np%d=%f\t",i+1,x[i]);



    return 0 ;

}
int main(){
    int n;
    int i;
    scanf("%d",&n);
input(n-1);



}
```

INPUT: n=100 and n=1000 RESULT : By solving the system for n=100
The solution is:

p1=0.500000 p2=0.333333 p3=0.250000 p4=0.200000 p5=0.166667 p6=0.142857 p7=0.125000 p8=0.111111 p9=0.100000 p10=0.090909 p11=0.083333 p12=0.076923 p13=0.071429 p14=0.066667 p15=0.062500 p16=0.058824 p17=0.055556 p18=0.052632 p19=0.050000 p20=0.047619 p21=0.045455 p22=0.043478 p23=0.041667 p24=0.040000 p25=0.038462 p26=0.037037 p27=0.035714 p28=0.034483 p29=0.033333

p30=0.032258 p31=0.031250 p32=0.030303 p33=0.029412 p34=0.028571 p35=0.027778 p36=0.027027
p37=0.026316 p38=0.025641 p39=0.025000 p40=0.024390 p41=0.023810 p42=0.023256 p43=0.022727
p44=0.022222 p45=0.021739 p46=0.021277 p47=0.020833 p48=0.020408 p49=0.020000 p50=0.019608
p51=0.019231 p52=0.018868 p53=0.018519 p54=0.018182 p55=0.017857 p56=0.017544 p57=0.017241
p58=0.016949 p59=0.016667 p60=0.016393 p61=0.016129 p62=0.015873 p63=0.015625 p64=0.015385
p65=0.015152 p66=0.014925 p67=0.014706 p68=0.014493 p69=0.014286 p70=0.014085 p71=0.013889
p72=0.013699 p73=0.013514 p74=0.013333 p75=0.013158 p76=0.012987 p77=0.012821 p78=0.012658
p79=0.012500 p80=0.012346 p81=0.012195 p82=0.012048 p83=0.011905 p84=0.011765 p85=0.011628
p86=0.011494 p87=0.011364 p88=0.011236 p89=0.011111 p90=0.010989 p91=0.010870 p92=0.010753
p93=0.010638 p94=0.010526 p95=0.010417 p96=0.010309 p97=0.010204 p98=0.010101 p99=0.010000
By solving the system for n=1000

The solution is:
p1=0.500000 p2=0.333333 p3=0.250000 p4=0.200000 p5=0.166667 p6=0.142857 p7=0.125000 p8=0.111111
p9=0.100000 p10=0.090909 p11=0.083333 p12=0.076923 p13=0.071429 p14=0.066667 p15=0.062500
p16=0.058824 p17=0.055556 p18=0.052632 p19=0.050000 p20=0.047619 p21=0.045455 p22=0.043478
p23=0.041667 p24=0.040000 p25=0.038462 p26=0.037037 p27=0.035714 p28=0.034483 p29=0.033333
p30=0.032258 p31=0.031250 p32=0.030303 p33=0.029412 p34=0.028571 p35=0.027778 p36=0.027027
p37=0.026316 p38=0.025641 p39=0.025000 p40=0.024390 p41=0.023810 p42=0.023256 p43=0.022727
p44=0.022222 p45=0.021739 p46=0.021277 p47=0.020833 p48=0.020408 p49=0.020000 p50=0.019608
p51=0.019231 p52=0.018868 p53=0.018519 p54=0.018182 p55=0.017857 p56=0.017544 p57=0.017241
p58=0.016949 p59=0.016667 p60=0.016393 p61=0.016129 p62=0.015873 p63=0.015625 p64=0.015385
p65=0.015152 p66=0.014925 p67=0.014706 p68=0.014493 p69=0.014286 p70=0.014085 p71=0.013889
p72=0.013699 p73=0.013514 p74=0.013333 p75=0.013158 p76=0.012987 p77=0.012821 p78=0.012658
p79=0.012500 p80=0.012346 p81=0.012195 p82=0.012048 p83=0.011905 p84=0.011765 p85=0.011628
p86=0.011494 p87=0.011364 p88=0.011236 p89=0.011111 p90=0.010989 p91=0.010870 p92=0.010753
p93=0.010638 p94=0.010526 p95=0.010417 p96=0.010309 p97=0.010204 p98=0.010101 p99=0.010000
p100=0.009901 p101=0.009804 p102=0.009709 p103=0.009615 p104=0.009524 p105=0.009434
p106=0.009346 p107=0.009259 p108=0.009174 p109=0.009091 p110=0.009009 p111=0.008929
p112=0.008850 p113=0.008772 p114=0.008696 p115=0.008621 p116=0.008547 p117=0.008475
p118=0.008403 p119=0.008333 p120=0.008264 p121=0.008197 p122=0.008130 p123=0.008065
p124=0.008000 p125=0.007937 p126=0.007874 p127=0.007813 p128=0.007752 p129=0.007692
p130=0.007634 p131=0.007576 p132=0.007519 p133=0.007463 p134=0.007407 p135=0.007353
p136=0.007299 p137=0.007246 p138=0.007194 p139=0.007143 p140=0.007092 p141=0.007042
p142=0.006993 p143=0.006944 p144=0.006897 p145=0.006849 p146=0.006803 p147=0.006757
p148=0.006711 p149=0.006667 p150=0.006623 p151=0.006579 p152=0.006536 p153=0.006494
p154=0.006452 p155=0.006410 p156=0.006369 p157=0.006329 p158=0.006289 p159=0.006250
p160=0.006211 p161=0.006173 p162=0.006135 p163=0.006098 p164=0.006061 p165=0.006024
p166=0.005988 p167=0.005952 p168=0.005917 p169=0.005882 p170=0.005848 p171=0.005814
p172=0.005780 p173=0.005747 p174=0.005714 p175=0.005682 p176=0.005650 p177=0.005618
p178=0.005587 p179=0.005556 p180=0.005525 p181=0.005495 p182=0.005464 p183=0.005435
p184=0.005405 p185=0.005376 p186=0.005348 p187=0.005319 p188=0.005291 p189=0.005263
p190=0.005236 p191=0.005208 p192=0.005181 p193=0.005155 p194=0.005128 p195=0.005102
p196=0.005076 p197=0.005051 p198=0.005025 p199=0.005000 p200=0.004975 p201=0.004951
p202=0.004926 p203=0.004902 p204=0.004878 p205=0.004854 p206=0.004831 p207=0.004808
p208=0.004785 p209=0.004762 p210=0.004739 p211=0.004717 p212=0.004695 p213=0.004673
p214=0.004651 p215=0.004630 p216=0.004608 p217=0.004587 p218=0.004566 p219=0.004545
