**Project Report**

**on**

**Optimal Vaccination Sequencer for Corona Virus**

Submitted by:

Simran Biswas 8591

Chelsea Dabre 8597

Celine D’silva 8602

**1.Problem Statement:**

An analysis of the ongoing Corona outbreak reveals that transmission of the virus occurs in social networks. A social network is represented as an undirected graph **G = (V, E)**, where **V** denotes the set of cities and **E** denotes the set of edges connecting the cities.

At time t = 0, all the cities are vulnerable. At each instant of time **t > 0**, any vulnerable city **v** which is connected to some infected city **u** (i.e. **(u, v) ∈ E**) also gets infected, unless it was vaccinated at some time **t' <= t**.

The aim of the project is to vaccinate cities optimally such that the **vaccination rate is maximum.**

**2.Functionalities/Features supported by the project:**

The project uses the technique of **recursive backtracking**. There are total **K** units of supply of vaccines. The target is to vaccinate certain number of nodes using these **K** units of vaccines. You will start vaccinating the nodes from time **t = 1**, up to time **t = K** . Note that you can not vaccinate more than one node a particular time.

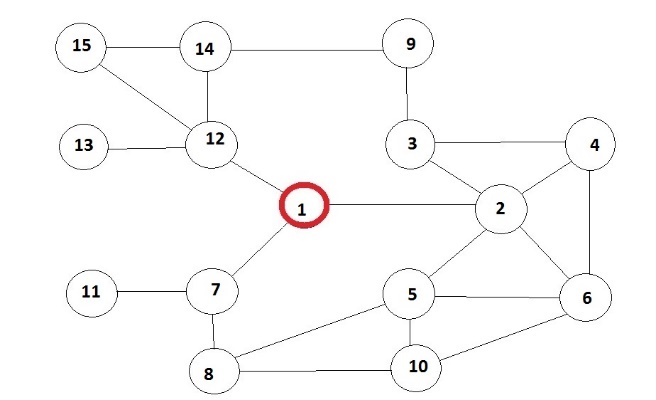
To form a graphical structure set of data to be entered by the officals is:

* Names of the cities under the province
* Population in lakhs
* Number of vaccines available
* Routes connecting the cities

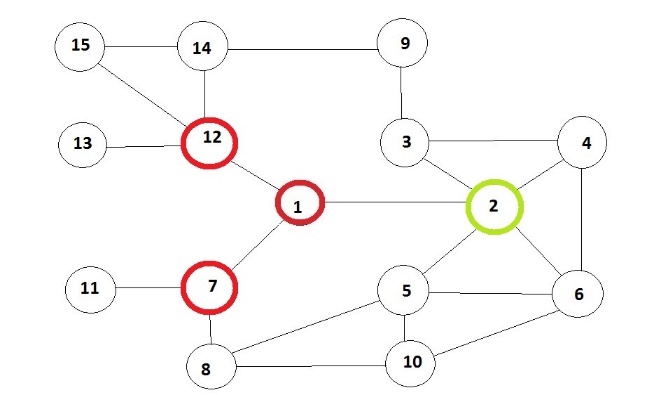
Thus by recursive backtracking we start vaccinating a city which is near to affected city at particular instant of time. An optimal order of vaccinating cities is found so that we maximize the number people vaccinated thus decreasing the spread of virus.

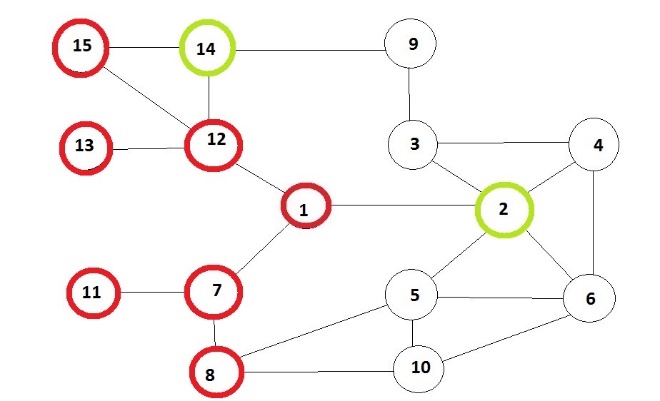
**3.Any Special Algorithm Or Method Used For Implementation:**

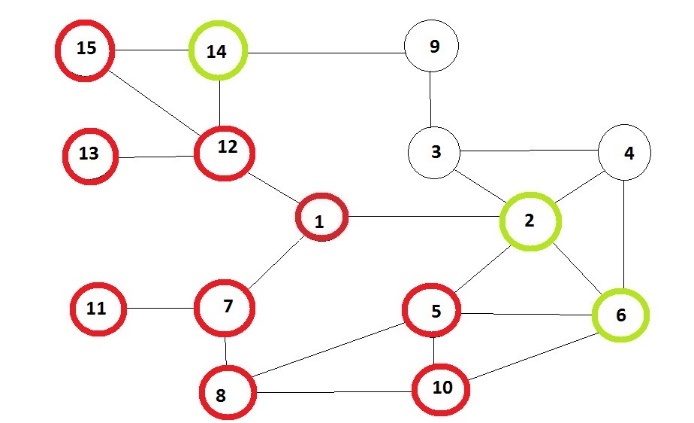
The project includes **breadth first search** algorithm to find optimal route for vaccines. A graphical example of how the process occurs at various instants of time is illustrated below:

