Logo, company name

Description automatically generated

**COMSATS University Islamabad (CUI)**

**Project Report**

**for**

**Crisis Responder System**

Phase 1

***By***

**Student Name 1 : Muhammad Fakhar**

**CIIT/SP21-BCS-061/ISB**

***Supervisor* SIR MUHAMMAD MUSTAFA KHATTAK**

Table of Contents

Abstract 1

1. Introduction 2

2. Problem Statement 2

3. Problem Solution/Objectives of the Proposed System 2

3.1 Objectives 2

4. Related System Analysis 3

5. Vision Statement 3

6. Scope 4

7. System Design/Modules 4

7.1 Module 1: Module Name 4

Module 1: Order Meal 4

Module 2: Menu Operations 4

8. System Limitations/Constraints 4

9. System Code

Project Category: (Select all the major domains of proposed project)

 **A-**Desktop Application/Information System  **B-**Web Application/Web Application based Information System  **C-**Problem Solving and Artificial Intelligence  **D-**Simulation and Modeling  **E-**Smartphone Application

 **F-**Smartphone Game  **G-**Networks  **H-**Image Processing Other (specify category) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Abstract

Write a one paragraph abstract keeping in view the following guideline:

I am creating this project to tackle the crisis response problems. To automate the ambulance, police, fire brigade and terrorism problems, this project is made.

In today's fast-paced world, timely response to accidents is crucial for ensuring the well-being and safety of individuals involved. Traditional emergency response systems often face challenges in identifying the nearest available resources and coordinating their dispatch.

There can be problems in controlling these crisis if the system is not automated properly.

My project will, on the condition of crisis, will respond to the caller. If someone had accident, on his call, ambulance will be sent from the nearest hospital. If someone had report of fire, fire brigade from the nearest station will be sent. Same goes for the robbery and terrorism problems.

# Introduction

The Crisis Responder System is a project designed to efficiently handle emergency calls related to accidents, fires, robbery and dispatch the nearest emergency response units such as ambulances and police vehicles. This system aims to minimize response times and ensure effective coordination between emergency services, with the ultimate goal of saving lives and providing prompt assistance during critical situations.

In today's fast-paced world, timely response to accidents is crucial for ensuring the well-being and safety of individuals involved. Traditional emergency response systems often face challenges in identifying the nearest available resources and coordinating their dispatch. Crisis Responder System addresses these challenges by leveraging Dijkstra's algorithm, a well-known graph traversal algorithm, to determine the optimal routes for emergency vehicles based on their proximity to the incident location and the availability of emergency resources.

The system handles incoming emergency calls, which report accidents or related incidents, and automatically identifies the nearest hospital or police station based on the geographic location provided. By employing Dijkstra's algorithm, the system calculates the shortest path from the incident location to the nearest emergency response facility, minimizing travel time and ensuring a swift response.

# Problem Statement

The Crisis Responder System is being developed to address the pressing issue of inefficient and delayed emergency response in the event of accidents or related incidents. The current emergency response systems often struggle with identifying the nearest available emergency resources, resulting in delayed assistance and potentially compromising the well-being of individuals involved in accidents. This software system aims to solve this problem by leveraging Dijkstra's algorithm to efficiently calculate the shortest path from the incident location to the nearest hospital or police station. By automating the dispatch process and ensuring timely coordination, the system aims to minimize response times, optimize resource allocation, and ultimately save lives by providing prompt and effective assistance during critical situations.

# Problem Solution/Objectives of the Proposed System

The Crisis Responder System is an application developed to address the problem of inefficient and delayed emergency response in the event of accidents or related incidents. By integrating geographic information and leveraging Dijkstra's algorithm, the system aims to accurately determine the incident location and identify the nearest hospital or police station. The primary objectives and goals of the software system include efficient emergency call handling, optimal resource allocation through the shortest path computation, real-time tracking and monitoring of dispatched emergency vehicles, and enhanced coordination and communication among emergency response units. By achieving these objectives, the application strives to minimize response times, optimize resource allocation, and ultimately contribute to saving lives and ensuring the well-being of individuals involved in accidents.

## Objectives

BO-1: Efficient emergency call handling.

BO-2: Accurate determination of incident location and identification of the nearest hospital or police station.

BO-3: Optimal resource allocation through the use of Dijkstra's algorithm for computing the shortest path.