p220=0.004525 p221=0.004505 p222=0.004484 p223=0.004464 p224=0.004444 p225=0.004425
p226=0.004405 p227=0.004386 p228=0.004367 p229=0.004348 p230=0.004329 p231=0.004310
p232=0.004292 p233=0.004274 p234=0.004255 p235=0.004237 p236=0.004219 p237=0.004202
p238=0.004184 p239=0.004167 p240=0.004149 p241=0.004132 p242=0.004115 p243=0.004098
p244=0.004082 p245=0.004065 p246=0.004049 p247=0.004032 p248=0.004016 p249=0.004000
p250=0.003984 p251=0.003968 p252=0.003953 p253=0.003937 p254=0.003922 p255=0.003906
p256=0.003891 p257=0.003876 p258=0.003861 p259=0.003846 p260=0.003831 p261=0.003817

p262=0.003802 p263=0.003788 p264=0.003774 p265=0.003759 p266=0.003745 p267=0.003731
p268=0.003717 p269=0.003704 p270=0.003690 p271=0.003676 p272=0.003663 p273=0.003650
p274=0.003636 p275=0.003623 p276=0.003610 p277=0.003597 p278=0.003584 p279=0.003571
p280=0.003559 p281=0.003546 p282=0.003534 p283=0.003521 p284=0.003509 p285=0.003496
p286=0.003484 p287=0.003472 p288=0.003460 p289=0.003448 p290=0.003436 p291=0.003425
p292=0.003413 p293=0.003401 p294=0.003390 p295=0.003378 p296=0.003367 p297=0.003356
p298=0.003344 p299=0.003333 p300=0.003322 p301=0.003311 p302=0.003300 p303=0.003289
p304=0.003279 p305=0.003268 p306=0.003257 p307=0.003247 p308=0.003236 p309=0.003226
p310=0.003215 p311=0.003205 p312=0.003195 p313=0.003185 p314=0.003175 p315=0.003165
p316=0.003155 p317=0.003145 p318=0.003135 p319=0.003125 p320=0.003115 p321=0.003106
p322=0.003096 p323=0.003086 p324=0.003077 p325=0.003067 p326=0.003058 p327=0.003049
p328=0.003039 p329=0.003030 p330=0.003021 p331=0.003012 p332=0.003003 p333=0.002994
p334=0.002985 p335=0.002976 p336=0.002967 p337=0.002959 p338=0.002950 p339=0.002941
p340=0.002933 p341=0.002924 p342=0.002915 p343=0.002907 p344=0.002899 p345=0.002890
p346=0.002882 p347=0.002874 p348=0.002865 p349=0.002857 p350=0.002849 p351=0.002841
p352=0.002833 p353=0.002825 p354=0.002817 p355=0.002809 p356=0.002801 p357=0.002793
p358=0.002785 p359=0.002778 p360=0.002770 p361=0.002762 p362=0.002755 p363=0.002747
p364=0.002740 p365=0.002732 p366=0.002725 p367=0.002717 p368=0.002710 p369=0.002703
p370=0.002695 p371=0.002688 p372=0.002681 p373=0.002674 p374=0.002667 p375=0.002660
p376=0.002652 p377=0.002645 p378=0.002638 p379=0.002632 p380=0.002625 p381=0.002618
p382=0.002611 p383=0.002604 p384=0.002597 p385=0.002591 p386=0.002584 p387=0.002577
p388=0.002571 p389=0.002564 p390=0.002557 p391=0.002551 p392=0.002544 p393=0.002538
p394=0.002532 p395=0.002525 p396=0.002519 p397=0.002513 p398=0.002506 p399=0.002500
p400=0.002494 p401=0.002488 p402=0.002481 p403=0.002475 p404=0.002469 p405=0.002463
p406=0.002457 p407=0.002451 p408=0.002445 p409=0.002439 p410=0.002433 p411=0.002427
p412=0.002421 p413=0.002415 p414=0.002410 p415=0.002404 p416=0.002398 p417=0.002392
p418=0.002387 p419=0.002381 p420=0.002375 p421=0.002370 p422=0.002364 p423=0.002358
p424=0.002353 p425=0.002347 p426=0.002342 p427=0.002336 p428=0.002331 p429=0.002326
p430=0.002320 p431=0.002315 p432=0.002309 p433=0.002304 p434=0.002299 p435=0.002294
p436=0.002288 p437=0.002283 p438=0.002278 p439=0.002273 p440=0.002268 p441=0.002262
p442=0.002257 p443=0.002252 p444=0.002247 p445=0.002242 p446=0.002237 p447=0.002232
p448=0.002227 p449=0.002222 p450=0.002217 p451=0.002212 p452=0.002207 p453=0.002203
p454=0.002198 p455=0.002193 p456=0.002188 p457=0.002183 p458=0.002179 p459=0.002174
p460=0.002169 p461=0.002164 p462=0.002160 p463=0.002155 p464=0.002150 p465=0.002146
p466=0.002141 p467=0.002137 p468=0.002132 p469=0.002128 p470=0.002123 p471=0.002119
p472=0.002114 p473=0.002110 p474=0.002105 p475=0.002101 p476=0.002096 p477=0.002092
p478=0.002088 p479=0.002083 p480=0.002079 p481=0.002075 p482=0.002070 p483=0.002066
p484=0.002062 p485=0.002058 p486=0.002053 p487=0.002049 p488=0.002045 p489=0.002041
p490=0.002037 p491=0.002032 p492=0.002028 p493=0.002024 p494=0.002020 p495=0.002016
p496=0.002012 p497=0.002008 p498=0.002004 p499=0.002000 p500=0.001996 p501=0.001992
p502=0.001988 p503=0.001984 p504=0.001980 p505=0.001976 p506=0.001972 p507=0.001968
p508=0.001965 p509=0.001961 p510=0.001957 p511=0.001953 p512=0.001949 p513=0.001945
p514=0.001942 p515=0.001938 p516=0.001934 p517=0.001930 p518=0.001927 p519=0.001923
p520=0.001919 p521=0.001916 p522=0.001912 p523=0.001908 p524=0.001905 p525=0.001901
p526=0.001897 p527=0.001894 p528=0.001890 p529=0.001887 p530=0.001883 p531=0.001880
p532=0.001876 p533=0.001873 p534=0.001869 p535=0.001866 p536=0.001862 p537=0.001859
p538=0.001855 p539=0.001852 p540=0.001848 p541=0.001845 p542=0.001842 p543=0.001838