**t =0 : Initial stage when one city is already affected and three vaccines are available  
**

**t = 1 : The neighbouring cities are affected so node 2 which is also a neighbour is immediately vaccinated to stop the further spread**



**t = 2 : The process continues further using BFS algorithm**   


**t = 3 : Depending on the number of vaccines available an optimal set is selected i.e. nodes 2,4,14 are vaccinated**  


**5.Code and output of the program:**

/\*

AOA Project:- Optimal Vaccination Sequencer

SE Computers, Batch A

8591

8597

8602

\*/

#include<bits/stdc++.h>

#define lim 10005

using namespace std;

//struct node indicates cities

struct node

{

int val;

int id;

int tim;

};

vector<struct node> ma;

bool vis[lim];

vector<int>adj[lim];

vector<int>v[lim];

int c[lim];

int dist[lim];

list<int>q;

char city[lim][lim];

int ind[lim];

//Function for calculating node at time = 1

int cmp(const void \*a,const void \*b)

{

struct node x = \*(struct node \*)a;

struct node y = \*(struct node \*)b;

return x.val - y.val;

}

//Function for calculating node at time = 2

int cmp3(const void \*a,const void \*b)

{

return c[\*(int \*)b] - c[\*(int \*)a];

}

//Function for calculating node at time = 3

int cmp2(const void \*a,const void\*b)

{

struct node x = \*(struct node \*)a;

struct node y = \*(struct node \*)b;

return x.tim-y.tim;

}

//BFS Function to find route for vaccines

void bfs(int s)

{

vector<int> :: iterator i;

int v;

vis[s] = true;

dist[s] = 0;

q.push\_back(s);

while(!q.empty()){

v = q.front();

q.pop\_front();

for(i = adj[v].begin();i != adj[v].end();i++){

if(!vis[\*i]){

q.push\_back(\*i);

dist[\*i] = dist[v] + 1;

vis[\*i] = true;

}

}

}

}

//MAIN function

int main()

{

int n,m,k,s,i,dis,tmp,max,ans,id,u,ve,j,cc=0,r[lim];

printf("\n\t\t\t~~~~~~~~~~~~~~~~~~ OPTIMAL VACCINATION SEQUENCER ~~~~~~~~~~~~~~~~~~\n\n");

printf("\n\t\t\t\t\t\t~~~~~~~~~~~~~~~~~~\n\n");

printf("\t\t\tDue to the rapid spread of the deadly corona virus, Government has taken out a \n\t\t\tforum to collect data and schedule a vaccination program in order to disinfect \n\t\t\tthe cities and curb its spread.\n");

printf("\n\t\t\tYou are requested to plan a graphical structure of your province and enter\n\t\t\t the following details.\n");

printf("\n\t\t\t\t\t\t~~~~~~~~~~~~~~~~~~\n\n");

printf("\n\t\t\t\tEnter the number of cities:\t");

scanf("%d", &n);

printf("\n\t\t\tEnter the number of units of vaccines provided:\t");

scanf("%d", &k);

printf("\n\t\t\tEnter names of the cities along with serial index\n");

for(i = 0;i <n;i++)

{

printf("\n\t\t\tCity: ");

scanf("%s", city[i]);

printf("\t\t\tIndex: ");

scanf("%d", &ind[i]);

}

printf("\n\t\t\tEnter the number of total routes(edges) through all cities: ");

scanf("%d", &m);

printf("\n\t\t\tCity\tIndex\n");

for(i=0;i<n;i++)

{

printf("\t\t\t%s : %d\n", city[i],ind[i]);

}

printf("\n\t\t\tFrom above list, enter the index of the first city affected:\t");

scanf("%d", &i);

s = i-1;

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

{

printf("\n\t\t\tEnter population in lakhs in city of index %d: ", i);

scanf("%d",&c[i]);

}

printf("\n\t\t\tEnter all routes :(For example: if route exists from city 1 and city 3, enter 1 and 3)\n\t\t\t\*\*\*DO NOT REPEAT ANY ROUTE\*\*\*\n");

for(i = 0;i < m;i++)

{

printf("\n\t\t\tEnter source index:");

scanf("%d",&u);

printf("\t\t\tEnter destination index:");

scanf("%d",&ve);

adj[u].push\_back(ve);

adj[ve].push\_back(u);

}

bfs(s);

dis = 0;

for(i = 1;i <= n;i++){

v[dist[i]].push\_back(i);

if(dist[i] > dis){

dis = dist[i];

}

}

int flag = 0;

if(k < dis){

dis = k;

flag = 1;

}

if(flag == 0){

for(i = 1;i <= dis;i++){

qsort(&v[i][0],v[i].size(),sizeof(int),cmp3);

