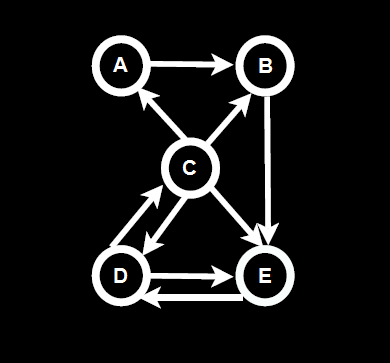
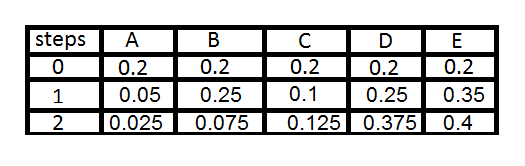
When you go and type some keywords in Google Search Engine a list of Web Pages will be displayed ,but how does the search engine know which page to be shown first to the user ? To solve this problem a  algorithm called PageRank was developed at Stanford university by Larry Page and Sergey Brin in 1996.The PageRank Algorithm uses probabilistic distribution to calculate rank of a Web page and using this rank display the search results to the user. The Pagerank is recalculated every time  the search engine crawls the web.

The original Page Rank algorithm which was described by Larry Page and Sergey Brin is :

Larry page is founder of google  
  
**PR(A) = (1-d) + d (PR(W1)/C(W1) + ... + PR(Wn)/C(Wn))**  
  
Where :  
PR(A) – Page Rank of page A  
PR(Wi) – Page Rank of pages Wi which link to page A  
C(Wi) - number of outbound links on page Wi  
d - damping factor which can be set between 0 and 1  
  
To calculate PageRank for the n Webpages ,First we initialise all Webpages with equal page rank of 1/n each.Then Step by Step we calculate Page Rank for each Webpage one after the other.  
  
  
Let us take one example :

[](https://2.bp.blogspot.com/-ax1tVrOD3GA/WEmkCUFTnXI/AAAAAAAAAYI/0l4-yoyzwMMV838bdRIqTHmJ5EEZhiP_wCLcB/s1600/Java%2BProgram%2Bto%2BImplement%2BSimple%2BPageRank%2BAlgorithm.png)

There are 5 Web pages represented by Nodes A, B, C , D, E .The hyperlink from each webpage to the other is represented by the arrow head.

[](https://3.bp.blogspot.com/--xQU4Qbl6BA/WEmgSLbO9GI/AAAAAAAAAX8/w9jfLQbHXpwPP_7HTrVITDrONvweAZp7gCLcB/s1600/PageRank%2BValues.png)

At 0th Step we have all Webpages PageRank values 0.2 that is 1/5 (1/n) . To get PageRank of Webpage A ,consider all the incoming links to A .So we have 1/4th the Page Rank of C is pointed to A. So it will be (1/5)\*(1/4) which is (1/20) or 0.05 the Page Rank of A.   
  
Similarly the Page Rank of B will be  (1/5)\*(1/4)+(1/5)\*(1/1) which is (5/20) or 0.25 because A's PageRank value is 1/5 or 0.2 from Step 0 . Even though we got 0.05 of A's PageRank in Step 1 we are considering 0.05 when we are Calculating Page Rank of B in Step 2.  
  
The general rule is --> we consider (N-1)th step values when we are calculating the Page Rank values for Nth Step . Not Clear ? Please Comment it below .  
  
In Similar way we calculate all the Page Rank Values and Sort them to Get the Most important Webpage to be displayed in the Search Results .

|  |
| --- |
| [Java Program to Implement Simple PageRank Algorithm](https://1.bp.blogspot.com/-mG1vM1WSvGM/WEmfcOVzNrI/AAAAAAAAAX0/PQ4fgS7qkvMrNBZ_FDEhm6ATOHQQOTUvwCPcB/s1600/PageRankTable.png) |
| Edith Law - lecture12 |

Java Code for Page Rank Algorithm :

import java.util.\*;

import java.io.\*;

public class PageRank {

public int path[][] = new int[10][10];

public double pagerank[] = new double[10];

public void calc(double totalNodes){

double InitialPageRank;

double OutgoingLinks=0;

double DampingFactor = 0.85;

double TempPageRank[] = new double[10];

int ExternalNodeNumber;

int InternalNodeNumber;

int k=1; // For Traversing

int ITERATION\_STEP=1;

InitialPageRank = 1/totalNodes;

System.out.printf(" Total Number of Nodes :"+totalNodes+"\t Initial PageRank of All Nodes :"+InitialPageRank+"\n");

// 0th ITERATION \_ OR \_ INITIALIZATION PHASE

for(k=1;k<=totalNodes;k++)

{

this.pagerank[k]=InitialPageRank;

}

System.out.printf("\n Initial PageRank Values , 0th Step \n");

for(k=1;k<=totalNodes;k++)

{

System.out.printf(" Page Rank of "+k+" is :\t"+this.pagerank[k]+"\n");

}

while(ITERATION\_STEP<=2) // Iterations

{

// Store the PageRank for All Nodes in Temporary Array

for(k=1;k<=totalNodes;k++)

{

TempPageRank[k]=this.pagerank[k];

this.pagerank[k]=0;

}

for(InternalNodeNumber=1;InternalNodeNumber<=totalNodes;InternalNodeNumber++)

{

for(ExternalNodeNumber=1;ExternalNodeNumber<=totalNodes;ExternalNodeNumber++)

{

if(this.path[ExternalNodeNumber][InternalNodeNumber] == 1)

{

k=1;

OutgoingLinks=0; // Count the Number of Outgoing Links for each ExternalNodeNumber

while(k<=totalNodes)

{

if(this.path[ExternalNodeNumber][k] == 1 )

{

OutgoingLinks=OutgoingLinks+1; // Counter for Outgoing Links

}

k=k+1;

}

// Calculate PageRank

this.pagerank[InternalNodeNumber]+=TempPageRank[ExternalNodeNumber]\*(1/OutgoingLinks);

}

}

}

System.out.printf("\n After "+ITERATION\_STEP+"th Step \n");

for(k=1;k<=totalNodes;k++)

System.out.printf(" Page Rank of "+k+" is :\t"+this.pagerank[k]+"\n");

ITERATION\_STEP = ITERATION\_STEP+1;

}

// Add the Damping Factor to PageRank

for(k=1;k<=totalNodes;k++)

{

this.pagerank[k]=(1-DampingFactor)+ DampingFactor\*this.pagerank[k];

}

// Display PageRank

System.out.printf("\n Final Page Rank : \n");

for(k=1;k<=totalNodes;k++)

{

System.out.printf(" Page Rank of "+k+" is :\t"+this.pagerank[k]+"\n");

}

}

public static void main(String args[])

{

int nodes,i,j,cost;

Scanner in = new Scanner(System.in);

System.out.println("Enter the Number of WebPages \n");

nodes = in.nextInt();

PageRank p = new PageRank();

System.out.println("Enter the Adjacency Matrix with 1->PATH & 0->NO PATH Between two WebPages: \n");

for(i=1;i<=nodes;i++)

for(j=1;j<=nodes;j++)

{

p.path[i][j]=in.nextInt();

if(j==i)

p.path[i][j]=0;

}

p.calc(nodes); } }

Run this code on Eclipse check output