p544=0.001835 p545=0.001831 p546=0.001828 p547=0.001825 p548=0.001821 p549=0.001818
p550=0.001815 p551=0.001812 p552=0.001808 p553=0.001805 p554=0.001802 p555=0.001799
p556=0.001795 p557=0.001792 p558=0.001789 p559=0.001786 p560=0.001782 p561=0.001779
p562=0.001776 p563=0.001773 p564=0.001770 p565=0.001767 p566=0.001764 p567=0.001761
p568=0.001757 p569=0.001754 p570=0.001751 p571=0.001748 p572=0.001745 p573=0.001742
p574=0.001739 p575=0.001736 p576=0.001733 p577=0.001730 p578=0.001727 p579=0.001724
p580=0.001721 p581=0.001718 p582=0.001715 p583=0.001712 p584=0.001709 p585=0.001706

p586=0.001704 p587=0.001701 p588=0.001698 p589=0.001695 p590=0.001692 p591=0.001689
p592=0.001686 p593=0.001683 p594=0.001681 p595=0.001678 p596=0.001675 p597=0.001672
p598=0.001669 p599=0.001667 p600=0.001664 p601=0.001661 p602=0.001658 p603=0.001656
p604=0.001653 p605=0.001650 p606=0.001647 p607=0.001645 p608=0.001642 p609=0.001639
p610=0.001637 p611=0.001634 p612=0.001631 p613=0.001629 p614=0.001626 p615=0.001623
p616=0.001621 p617=0.001618 p618=0.001615 p619=0.001613 p620=0.001610 p621=0.001608
p622=0.001605 p623=0.001603 p624=0.001600 p625=0.001597 p626=0.001595 p627=0.001592
p628=0.001590 p629=0.001587 p630=0.001585 p631=0.001582 p632=0.001580 p633=0.001577
p634=0.001575 p635=0.001572 p636=0.001570 p637=0.001567 p638=0.001565 p639=0.001562
p640=0.001560 p641=0.001558 p642=0.001555 p643=0.001553 p644=0.001550 p645=0.001548
p646=0.001546 p647=0.001543 p648=0.001541 p649=0.001538 p650=0.001536 p651=0.001534
p652=0.001531 p653=0.001529 p654=0.001527 p655=0.001524 p656=0.001522 p657=0.001520
p658=0.001517 p659=0.001515 p660=0.001513 p661=0.001511 p662=0.001508 p663=0.001506
p664=0.001504 p665=0.001501 p666=0.001499 p667=0.001497 p668=0.001495 p669=0.001493
p670=0.001490 p671=0.001488 p672=0.001486 p673=0.001484 p674=0.001481 p675=0.001479
p676=0.001477 p677=0.001475 p678=0.001473 p679=0.001471 p680=0.001468 p681=0.001466
p682=0.001464 p683=0.001462 p684=0.001460 p685=0.001458 p686=0.001456 p687=0.001453
p688=0.001451 p689=0.001449 p690=0.001447 p691=0.001445 p692=0.001443 p693=0.001441
p694=0.001439 p695=0.001437 p696=0.001435 p697=0.001433 p698=0.001431 p699=0.001429
p700=0.001427 p701=0.001424 p702=0.001422 p703=0.001420 p704=0.001418 p705=0.001416
p706=0.001414 p707=0.001412 p708=0.001410 p709=0.001408 p710=0.001406 p711=0.001404
p712=0.001402 p713=0.001401 p714=0.001399 p715=0.001397 p716=0.001395 p717=0.001393
p718=0.001391 p719=0.001389 p720=0.001387 p721=0.001385 p722=0.001383 p723=0.001381
p724=0.001379 p725=0.001377 p726=0.001375 p727=0.001374 p728=0.001372 p729=0.001370
p730=0.001368 p731=0.001366 p732=0.001364 p733=0.001362 p734=0.001361 p735=0.001359
p736=0.001357 p737=0.001355 p738=0.001353 p739=0.001351 p740=0.001350 p741=0.001348
p742=0.001346 p743=0.001344 p744=0.001342 p745=0.001340 p746=0.001339 p747=0.001337
p748=0.001335 p749=0.001333 p750=0.001332 p751=0.001330 p752=0.001328 p753=0.001326
p754=0.001324 p755=0.001323 p756=0.001321 p757=0.001319 p758=0.001317 p759=0.001316
p760=0.001314 p761=0.001312 p762=0.001311 p763=0.001309 p764=0.001307 p765=0.001305
p766=0.001304 p767=0.001302 p768=0.001300 p769=0.001299 p770=0.001297 p771=0.001295
p772=0.001294 p773=0.001292 p774=0.001290 p775=0.001289 p776=0.001287 p777=0.001285
p778=0.001284 p779=0.001282 p780=0.001280 p781=0.001279 p782=0.001277 p783=0.001275
p784=0.001274 p785=0.001272 p786=0.001271 p787=0.001269 p788=0.001267 p789=0.001266
p790=0.001264 p791=0.001263 p792=0.001261 p793=0.001259 p794=0.001258 p795=0.001256
p796=0.001255 p797=0.001253 p798=0.001252 p799=0.001250 p800=0.001248 p801=0.001247
p802=0.001245 p803=0.001244 p804=0.001242 p805=0.001241 p806=0.001239 p807=0.001238
p808=0.001236 p809=0.001235 p810=0.001233 p811=0.001232 p812=0.001230 p813=0.001228
p814=0.001227 p815=0.001225 p816=0.001224 p817=0.001222 p818=0.001221 p819=0.001219
p820=0.001218 p821=0.001217 p822=0.001215 p823=0.001214 p824=0.001212 p825=0.001211
p826=0.001209 p827=0.001208 p828=0.001206 p829=0.001205 p830=0.001203 p831=0.001202
p832=0.001200 p833=0.001199 p834=0.001198 p835=0.001196 p836=0.001195 p837=0.001193
p838=0.001192 p839=0.001190 p840=0.001189 p841=0.001188 p842=0.001186 p843=0.001185
p844=0.001183 p845=0.001182 p846=0.001181 p847=0.001179 p848=0.001178 p849=0.001176
p850=0.001175 p851=0.001174 p852=0.001172 p853=0.001171 p854=0.001170 p855=0.001168
p856=0.001167 p857=0.001165 p858=0.001164 p859=0.001163 p860=0.001161 p861=0.001160