BO-4: Real-time tracking and monitoring of dispatched emergency vehicles.

BO-5: Enhanced coordination and communication among emergency response units, hospitals, and police stations.

# Related System Analysis

# Vision Statement

Our vision is to revolutionize emergency response systems and establish the Crisis Responder System as the leading solution for efficient and timely assistance during accidents. By leveraging advanced technologies and intelligent algorithms, we aim to minimize response times, optimize resource allocation, and ultimately save lives. Through seamless coordination and effective communication, our system will ensure that individuals involved in accidents receive prompt and appropriate emergency services, thus enhancing their safety and well-being. We envision a future where our software plays a vital role in mitigating the impact of accidents, contributing to a safer and more secure society.

# Scope

The scope of the Crisis Responder System encompasses the development of a comprehensive software solution that handles emergency calls related to accidents and efficiently dispatches the nearest emergency response units. The project focuses on the following main functionalities:

1. Emergency Call Handling: The system will provide a user-friendly interface to receive and manage emergency calls. It will capture essential details such as incident location, nature of the accident, and contact information.
2. Geographic Information Integration: The application will incorporate geographic information, including maps and location data, to accurately determine the incident location and identify the nearest hospital or police station.
3. Dijkstra's Algorithm Implementation: Utilizing Dijkstra's algorithm, the system will calculate the shortest path from the incident location to the nearest emergency response facility, optimizing response times and resource allocation.
4. Resource Dispatch and Coordination: Once the nearest emergency response facility is identified, the system will automatically dispatch the appropriate resources, such as ambulances or police vehicles, ensuring a prompt and coordinated response.
5. Real-time Tracking and Monitoring: The system will provide real-time tracking and monitoring of dispatched emergency vehicles, allowing stakeholders to monitor their progress and make informed decisions.

The project will focus on the software development aspect, including designing an intuitive user interface, implementing efficient algorithms for incident location identification and route calculation, and integrating real-time tracking functionalities. The scope will not cover hardware implementation or infrastructure setup for emergency response services. The system will primarily serve as a tool for emergency call handling, resource allocation, and coordination, facilitating effective and efficient emergency response in the event of accidents.

# Modules

## **Module 1: Police Station:**

There are 2 police stations located in city. A police has a number of (12).

Depending upon the seriousness of crime if there is local crime (i.e. mobile

snatching) then bike squad is send on the specific place by using the shortest

path with the help of dijkastra and is those are decremented from the total bike

squad present.

If there is big crime (i.e. bank robbery, Terrorism) then auto mobile of police is send.

Anti-Terrorist squad is also an option after that.

**7.2 Module 2: Ambulance:**

There are 4 hospitals located in city:

A call number for the Ambulance is (1122) .

A user calls and if there is normal accident then the help

is provided to them with the help of bike squad.

On contrary if patient condition is serious and needed to shift in hospital the

ambulance service is provided

With the help of dijkastra by finding the smallest path

**7.3 Module 3: Anti-Terrorist squad(SSG):**

There is one zarrar unit in city. These are specialized commandos act as a last resort.

If there is terrorist activity which police can’t handle,

then these squad is called . They are 20 in total.

**7.4 Module 4: Fire Brigade:**

Similarly, if there is fire blown in some area

then Help is provided to particular area using the smallest

distance.

# System Limitations/Constraints

Write down the limitations and constraints of the proposed project.

Example:

LI-1: Some food items that are available from the cafeteria will not be suitable for delivery, so the delivery menus available to patrons of the COS must be a subset of the full cafeteria menus.

LI-2: The COS shall be used only for the cafeteria at the Process Impact campus in Clackamas, Oregon.

# SYSTEM CODE

//Main class

#include <iostream>

#include <time.h>

#include "Path.h"

#include "hospital.h"

#include "que.h"

#include "Firebrigade.h"

#include "policestation.h"

#include "tree.h"

#include "SSg.h"

using namespace std;

void handlecall();

void adminmenu();

void Call();

void Firebrig(string Sloc);

void policestat(string Sloc);

void SSgcalling(string Sloc);

void response();

void available();

hospital shifa;hospital medcentre;hospital rawal;hospital quaid;

Firebrigade fbrigade;Firebrigade pindifbrigade;

policestation islamabad;policestation sadaqabad;

que line;que type;

tree database;