}

struct node tmp;

for(i = 1;i <= dis;i++){

int siz = v[i].size();

for(j = 0;j < min(i,siz);j++){

tmp.val = c[v[i][j]];

tmp.id = v[i][j];

tmp.tim = i;

ma.push\_back(tmp);

}

}

qsort(&ma[0],ma.size(),sizeof(struct node),cmp);

qsort(&ma[0],dis,sizeof(struct node),cmp2);

for(i = 0;i < dis;i++)

{

printf(" %d ",ma[i].id);

r[cc++] = ma[i].id;

}

for(;i <= k;i++){

ans = rand()%n;

ans++;

printf(" %d ",ans);

r[cc++] = ma[i].id;

}

printf("\n");

}

else{

for(i = 1;i <= dis;i++){

qsort(&v[i][0],v[i].size(),sizeof(int),cmp3);

}

struct node tmp;

for(i = 1;i <= dis;i++){

int siz = v[i].size();

for(j = 0;j < min(i,siz);j++){

tmp.val = c[v[i][j]];

tmp.id = v[i][j];

tmp.tim = i;

ma.push\_back(tmp);

}

}

qsort(&ma[0],ma.size(),sizeof(struct node),cmp);

qsort(&ma[0],k,sizeof(struct node),cmp2);

for(i = 0;i < k;i++){

printf(" %d ",ma[i].id);

r[cc++] = ma[i].id;

}

printf("\n");

}

printf("\n\t\t\tOrder in which cities need to be vaccinated in order to reduce the spread of the virus are:\n");

for(i = 0;i<cc;i++)

{

printf("\t\t\t%s", city[r[i]-1]);

}

}

/\***OUTPUT**

C:\Users\Simran\Desktop\Practicals>a

~~~~~~~~~~~~~~~~~~ OPTIMAL VACCINATION SEQUENCER ~~~~~~~~~~~~~~~~~~

~~~~~~~~~~~~~~~~~~

Due to the rapid spread of the deadly corona virus, Government has taken out a forum to collect data and schedule a vaccination program in order to disinfect the cities and curb its spread.

You are requested to plan a graphical structure of your province and enter the following details.

~~~~~~~~~~~~~~~~~~

Enter the number of cities: 15

Enter the number of units of vaccines provided:3

Enter names of the cities along with serial index

City: Thane

Index: 1

City: Navi-Mumbai

Index: 2

City: Solapur

Index: 3

City: Latur

Index: 4

City: Pune

Index: 5

City: Mira-Bhayandar

Index: 6

City: Mumbai

Index: 7

City: Vasai-Virar

Index: 8

City: Nagpur

Index: 9

City: Kalyan

Index: 10

City: Pimpri

Index: 11

City: Amravati

Index: 12

City: Nanded

Index: 13

City: Bhiwandi

Index: 14

City: Nashik

Index: 15

Enter the number of total routes(edges) through all cities: 22

City Index

Thane : 1

Navi-Mumbai : 2

Solapur : 3

Latur : 4

Pune : 5

Mira-Bhayandar : 6

Mumbai : 7

Vasai-Virar : 8

Nagpur : 9

Kalyan : 10

Pimpri : 11

Amravati : 12

Nanded : 13

Bhiwandi : 14

Nashik : 15

From above list, enter the index of the first city affected: 2

Enter population in lakhs in city of index 1: 2

Enter population in lakhs in city of index 2: 3

Enter population in lakhs in city of index 3: 1

Enter population in lakhs in city of index 4: 4

Enter population in lakhs in city of index 5: 5

Enter population in lakhs in city of index 6: 3

Enter population in lakhs in city of index 7: 6

Enter population in lakhs in city of index 8: 8

Enter population in lakhs in city of index 9: 2

Enter population in lakhs in city of index 10: 4

Enter population in lakhs in city of index 11: 8

Enter population in lakhs in city of index 12: 1

Enter population in lakhs in city of index 13: 4

Enter population in lakhs in city of index 14: 8

Enter population in lakhs in city of index 15: 9

Enter all routes :(For example: if route exists from city 1 and city 3, enter 1 and 3)

\*\*\*DO NOT REPEAT ANY ROUTE\*\*\*

Enter source index:1

Enter destination index:2

Enter source index:1

Enter destination index:7

Enter source index:1

Enter destination index:12

Enter source index:2

Enter destination index:3

Enter source index:2

Enter destination index:4

Enter source index:2

Enter destination index:5

Enter source index:2

Enter destination index:6

Enter source index:3

Enter destination index:4

Enter source index:3

Enter destination index:9

Enter source index:4

Enter destination index:6

Enter source index:5

Enter destination index:6

Enter source index:5

Enter destination index:8

Enter source index:5

Enter destination index:10

Enter source index:6

Enter destination index:10

Enter source index:7

Enter destination index:8

Enter source index:7

Enter destination index:11

Enter source index:8

Enter destination index:10

Enter source index:9

Enter destination index:14

Enter source index:12

Enter destination index:13

Enter source index:12

Enter destination index:14

Enter source index:12

Enter destination index:15

Enter source index:14

Enter destination index:15

7 10 9

Order in which cities need to be vaccinated in order to reduce the spread of the virus are:

Mumbai Kalyan Nagpur

\*/