p862=0.001159 p863=0.001157 p864=0.001156 p865=0.001155 p866=0.001153 p867=0.001152
p868=0.001151 p869=0.001149 p870=0.001148 p871=0.001147 p872=0.001145 p873=0.001144
p874=0.001143 p875=0.001142 p876=0.001140 p877=0.001139 p878=0.001138 p879=0.001136
p880=0.001135 p881=0.001134 p882=0.001132 p883=0.001131 p884=0.001130 p885=0.001129
p886=0.001127 p887=0.001126 p888=0.001125 p889=0.001124 p890=0.001122 p891=0.001121
p892=0.001120 p893=0.001119 p894=0.001117 p895=0.001116 p896=0.001115 p897=0.001114
p898=0.001112 p899=0.001111 p900=0.001110 p901=0.001109 p902=0.001107 p903=0.001106
p904=0.001105 p905=0.001104 p906=0.001103 p907=0.001101 p908=0.001100 p909=0.001099

p910=0.001098	p911=0.001096	p912=0.001095	p913=0.001094	p914=0.001093	p915=0.001092
p916=0.001090	p917=0.001089	p918=0.001088	p919=0.001087	p920=0.001086	p921=0.001085
p922=0.001083	p923=0.001082	p924=0.001081	p925=0.001080	p926=0.001079	p927=0.001078
p928=0.001076	p929=0.001075	p930=0.001074	p931=0.001073	p932=0.001072	p933=0.001071
p934=0.001069	p935=0.001068	p936=0.001067	p937=0.001066	p938=0.001065	p939=0.001064
p940=0.001063	p941=0.001062	p942=0.001060	p943=0.001059	p944=0.001058	p945=0.001057
p946=0.001056	p947=0.001055	p948=0.001054	p949=0.001053	p950=0.001051	p951=0.001050
p952=0.001049	p953=0.001048	p954=0.001047	p955=0.001046	p956=0.001045	p957=0.001044
p958=0.001043	p959=0.001042	p960=0.001041	p961=0.001039	p962=0.001038	p963=0.001037
p964=0.001036	p965=0.001035	p966=0.001034	p967=0.001033	p968=0.001032	p969=0.001031
p970=0.001030	p971=0.001029	p972=0.001028	p973=0.001027	p974=0.001026	p975=0.001025
p976=0.001024	p977=0.001022	p978=0.001021	p979=0.001020	p980=0.001019	p981=0.001018
p982=0.001017	p983=0.001016	p984=0.001015	p985=0.001014	p986=0.001013	p987=0.001012
p988=0.001011	p989=0.001010	p990=0.001009	p991=0.001008	p992=0.001007	p993=0.001006
p994=0.001005	p995=0.001004	p996=0.001003	p997=0.001002	p998=0.001001	p999=0.001000

part-c

From changing the probabilities to

$$\alpha and (1-\alpha) \tag{1}$$

for movement to the left or right.

$$P_i = \alpha P_{i-1} + (1-\alpha) P_{i+1} \tag{2}$$

Solving the equation For i=1

$$P_1 = \alpha P_0 + (1-\alpha) P_2 \tag{3}$$

For i=2

$$P_2 = \alpha P_1 + (1-\alpha) P_3 \tag{4}$$

For i=3

$$P_3 = \alpha P_2 + (1-\alpha) P_4 \tag{5}$$

For i=4

$$P_4 = \alpha P_3 + (1-\alpha) P_5 \tag{6}$$

For i=n-1

$$P_{n-1} = \alpha P_{n-2} + (1-\alpha) P_n \tag{7}$$

Since the equations are nothing but linear transformations.Hence writing the above equations in terms of matrix

$$\begin{bmatrix} 1 & -(1-\alpha) & 0 & \dots & 0 \\ \alpha & -1 & (1-\alpha) & \ddots & 0 \\ 0 & \alpha & \ddots & 1 & (1-\alpha) \\ 0 & 0 & \alpha & \ddots -1 & \\ . & . & . & . & . \\ . & . & . & . & . \\ 0 & 0 & \alpha & \dots & (1-\alpha) \end{bmatrix}$$

part-d

Assumptions:
Here the input is taken dynamically from the user.So the code below holds good for all the values of n i.e for n=100 and n=1000.

Used Gaussian elimination to find the to solve the given system.
Basically the programm itself generates the matrix with the value of alpha as 0.333 depending on the input given by the user (i.e n=100 or 1000 in this case)

Gaussian Elimination.

for j=0,...,n **do**
  for i=0,...,n **do**
    if $i > j$ **then**
      $c = a[i][j]/a[j][j]$
      Dividing each element by diagonal to generate a upper triangular matrix.
      for k=0,...,n **do**
        $a[i][k] = a[i][k] - c * a[j][k]$
      end for
    end if
  end for
end for
$x[n-1] = a[n-1][n]/a[n-1][n-1]$

for i=n-1,...,0 **do**
  $Initialize sum \leftarrow 0.$
  for j=i+1,...,n-1 **do**
    $sum = sum + a[i][j] * x[j]$
  end for
  $x[i] = (a[i][n] - sum)/a[i][i]$
end for

CODE

```
#include<stdio.h>


#include<stdlib.h>




int input(int n)
{
  int i;
  int j;
  //int *a;
  double **a=(double **)malloc((n+1)*sizeof(double *));
  for(i=0;i<=n;i++)
  {
    a[i]=(double *)malloc(n*sizeof(double));
  }
     //For generating the input matrix
  for(i=0;i<=n-1;i++)
  {
    for(j=0;j<=n;j++)
    {
      if(i==j)
      {
        a[i][j]=1;
      }
      else if((j==i+1) && j!=n)
      {
        a[i][j]=-0.6666;
      }
      else if(j==i-1 && j!=n)
        a[i][j]=-0.3333;
      else if(j!=n)
      {
        a[i][j]=0;
      }
      else if(j==n && i==0)
        a[i][j]=0.3333;
      else if (j==n && i!=0)
        a[i][j]=0;
    }
  }
  for(i=0;i<=n;i++)
  {
    for(j=0;j<=n;j++)
    {
      printf("%f\t",a[i][j]);
    }
printf("\n");
  }

  int k;
  float c;
    //Gausian elimination:To generate a upper traingular matrix.