SSg zarrar;

void Ambulance(string Sloc){

convert w;

cout<<"\nAmbulance needed at "<<Sloc<<endl;

cout<<"\nSuggestion for Road ways to Hospital\n";

int location =w.stringconversion(Sloc);

Path p(location);

p.dijkstra();

p.display(0);

p.display(15);

p.display(11);

p.display(20);

string hospitalname;

bool success = false;

while(!success){

cout<<"\nEnter Hospital to respond\n";

cin>>hospitalname;

int hospitalnum = w.stringconversion(hospitalname);

if(hospitalnum==0){

if(shifa.ambulance!=0){

shifa.ambulance--;

cout<<"\nAmbulance sended successfully\n";

success = true;

}else{

cout<<"\nAll Ambulances are out for service\nSelect an other hospital\n";

success = false;

}

}else if(hospitalnum==11){

if(medcentre.ambulance!=0){

medcentre.ambulance--;

cout<<"\nAmbulance sended successfully\n";

success = true;

}else{

cout<<"\nAll Ambulances are out for service\nSelect an other hospital\n";

success = false;

}

}else if(hospitalnum==15){

if(rawal.ambulance!=0){

rawal.ambulance--;

cout<<"\nAmbulance sended successfully\n";

success = true;

}else{

cout<<"\nAll Ambulances are out for service\nSelect an other hospital\n";

success = false;

}

}else if(hospitalnum==20){

if(quaid.ambulance!=0){

quaid.ambulance--;

cout<<"\nAmbulance sended successfully\n";

success = true;

}else{

cout<<"\nAll Ambulances are out for service\nSelect an other hospital\n";

success = false;

}

}

}

handlecall();

}

void Firebrig(string Sloc){

convert w;

int typenumber = type.dequeue();

if(typenumber==0){

cout<<"\nFire brigade needed at "<<Sloc<<" Fire is local\n";

}else{

cout<<"\n03 Fire brigades needed at "<<Sloc<<" Fire is high\n";

}

int location =w.stringconversion(Sloc);

Path p(location);

p.dijkstra();

p.display(9);

p.display(19);

string fstationname;

bool success = false;

while(!success){

cout<<"\nEnter Firebrigade station to respond\n";

cin>>fstationname;

int fstationnum = w.stringconversion(fstationname);

if(fstationnum==9){

if(typenumber==0){

if(fbrigade.firebrigade!=0){

fbrigade.firebrigade--;

cout<<"\n1 Firebrigade sended successfully";

success = true;

}else{

cout<<"\nAll Firebrigade are out for service\nSelect an other Station\n";

success = false;

}

}else{

if(fbrigade.firebrigade!=0){

int a=fbrigade.firebrigade;

fbrigade.firebrigade=fbrigade.firebrigade-a;

cout<<a<<" Firebrigade available are sended successfully";

success = true;

}else{

cout<<"\nAll Firebrigade are out for service\nSelect an other Station\n";

success = false;

}

}

}else if(fstationnum==19){

if(typenumber==0){

if(pindifbrigade.firebrigade!=0){

pindifbrigade.firebrigade--;

cout<<"\n1 Firebrigade sended successfully\n";

success = true;

}else{

cout<<"\nAll Firebrigade are out for service\nSelect an other Station\n";

success = false;

}

}else{

if(pindifbrigade.firebrigade!=0){

int a=pindifbrigade.firebrigade;

pindifbrigade.firebrigade=pindifbrigade.firebrigade-a;

cout<<a<<" Firebrigade available are sended successfully";

success = true;

}else{

cout<<"\nAll Firebrigade are out for service\nSelect an other Station\n";

success = false;

}

}

}

}

handlecall();

}

void policestat(string Sloc){

convert w;

int typenumber = type.dequeue();

if(typenumber==0){

cout<<"\n Police Bike squad needed at "<<Sloc<<endl;

}else{

cout<<"\n Police Mobile needed at "<<Sloc<<endl;

}

cout<<"\n Suggestion for Road ways to Police station\n";

int location =w.stringconversion(Sloc);

Path p(location);

p.dijkstra();

p.display(10);

p.display(12);

string pstationname;

bool success = false;

while(!success){

cout<<"\nEnter Police station to respond\n";

cin>>pstationname;

int pstationnum = w.stringconversion(pstationname);

if(pstationnum==10){

if(typenumber==0){

if(islamabad.bikesquad!=0){

islamabad.bikesquad--;

cout<<"\nBike squad sended successfully\n";

success = true;

}

else{

cout<<"\nBike squads are out for service\nSelect an other Station\n";

success = false;

}

}else{

if(islamabad.pmobile!=0){

islamabad.pmobile--;

cout<<"Police mobile sended successfully\n";

success = true;

int num;

cout<<"\nEnter choices::\n 1 :SSGs needed\n 2 :SSGs not needed\n";

cin>>num;

if(num==1){

SSgcalling(Sloc);

}

}else{

cout<<"Police mobile are out for service\nSelect an other Station\n";

success = false;

}

}

}

else if(pstationnum==12){

if(typenumber==0){

if(sadaqabad.bikesquad!=0){

sadaqabad.bikesquad--;

cout<<"\nBike squad sended successfully\n";

success = true;

}

else{

cout<<"\nBike squads are out for service\nSelect an other Station\n";

success = false;

}

}else{

if(sadaqabad.pmobile!=0){

sadaqabad.pmobile--;

cout<<"\nPolice mobile sended successfully\n";

success = true;

int num;

cout<<"\nEnter choices::\n 1 :ssgs needed\n 2 : ssgs not needed";

cin>>num;

if(num==1){

SSgcalling(Sloc);

}

}else{

cout<<"\nPolice mobile are out for service\nSelect an other Station\n";

success = false;

}

}

}

}

handlecall();

}

void SSgcalling(string Sloc){

convert w;

cout<<"\n Suggestion for Road ways to SSG Unit\n";

int location =w.stringconversion(Sloc);

Path p(location);

p.dijkstra();

p.display(5);

int ssgmen;

cout<<"\nHow many ssgs required :? \n";

cin>>ssgmen;

int a = zarrar.men;

if(zarrar.men!=0){

if(zarrar.men<ssgmen){

zarrar.men = zarrar.men-a;

cout<<"\n"<<a<<"Available ssg commandoes are sended succesfully: \n";

}

else {

zarrar.men=zarrar.men-ssgmen;

cout<<"\n"<< ssgmen <<"Available ssg commandoes are sended succesfully: \n";

}

}

else {

cout<<"\nNo ssgs avaliable all are out for operations\n";

}

handlecall();

}

void response(){

cout<<"\n1:Check Ambulance\n2:Check FireBrigade\n3:Check PoliceAutomobiles\n4:Check SSG commadoes\n5:Admin menu\n";

int ver;

cin>>ver;

switch(ver){

case 1:

if(shifa.ambulance!=3){

cout<<3-shifa.ambulance<<"\nAmbulances of Shifa Hospital were out for service and\n";

srand( time(NULL) );

int randNum = rand() % 2;

if(randNum==0){

cout<<"\nNot returned yet\n";

}else{

shifa.ambulance=3;

cout<<"\nReturned\n";

}

}

if(medcentre.ambulance!=3){

cout<<3-medcentre.ambulance<<"\nAmbulances of Medcentre Hospital were out for service and\n";

srand( time(NULL) );

int randNum = rand() % 2;

if(randNum==0){

cout<<"\nNot returned yet\n";

}else{

medcentre.ambulance=3;

cout<<"\nReturned\n";

}

}

if(rawal.ambulance!=3){

cout<<3-rawal.ambulance<<"\nAmbulances of Rawal Hospital were out for service and\n";

//srand( time(NULL) );

int randNum = rand() % 2;

if(randNum==0){

cout<<"\nNot returned yet\n";

}else{

rawal.ambulance=3;

cout<<"\nReturned\n";

}

}

if(quaid.ambulance!=3){

cout<<3-quaid.ambulance<<"\nAmbulances of Shifa Hospital were out for service and\n";

//srand( time(NULL) );

int randNum = rand() % 2;

if(randNum==0){

cout<<"\nNot returned yet\n";

}else{

quaid.ambulance=3;

cout<<"\nReturned\n";

}

}response();

break;

case 2:

if(fbrigade.firebrigade!=3){

cout<<3-fbrigade.firebrigade<<"\nFirebrigade of Fbrigade were out for service and\n";

srand( time(NULL) );

int randNum = rand() % 2;

if(randNum==0){

cout<<"\nNot returned yet\n";