       for(j=0; j<=n; j++)
```

```c
        {
        for(i=0; i<=n; i++)
        {
            if(i>j)
            {
                c=a[i][j]/a[j][j];
                for(k=0; k<=n+1; k++)
                {
                    a[i][k]=a[i][k]-c*a[j][k];
                }
            }
        }
        }
  for(i=0;i<=n-1;i++)
  {
    for(j=0;j<=n;j++)
    {
      printf("%2f  ",a[i][j]);
    }
    printf("\n");
  }

  double *x=(double *)malloc((n)*sizeof(double));

    x[n-1]=a[n-1][n]/a[n-1][n-1]; int sum;
  /Back-substitution :to find the solution i.e to get the values of probabilities in this case.
    for(i=n-1; i>=0; i--)
    {
        sum=0;
        for(j=i+1; j<=n-1; j++)
        {
            sum=sum+a[i][j]*x[j];
        }
        x[i]=(a[i][n]-sum)/a[i][i];
    }
    printf("\nThe solution is: \n");
    for(i=0; i<n; i++)

        printf("\np%d=%f\t",i+1,x[i]);



    return 0 ;

}
int main(){
    int n;
    int i;
    scanf("%d",&n);
input(n-1);

}
```

INPUT: INPUT: n=100 and n=1000 RESULT : By solving the system for n=100
The solution is:
p1=0.333300 p2=0.142820 p3=0.066636 p4=0.032236 p5=0.015858 p6=0.007864 p7=0.003916 p8=0.001953 p9=0.000975 p10=0.000487 p11=0.000244 p12=0.000122 p13=0.000061 p14=0.000030 p15=0.000015 p16=0.000008 p17=0.000004 p18=0.000002 p19=0.000001 p20=0.000000 p21=0.000000 p22=0.000000 p23=0.000000 p24=0.000000 p25=0.000000 p26=0.000000 p27=0.000000 p28=0.000000 p29=0.000000 p30=0.000000 p31=0.000000 p32=0.000000 p33=0.000000 p34=0.000000 p35=0.000000 p36=0.000000 p37=0.000000 p38=0.000000 p39=0.000000 p40=0.000000 p41=0.000000 p42=0.000000 p43=0.000000 p44=0.000000 p45=0.000000 p46=0.000000 p47=0.000000 p48=0.000000 p49=0.000000 p50=0.000000 p51=0.000000 p52=0.000000 p53=0.000000 p54=0.000000 p55=0.000000 p56=0.000000 p57=0.000000 p58=0.000000 p59=0.000000 p60=0.000000 p61=0.000000 p62=0.000000 p63=0.000000 p64=0.000000

p65=0.000000 p66=0.000000 p67=0.000000 p68=0.000000 p69=0.000000 p70=0.000000 p71=0.000000 p72=0.000000 p73=0.000000 p74=0.000000 p75=0.000000 p76=0.000000 p77=0.000000 p78=0.000000 p79=0.000000 p80=0.000000 p81=0.000000 p82=0.000000 p83=0.000000 p84=0.000000 p85=0.000000 p86=0.000000 p87=0.000000 p88=0.000000 p89=0.000000 p90=0.000000 p91=0.000000 p92=0.000000 p93=0.000000 p94=0.000000 p95=0.000000 p96=0.000000 p97=0.000000 p98=0.000000 p99=0.000000

For n=1000

The solution is:
p1= 0.333300 p2= 0.142820 p3= 0.066636 p4= 0.032236 p5= 0.015858 p6= 0.007864 p7= 0.003916 p8= 0.001953 p9= 0.000975 p10= 0.000487 p11= 0.000244 p12= 0.000122 p13= 0.000061 p14= 0.000030 p15= 0.000015 p16= 0.000008 p17= 0.000004 p18= 0.000002 p19= 0.000001 p20= 0.000000 p21= 0.000000 p22= 0.000000 p23= 0.000000 p24= 0.000000 p25= 0.000000 p26= 0.000000 p27= 0.000000 p28= 0.000000 p29= 0.000000 p30= 0.000000 p31= 0.000000 p32= 0.000000 p33= 0.000000 p34= 0.000000 p35= 0.000000 p36= 0.000000 p37= 0.000000 p38= 0.000000 p39= 0.000000 p40= 0.000000 p41= 0.000000 p42= 0.000000 p43= 0.000000 p44= 0.000000 p45= 0.000000 p46= 0.000000 p47= 0.000000 p48= 0.000000 p49= 0.000000 p50= 0.000000 p51= 0.000000 p52= 0.000000 p53= 0.000000 p54= 0.000000 p55= 0.000000 p56= 0.000000 p57= 0.000000 p58= 0.000000 p59= 0.000000 p60= 0.000000 p61= 0.000000 p62= 0.000000 p63= 0.000000 p64= 0.000000 p65= 0.000000 p66= 0.000000 p67= 0.000000 p68= 0.000000 p69= 0.000000 p70= 0.000000 p71= 0.000000 p72= 0.000000 p73= 0.000000 p74= 0.000000 p75= 0.000000 p76= 0.000000 p77= 0.000000 p78= 0.000000 p79= 0.000000 p80= 0.000000 p81= 0.000000 p82= 0.000000 p83= 0.000000 p84= 0.000000 p85= 0.000000 p86= 0.000000 p87= 0.000000 p88= 0.000000 p89= 0.000000 p90= 0.000000 p91= 0.000000 p92= 0.000000 p93= 0.000000 p94= 0.000000 p95= 0.000000 p96= 0.000000 p97= 0.000000 p98= 0.000000 p99= 0.000000 p100= 0.000000 p101= 0.000000 p102= 0.000000 p103= 0.000000 p104= 0.000000 p105= 0.000000 p106= 0.000000 p107= 0.000000 p108= 0.000000 p109= 0.000000 p110= 0.000000 p111= 0.000000 p112= 0.000000 p113= 0.000000 p114= 0.000000 p115= 0.000000 p116= 0.000000 p117= 0.000000 p118= 0.000000 p119= 0.000000 p120= 0.000000 p121= 0.000000 p122= 0.000000 p123= 0.000000 p124= 0.000000 p125= 0.000000 p126= 0.000000 p127= 0.000000 p128= 0.000000 p129= 0.000000 p130= 0.000000 p131= 0.000000 p132= 0.000000 p133= 0.000000 p134= 0.000000 p135= 0.000000 p136= 0.000000 p137= 0.000000 p138= 