}else{

fbrigade.firebrigade=3;

cout<<"\nReturned\n";

}

}

if(pindifbrigade.firebrigade!=3){

cout<<3-pindifbrigade.firebrigade<<"\nFirebrigade of Pindi were out for service and\n";

srand( time(NULL) );

int randNum = rand() % 2;

if(randNum==0){

cout<<"\nNot returned yet\n";

}else{

pindifbrigade.firebrigade=3;

cout<<"\nReturned\n";

}

}response();

break;

case 3:

if(islamabad.bikesquad!=3){

cout<<3-islamabad.bikesquad<<"\nBikesquad of Islamabad policestation were out for service and\n";

srand( time(NULL) );

int randNum = rand() % 2;

if(randNum==0){

cout<<"\nNot returned yet\n";

}else{

islamabad.bikesquad=3;

cout<<"\nReturned\n";

}

}

if(islamabad.pmobile!=3){

cout<<3-islamabad.bikesquad<<"\nPolice mobile of Islamabad policestation were out for service and\n";

srand( time(NULL) );

int randNum = rand() % 2;

if(randNum==0){

cout<<"\nNot returned yet\n";

}else{

islamabad.pmobile=3;

cout<<"\nReturned\n";

}

}

if(sadaqabad.bikesquad!=3){

cout<<3-sadaqabad.bikesquad<<"\nBikesquad of sadaqabad policestation were out for service and\n";

srand( time(NULL) );

int randNum = rand() % 2;

if(randNum==0){

cout<<"\nNot returned yet\n";

}else{

sadaqabad.bikesquad=3;

cout<<"\nReturned\n";

}

}

if(sadaqabad.pmobile!=3){

cout<<3-sadaqabad.bikesquad<<"\nPolice mobile of sadaqabad policestation were out for service and\n";

srand( time(NULL) );

int randNum = rand() % 2;

if(randNum==0){

cout<<"\nNot returned yet\n";

}else{

sadaqabad.pmobile=3;

cout<<"\nReturned\n";

}

}response();

break;

case 4:

if(zarrar.men!=20){

cout<<3-sadaqabad.bikesquad<<"\nZarrar Commandoes were out for service and\n";

srand( time(NULL) );

int randNum = rand() % 2;

if(randNum==0){

cout<<"\nNot returned yet\n";

}else{

zarrar.men=20;

cout<<"\nReturned\n";

}

}response();

break;

case 5:

adminmenu();

break;

}

}

void handlecall(){

line.display();

while(line.is\_empty()!=true){

int value = line.dequeue();

if(value==1122){

Ambulance(line.sdequeue());

}else if(value==101){

Firebrig(line.sdequeue());

}else if(value==15){

policestat(line.sdequeue());

}

}

cout<<" \nAll Peace! No call!\n";

adminmenu();

}

void available(){

cout<<"Ambulances at Shifa hospital "<<shifa.ambulance<<endl;

cout<<"Ambulances at Medical centre hospital "<<medcentre.ambulance<<endl;

cout<<"Ambulances at Rawal hospital "<<rawal.ambulance<<endl;

cout<<"Ambulances at Quaid e Azam hospital "<<quaid.ambulance<<endl;

cout<<endl;

cout<<"Firebrigade at Fbrigade "<<fbrigade.firebrigade<<endl;

cout<<"Firebrigade at Pindi firebrigade "<<pindifbrigade.firebrigade<<endl;

cout<<endl;

cout<<"Police bikeSqaud at Islamabad PoliceStation "<<islamabad.bikesquad<<endl;

cout<<"Police Mobile at Islamabad PoliceStation "<<islamabad.pmobile<<endl;

cout<<"Police bikeSqaud at Sadaqabad PoliceStation "<<sadaqabad.bikesquad<<endl;

cout<<"Police Mobile at Sadaqabad PoliceStation "<<sadaqabad.pmobile<<endl;

cout<<endl;

cout<<"SSGs commandos at station are "<<zarrar.men<<endl;

adminmenu();

}

void adminmenu(){

int num;

cout<<"\n1:Handle calls\n2:See Operation response\n3:See all Calling record\n4:Check Available services\n5:Switch to User to side\n";

cin>>num;

switch(num){

case 1:

handlecall();

break;

case 2:

response();

break;

case 3:

database.inorder(database.root);

adminmenu();

break;

case 4:

available();

break;

case 5:

Call();

break;

}

}

void Call(){

int callnum;

string Sloc;

int typenum;

cout<<"\nCALL \n1122 Ambulance\n101 Firebrigade\n15 Police station\n1:Switch to Admin portal\n";

cin>>callnum;

switch(callnum){

case 1:

adminmenu();

break;

case 15:

cout<<"\nEnter accident location\n";

cin>>Sloc;

cout<<"\nChoose Seriousness\n0:City crime like Thief,Street fights\n1:High profile Crime like Terrorism\n";

cin>>typenum;

database.insert(Sloc,callnum);

type.enqueue(typenum);

cout<<"\nHelp is on the way\n";

line.enqueue(15);

line.senqueue(Sloc);

Call();

break;

case 101:

cout<<"\nEnter accident location\n";

cin>>Sloc;

cout<<"\nChoose Seriousness\n0:Home Fire,appartment fire\n1:High flames/ Buildings on fire\n";

cin>>typenum;

database.insert(Sloc,callnum);

type.enqueue(typenum);

cout<<"\nHelp is on the way\n";

line.enqueue(101);

line.senqueue(Sloc);

Call();

break;

case 1122:

cout<<"\nEnter accident location\n";

cin>>Sloc;

database.insert(Sloc,callnum);

cout<<"\nHelp is on the way\n";

line.enqueue(1122);

line.senqueue(Sloc);

Call();

break;

}

}

int main(){

Call();

}

//Path class

#include <iostream>

#include "convert.h"

using namespace std;

class Path{

public:

int v;

int cost[21][21]={{0,2,3,999,999,995,996,997,998,999,9910,9911,912,913,914,915,916,917,918,919,920},

{2,0,3,4,5,6,996,997,989,999,910,911,912,913,914,915,916,917,918,919,920},

{3,3,0,4,999,995,5,2,998,999,910,911,912,913,914,915,916,917,918,919,920},

{999,4,4,0,4,999,999,1,3,3,990,911,912,913,914,915,916,917,918,919,920},

{999,5,999,4,0,7,999,999,999,1,3,1,912,913,914,915,916,917,918,919,920},

{999,6,999,993,7,0,996,9997,989,999,999,2,912,913,914,915,916,917,918,919,920},

{999,991,5,939,994,995,0,2,998,999,991,911,8,913,914,915,916,917,918,919,920},

{999,991,2,1,994,995,2,0,3,6,919,911,4,9,914,915,916,917,918,919,920},

{999,991,992,3,994,995,996,3,0,9,910,911,4,5,1,915,916,917,918,919,920},

{999,991,992,3,1,995,996,6,9,0,4,911,912,913,1,4,916,917,918,919,920},

{999,991,992,993,3,995,996,997,998,4,0,3,912,913,914,9,9,917,918,7,920},

{999,991,992,939,1,2,996,979,998,999,3,0,912,913,914,915,1,917,918,919,920},

{990,9991,992,993,994,959,8,4,4,999,910,911,0,9,914,915,916,1,918,919,920},

{999,919,992,993,994,995,996,9,5,999,910,911,9,0,2,915,916,3,3,919,920},

{999,919,992,993,994,995,996,997,1,1,910,911,912,2,0,3,916,917,4,919,2},

{999,991,992,993,994,995,996,997,998,4,9,911,912,913,3,0,3,999,918,1,9},

{999,991,992,993,994,995,996,997,998,999,9,1,999,913,914,3,0,917,918,5,999},

{999,991,992,993,994,995,996,997,998,999,910,911,1,3,914,915,916,0,2,919,920},

{990,991,992,993,994,995,996,997,998,999,991,911,912,3,4,915,916,2,0,919,1},

{999,991,992,993,994,995,996,997,998,999,7,911,912,913,914,1,5,917,918,0,4},

{999,991,992,993,994,959,999,999,998,999,991,911,912,913,2,9,916,917,1,4,0}

};

int src;

int dist[21];

int parent[21];

bool visited[21]={0};

Path(int src){

v=21;

src = src;

for(int i=0;i<v;i++){

parent[i]=i;

dist[i] = 999;

}

dist[src]=0;

parent[src]=-1;

}

int getnearest(){

int minvalue = 999;

int minnode = 0;

for(int i=0;i<v;i++){

if(!visited[i] and dist[i]<minvalue){

minvalue=dist[i];

minnode=i;