0.000000 p139= 0.000000 p140= 0.000000 p141= 0.000000 p142= 0.000000 p143= 0.000000 p144= 0.000000 p145= 0.000000 p146= 0.000000 p147= 0.000000 p148= 0.000000 p149= 0.000000 p150= 0.000000 p151= 0.000000 p152= 0.000000 p153= 0.000000 p154= 0.000000 p155= 0.000000 p156= 0.000000 p157= 0.000000 p158= 0.000000 p159= 0.000000 p160= 0.000000 p161= 0.000000 p162= 0.000000 p163= 0.000000 p164= 0.000000 p165= 0.000000 p166= 0.000000 p167= 0.000000 p168= 0.000000 p169= 0.000000 p170= 0.000000 p171= 0.000000 p172= 0.000000 p173= 0.000000 p174= 0.000000 p175= 0.000000 p176= 0.000000 p177= 0.000000 p178= 0.000000 p179= 0.000000 p180= 0.000000 p181= 0.000000 p182= 0.000000 p183= 0.000000 p184= 0.000000 p185= 0.000000 p186= 0.000000 p187= 0.000000 p188= 0.000000 p189= 0.000000 p190= 0.000000 p191= 0.000000 p192= 0.000000 p193= 0.000000 p194= 0.000000 p195= 0.000000 p196= 0.000000 p197= 0.000000 p198= 0.000000 p199= 0.000000 p200= 0.000000 p201= 0.000000 p202= 0.000000 p203= 0.000000 p204= 0.000000 p205= 0.000000 p206= 0.000000 p207= 0.000000 p208= 0.000000 p209= 0.000000 p210= 0.000000 p211= 0.000000 p212= 0.000000 p213= 0.000000 p214= 0.000000 p215= 0.000000 p216= 0.000000 p217= 0.000000 p218= 0.000000 p219= 0.000000 p220= 0.000000 p221= 0.000000 p222= 0.000000 p223= 0.000000 p224= 0.000000 p225= 0.000000 p226= 0.000000 p227= 0.000000 p228= 0.000000 p229= 0.000000 p230= 0.000000 p231= 0.000000 p232= 0.000000 p233= 0.000000 p234= 0.000000 p235= 0.000000 p236= 0.000000 p237= 0.000000 p238= 0.000000 p239= 0.000000 p240= 0.000000 p241= 0.000000 p242= 0.000000 p243= 0.000000 p244= 0.000000 p245= 0.000000 p246= 0.000000 p247= 0.000000 p248= 0.000000 p249= 0.000000 p250= 0.000000 p251= 0.000000 p252= 0.000000 p253= 0.000000 p254= 0.000000 p255= 0.000000 p256= 0.000000 p257= 0.000000 p258= 0.000000 p259= 0.000000 p260= 0.000000 p261= 0.000000 p262= 0.000000 p263= 0.000000 p264= 0.000000 p265= 0.000000 p266= 0.000000 p267= 0.000000 p268= 0.000000 p269= 0.000000 p270= 0.000000 p271= 0.000000 p272= 0.000000 p273= 0.000000 p274= 0.000000 p275= 0.000000 p276= 0.000000 p277= 0.000000 p278= 0.000000 p279= 0.000000 p280= 0.000000 p281= 0.000000 p282= 0.000000 p283= 0.000000 p284= 0.000000 p285= 0.000000 p286=

0.000000 p287= 0.000000 p288= 0.000000 p289= 0.000000 p290= 0.000000 p291= 0.000000 p292=
0.000000 p293= 0.000000 p294= 0.000000 p295= 0.000000 p296= 0.000000 p297= 0.000000 p298=
0.000000 p299= 0.000000 p300= 0.000000 p301= 0.000000 p302= 0.000000 p303= 0.000000 p304=
0.000000 p305= 0.000000 p306= 0.000000 p307= 0.000000 p308= 0.000000 p309= 0.000000 p310=
0.000000 p311= 0.000000 p312= 0.000000 p313= 0.000000 p314= 0.000000 p315= 0.000000 p316=
0.000000 p317= 0.000000 p318= 0.000000 p319= 0.000000 p320= 0.000000 p321= 0.000000 p322=
0.000000 p323= 0.000000 p324= 0.000000 p325= 0.000000 p326= 0.000000 p327= 0.000000 p328=
0.000000 p329= 0.000000 p330= 0.000000 p331= 0.000000 p332= 0.000000 p333= 0.000000 p334=
0.000000 p335= 0.000000 p336= 0.000000 p337= 0.000000 p338= 0.000000 p339= 0.000000 p340=
0.000000 p341= 0.000000 p342= 0.000000 p343= 0.000000 p344= 0.000000 p345= 0.000000 p346=
0.000000 p347= 0.000000 p348= 0.000000 p349= 0.000000 p350= 0.000000 p351= 0.000000 p352=
0.000000 p353= 0.000000 p354= 0.000000 p355= 0.000000 p356= 0.000000 p357= 0.000000 p358=
0.000000 p359= 0.000000 p360= 0.000000 p361= 0.000000 p362= 0.000000 p363= 0.000000 p364=
0.000000 p365= 0.000000 p366= 0.000000 p367= 0.000000 p368= 0.000000 p369= 0.000000 p370=
0.000000 p371= 0.000000 p372= 0.000000 p373= 0.000000 p374= 0.000000 p375= 0.000000 p376=
0.000000 p377= 0.000000 p378= 0.000000 p379= 0.000000 p380= 0.000000 p381= 0.000000 p382=
0.000000 p383= 0.000000 p384= 0.000000 p385= 0.000000 p386= 0.000000 p387= 0.000000 p388=
0.000000 p389= 0.000000 p390= 0.000000 p391= 0.000000 p392= 0.000000 p393= 0.000000 p394=
0.000000 p395= 0.000000 p396= 0.000000 p397= 0.000000 p398= 0.000000 p399= 0.000000 p400=
0.000000 p401= 0.000000 p402= 0.000000 p403= 0.000000 p404= 0.000000 p405= 0.000000 p406=
0.000000 p407= 0.000000 p408= 0.000000 p409= 0.000000 p410= 0.000000 p411= 0.000000 p412=
0.000000 p413= 0.000000 p414= 0.000000 p415= 0.000000 p416= 0.000000 p417= 0.000000 p418=
0.000000 p419= 0.000000 p420= 0.000000 p421= 0.000000 p422= 0.000000 p423= 0.000000 p424=
0.000000 p425= 0.000000 p426= 0.000000 p427= 0.000000 p428= 0.000000 p429= 0.000000 p430=
0.000000 p431= 0.000000 p432= 0.000000 p433= 0.000000 p434= 0.000000 p435= 0.000000 p436=
0.000000 p437= 0.000000 p438= 0.000000 p439= 0.000000 p440= 0.000000 p441= 0.000000 p442=
0.000000 p443= 0.000000 p444= 0.000000 p445= 0.000000 p446= 0.000000 p447= 0.000000 p448=
0.000000 p449= 0.000000 p450= 0.000000 p451= 0.000000 p452= 0.000000 p453= 0.000000 p454=
0.000000 p455= 0.000000 p456= 0.000000 p457= 0.000000 p458= 0.000000 p459= 0.000000 p460=
0.000000 p461= 0.000000 p462= 0.000000 p463= 0.000000 p464= 0.000000 p465= 0.000000 p466=
0.000000 p467= 0.000000 p468= 0.000000 p469= 0.000000 p470= 0.000000 p471= 0.000000 p472=
0.000000 p473= 0.000000 p474= 0.000000 p475= 0.000000 p476= 0.000000 p477= 0.000000 p478=
0.000000 p479= 0.000000 p480= 0.000000 p481= 0.000000 p482= 0.000000 p483= 0.000000 p484=
0.000000 p485= 0.000000 p486= 0.000000 p487= 0.000000 p488= 0.000000 p489= 0.000000 p490=
0.000000 p491= 0.000000 p492= 0.000000 p493= 0.000000 p494= 0.000000 p495= 0.000000 p496=
0.000000 p497= 0.000000 p498= 0.000000 p499= 0.000000 p500= 0.000000 p501= 0.000000 p502=
0.000000 p503= 0.000000 p504= 0.000000 p505= 0.000000 p506= 0.000000 p507= 0.000000 p508=
0.000000 p509= 0.000000 p510= 0.000000 p511= 0.000000 p512= 0.000000 p513= 0.000000 p514=
0.000000 p515= 0.000000 p516= 0.000000 p517= 0.000000 p518= 0.000000 p519= 0.000000 p520=
0.000000 p521= 0.000000 p522= 0.000000 p523= 0.000000 p524= 0.000000 p525= 0.000000 p526=
0.000000 p527= 0.000000 p528= 0.000000 p529= 0.000000 p530= 0.000000 p531= 0.000000 p532=
0.000000 p533= 0.000000 p534= 0.000000 p535= 0.000000 p536= 0.000000 p537= 0.000000 p538=
0.000000 p539= 0.000000 p540= 0.000000 p541= 0.000000 p542= 0.000000 p543= 0.000000 p544=
0.000000 p545= 0.000000 p546= 0.000000 p547= 0.000000 p548= 0.000000 p549= 0.000000 p550=
0.000000 p551= 0.000000 p552= 0.000000 p553= 0.000000 p554= 0.000000 p555= 0.000000 p556=
0.000000 p557= 0.000000 p558= 0.000000 p559= 0.000000 p560= 0.000000 p561= 0.000000 p562=
0.000000 p563= 0.000000 p564= 0.000000 p565= 0.000000 p566= 0.000000 p567= 0.000000 p568=
0.000000 p569= 0.000000 p570= 0.000000 p571= 0.000000 p572= 0.000000 p573= 0.000000 p574=
0.000000 p575= 0.000000 p576= 0.000000 p577= 0.000000 p578= 0.000000 p579= 0.000000 p580=
0.000000 p581= 0.000000 p582= 0.000000 p583= 0.000000 p584= 0.000000 p585= 0.000000 p586=
0.000000 p587= 0.000000 p588= 0.000000 p589= 0.000000 p590= 0.000000 p591= 0.000000 p592=
0.000000 p593= 0.000000 p594= 0.000000 p595= 0.000000 p596= 0.000000 p597= 0.000000 p598=
0.000000 p599= 0.000000 p600= 0.000000 p601= 0.000000 p602= 0.000000 p603= 0.000000 p604=
0.000000 p605= 0.000000 p606= 0.000000 p607= 0.000000 p608= 0.000000 p609= 0.000000 p610=

0.000000 p611= 0.000000 p612= 0.000000 p613= 0.000000 p614= 0.000000 p615= 0.000000 p616= 0.000000 p617= 0.000000 p618= 0.000000 p619= 0.000000 p620= 0.000000 p621= 0.000000 p622= 0.000000 p623= 0.000000 p624= 0.000000 p625= 0.000000 p626= 0.000000 p627= 0.000000 p628= 0.000000 p629= 0.000000 p630= 0.000000 p631= 0.000000 p632= 0.000000 p633= 0.000000 p634= 0.000000 p635= 0.000000 p636= 0.000000 p637= 0.000000 p638= 0.000000 p639= 0.000000 p640= 0.000000 p641= 0.000000 p642= 0.000000 p643= 0.000000 p644= 0.000000 p645= 0.000000 p646= 0.000000 p647= 0.000000 p648= 0.000000 p649= 0.000000 p650= 0.000000 p651= 0.000000 p652= 0.000000 p653= 0.000000 p654= 0.000000 p655= 0.000000 p656= 0.000000 p657= 0.000000 p658= 0.000000 p659= 0.000000 p660= 0.000000 p661= 0.000000 p662= 0.000000 p663= 0.000000 p664= 0.000000 p665= 0.000000 p666= 0.000000 p667= 0.000000 p668= 0.000000 p669= 0.000000 p670= 0.000000 p671= 0.000000 p672= 0.000000 p673= 0.000000 p674= 0.000000 p675= 0.000000 p676= 0.000000 p677= 0.000000 p678= 0.000000 p679= 0.000000 p680= 0.000000 p681= 0.000000 p682= 0.000000 p683= 0.000000 p684= 0.000000 p685= 0.000000 p686= 0.000000 p687= 0.000000 p688= 0.000000 p689= 0.000000 p690= 0.000000 p691= 0.000000 p692= 0.000000 p693= 0.000000 p694= 0.000000 p695= 0.000000 p696= 0.000000 p697= 0.000000 p698= 0.000000 p699= 0.000000 p700= 0.000000 p701= 0.000000 p702= 0.000000 p703= 0.000000 p704= 0.000000 p705= 0.000000 p706= 0.000000 p707= 0.000000 p708= 0.000000 p709= 0.000000 p710= 0.000000 p711= 0.000000 p712= 0.000000 p713= 0.000000 p714= 0.000000 p715= 0.000000 p716= 