}

}return minnode;

}

void dijkstra(){

//int end =3;

for(int i=0;i<v;i++){

int nearest=getnearest();

visited[nearest]=true;

for(int adj = 0;adj<v;adj++){

if(cost[nearest][adj]!=999 and dist[adj]>dist[nearest]+cost[nearest][adj]){

dist[adj]=dist[nearest]+cost[nearest][adj];

parent[adj]=nearest;

}

}

}

}

void prims(){

int total=0;

for(int k=0;k<v;k++){

int nearest=getnearest();

visited[nearest]=true;

for(int adj = 0;adj<v;adj++){

if(cost[nearest][adj]!=999 and dist[adj]>dist[nearest]+cost[nearest][adj]){

dist[adj]=cost[nearest][adj];

parent[adj]=nearest;

}

}

}for(int i =0 ;i < v ;i++){

// int temp = parent[i] ;

cout<<i << " -> "<< parent[i];

// while(temp!=-1)

// {

// cout<< temp << " <- " ;

// temp = parent[temp] ;

// }

cout<<endl;

cout<<i << " -> "<< parent[i];

cout<<endl;

total=total+dist[i];

}cout<<"Total "<<total;

}

void display(int end){

convert c ;

int temp = parent[end] ;

cout<<c.intconversion(end) << " -> " ;

while(temp!=-1)

{

cout<< c.intconversion(temp) << " -> " ;

temp = parent[temp] ;

}

cout<<endl;

cout<<"::::Distance = " << dist[end] <<" KM " ;

cout<<endl;

}

};

//tree class

#include <iostream>

using namespace std;

class tree{

public:

double num;

string name;

string contactnum;

string Location;

int Callnum1;

tree \*left;

tree \*right;

tree \*root;

tree(){

num=0;

name="";

contactnum="";

Location="";

Callnum1=0;

left = NULL;

right = NULL;

root = NULL;

}

void insert(string l,int c){

tree \*curr = new tree;

cout<<"Enter your name\n";

cin>>curr->name;

cout<<"Enter your CNIC\n";

cin>>curr->num;

cout<<"Enter your Contact number\n";

cin>>curr->contactnum;

curr->Location=l;

curr->Callnum1=c;

if(root==NULL){

root = curr;

}else{

tree \*p = root;

tree \*k = p;

while(p!=NULL){

k=p;

if(curr->num < p->num){

p=p->left;

}else{

p=p->right;

}

}

if(curr->num < k->num){

k->left = curr;

}else{

k->right = curr;

}

}

}//when root has one branch

void deletenode(int key){

tree \*p = root;

tree \*k = p;

while(p->num!=key && p!=NULL){

k=p;

if(key<p->num){

p=p->left;

}else{

p=p->right;

}

}

if(p->left==NULL && p->right==NULL){

if(k->left=p){

k->left=NULL;

}else{

k->right=NULL;

}

}else if(p->left==NULL or p->right==NULL){

if(k->left==p && p->right==NULL){

k->left=p->left;

}else if(k->left==p && p->left==NULL){

k->left = p->right;

}else if(k->right==p && p->left==NULL){

k->right = p->right;

}else if(k->right==p && p->right==NULL){

k->right = p->left;

}

}else{

tree \*p2 = p->right;

if(p2->left==NULL){

p->num=p2->num;

p->right=p2->right;

}else{

while(p2->left!=NULL){

k=p2;

p2=p2->left;

}

p->num=p2->num;

k->left=NULL;

}

}

}

void inorder(tree \*p){

if(p!=0){

inorder(p->left);

cout<<"Call Number "<<p->Callnum1<<endl;

cout<<"Name "<<p->name<<endl;

cout<<"Location "<<p->Location<<endl;

cout<<"Contact Number "<<p->contactnum<<endl;

inorder(p->right);

}

}

int search(tree \*p,string loc){

if(p!=0){

inorder(p->left);

if(p->Location==loc){

return 1;

}

inorder(p->right);

}return 0;

}

};

//int main(){

// insert();

// inorder(root);

// int num;

// cout<<"\nEnter num to delete";

// cin>>num;

// deletenode(num);

// inorder(root);