0.000000 p717= 0.000000 p718= 0.000000 p719= 0.000000 p720= 0.000000 p721= 0.000000 p722= 0.000000 p723= 0.000000 p724= 0.000000 p725= 0.000000 p726= 0.000000 p727= 0.000000 p728= 0.000000 p729= 0.000000 p730= 0.000000 p731= 0.000000 p732= 0.000000 p733= 0.000000 p734= 0.000000 p735= 0.000000 p736= 0.000000 p737= 0.000000 p738= 0.000000 p739= 0.000000 p740= 0.000000 p741= 0.000000 p742= 0.000000 p743= 0.000000 p744= 0.000000 p745= 0.000000 p746= 0.000000 p747= 0.000000 p748= 0.000000 p749= 0.000000 p750= 0.000000 p751= 0.000000 p752= 0.000000 p753= 0.000000 p754= 0.000000 p755= 0.000000 p756= 0.000000 p757= 0.000000 p758= 0.000000 p759= 0.000000 p760= 0.000000 p761= 0.000000 p762= 0.000000 p763= 0.000000 p764= 0.000000 p765= 0.000000 p766= 0.000000 p767= 0.000000 p768= 0.000000 p769= 0.000000 p770= 0.000000 p771= 0.000000 p772= 0.000000 p773= 0.000000 p774= 0.000000 p775= 0.000000 p776= 0.000000 p777= 0.000000 p778= 0.000000 p779= 0.000000 p780= 0.000000 p781= 0.000000 p782= 0.000000 p783= 0.000000 p784= 0.000000 p785= 0.000000 p786= 0.000000 p787= 0.000000 p788= 0.000000 p789= 0.000000 p790= 0.000000 p791= 0.000000 p792= 0.000000 p793= 0.000000 p794= 0.000000 p795= 0.000000 p796= 0.000000 p797= 0.000000 p798= 0.000000 p799= 0.000000 p800= 0.000000 p801= 0.000000 p802= 0.000000 p803= 0.000000 p804= 0.000000 p805= 0.000000 p806= 0.000000 p807= 0.000000 p808= 0.000000 p809= 0.000000 p810= 0.000000 p811= 0.000000 p812= 0.000000 p813= 0.000000 p814= 0.000000 p815= 0.000000 p816= 0.000000 p817= 0.000000 p818= 0.000000 p819= 0.000000 p820= 0.000000 p821= 0.000000 p822= 0.000000 p823= 0.000000 p824= 0.000000 p825= 0.000000 p826= 0.000000 p827= 0.000000 p828= 0.000000 p829= 0.000000 p830= 0.000000 p831= 0.000000 p832= 0.000000 p833= 0.000000 p834= 0.000000 p835= 0.000000 p836= 0.000000 p837= 0.000000 p838= 0.000000 p839= 0.000000 p840= 0.000000 p841= 0.000000 p842= 0.000000 p843= 0.000000 p844= 0.000000 p845= 0.000000 p846= 0.000000 p847= 0.000000 p848= 0.000000 p849= 0.000000 p850= 0.000000 p851= 0.000000 p852= 0.000000 p853= 0.000000 p854= 0.000000 p855= 0.000000 p856= 0.000000 p857= 0.000000 p858= 0.000000 p859= 0.000000 p860= 0.000000 p861= 0.000000 p862= 0.000000 p863= 0.000000 p864= 0.000000 p865= 0.000000 p866= 0.000000 p867= 0.000000 p868= 0.000000 p869= 0.000000 p870= 0.000000 p871= 0.000000 p872= 0.000000 p873= 0.000000 p874= 0.000000 p875= 0.000000 p876= 0.000000 p877= 0.000000 p878= 0.000000 p879= 0.000000 p880= 0.000000 p881= 0.000000 p882= 0.000000 p883= 0.000000 p884= 0.000000 p885= 0.000000 p886= 0.000000 p887= 0.000000 p888= 0.000000 p889= 0.000000 p890= 0.000000 p891= 0.000000 p892= 0.000000 p893= 0.000000 p894= 0.000000 p895= 0.000000 p896= 0.000000 p897= 0.000000 p898= 0.000000 p899= 0.000000 p900= 0.000000 p901= 0.000000 p902= 0.000000 p903= 0.000000 p904= 0.000000 p905= 0.000000 p906= 0.000000 p907= 0.000000 p908= 0.000000 p909= 0.000000 p910= 0.000000 p911= 0.000000 p912= 0.000000 p913= 0.000000 p914= 0.000000 p915= 0.000000 p916= 0.000000 p917= 0.000000 p918= 0.000000 p919= 0.000000 p920= 0.000000 p921= 0.000000 p922= 0.000000 p923= 0.000000 p924= 0.000000 p925= 0.000000 p926= 0.000000 p927= 0.000000 p928= 0.000000 p929= 0.000000 p930= 0.000000 p931= 0.000000 p932= 0.000000 p933= 0.000000 p934=

0.000000 p935= 0.000000 p936= 0.000000 p937= 0.000000 p938= 0.000000 p939= 0.000000 p940= 0.000000 p941= 0.000000 p942= 0.000000 p943= 0.000000 p944= 0.000000 p945= 0.000000 p946= 0.000000 p947= 0.000000 p948= 0.000000 p949= 0.000000 p950= 0.000000 p951= 0.000000 p952= 0.000000 p953= 0.000000 p954= 0.000000 p955= 0.000000 p956= 0.000000 p957= 0.000000 p958= 0.000000 p959= 0.000000 p960= 0.000000 p961= 0.000000 p962= 0.000000 p963= 0.000000 p964= 0.000000 p965= 0.000000 p966= 0.000000 p967= 0.000000 p968= 0.000000 p969= 0.000000 p970= 0.000000 p971= 0.000000 p972= 0.000000 p973= 0.000000 p974= 0.000000 p975= 0.000000 p976= 0.000000 p977= 0.000000 p978= 0.000000 p979= 0.000000 p980= 0.000000 p981= 0.000000 p982= 0.000000 p983= 0.000000 p984= 0.000000 p985= 0.000000 p986= 0.000000 p987= 0.000000 p988= 0.000000 p989= 0.000000 p990= 0.000000 p991= 0.000000 p992= 0.000000 p993= 0.000000 p994= 0.000000 p995= 0.000000 p996= 0.000000 p997= 0.000000 p998= 0.000000 p999= 0.000000

References:

`http://www.codewithc.com/gauss-elimination-method-algorithm-flowchart/`