//}

//que class

#include <iostream>

using namespace std;

class que{

public:

int size;

int \*q;

string \*s;

int front;

int rear;

int sfront;

int srear;

que(){

size = 10;

q = new int[size];

s = new string[size];

front =-1;

rear = -1;

sfront =-1;

srear = -1;

}

que(int x){

size =x;

q = new int[size];

s = new string[size];

front = -1;

rear = -1;

sfront =-1;

srear = -1;

}

bool is\_empty(){

if(rear==-1 && front == -1)

return true;

else

return false;

}

bool is\_sempty(){

if(srear==-1 && sfront == -1)

return true;

else

return false;

}

bool is\_full(){

if((rear+1)%size == front){

return true;

}else

return false;

}

bool is\_sfull(){

if((srear+1)%size == sfront){

return true;

}else

return false;

}

void display(){

if(is\_empty()){

cout<<"\nQueue Empty"<<endl;

}else {

int r = front;

while(r!=rear){

cout<<"call at index "<<r<<" is "<<q[r]<<endl;

r=(r+1)%size;

}cout<<"Call at index "<<r<<" is "<<q[r]<<endl;

}

}

int search(int key,string loc){

if(is\_sempty()){

return 0;

}else {

int r = sfront;

while(r!=srear){

if(q[r]==key && s[r]==loc){

return 1;

}

r=(r+1)%size;

}

}return 0;

}

void sdisplay(){

if(is\_sempty()){

cout<<"";

}else {

int r = sfront;

while(r!=srear){

cout<<"Value at index "<<s[r];

r=(r+1)%size;

}

}

}

void enqueue(int x){

if(is\_full()){

cout<<"\nQueue is full";

}

else{

if(is\_empty()){

front=rear=0;

}else{

rear = (rear + 1) % size;

}q[rear] = x;

}

}

void senqueue(string x){

if(is\_sfull()){

cout<<"\nQueue is full";

}

else{

if(is\_sempty()){

sfront=srear=0;

}else{

srear = (srear + 1) % size;

}s[srear] = x;

}

}

int dequeue(){

if(is\_empty()){

cout<<"\n";

return -1;

}else{

int x = q[front];

if(front==rear){

front=rear=-1;

}else{

front=(front+1) % size;

}

return x;

}

}

string sdequeue(){

if(is\_sempty()){

cout<<"\n";

return "";

}else{

string x = s[sfront];

if(sfront==srear){

sfront=srear=-1;

}else{

sfront=(sfront+1) % size;

}

return x;

}

}

};

//Policestation

#include <iostream>

using namespace std;

class policestation{

public:

int pmobile;

int bikesquad;

policestation(){

bikesquad=3;

pmobile=3;

}

};

//Firebrigade

#include <iostream>

using namespace std;

class Firebrigade{

public:

int firebrigade;

Firebrigade(){

firebrigade=3;

}

};

//Hospital

#include <iostream>

using namespace std;

class hospital{

public:

int ambulance;

hospital(){

ambulance=3;

}

};

//SSG class

#include <iostream>

using namespace std;

class SSg{

public:

int men;

SSg(){

men=20;

}

};

//convert class

#include <iostream>

using namespace std;

class convert{

public:convert(){

}

int stringconversion(string s){

if(s.compare("Shifa\_hospital")==0){

return 0;

}else if(s.compare("Paf\_hospital")==0){

return 1;

}else if(s.compare("Madina\_market")==0){

return 2;

}else if(s.compare("Raj\_shadi\_hall")==0){

return 3;

}else if(s.compare("Ali\_mobile\_plaza")==0){

return 4;

}else if(s.compare("ssg-qrf")==0){

return 5;

}else if(s.compare("pindi\_cricket\_stadium")==0){

return 6;

}else if(s.compare("Comsats\_university\_islamabad")==0){

return 7;

}else if(s.compare("Hostel\_city")==0){

return 8;

} else if(s.compare("Fire\_brigade\_station")==0){

return 9;

} else if(s.compare("Islamabad\_police\_station")==0){

return 10;

} else if (s.compare("Medical\_centre")==0){

return 11;

}else if(s.compare("sadqabad\_police\_station")==0){

return 12;

}else if(s.compare("Ayub\_nation\_park")==0){

return 13;

}else if(s.compare("ali\_pur")==0){

return 14;

}else if(s.compare("rawal\_hospital")==0){

return 15;

} else if(s.compare("Bani\_gala")==0){

return 16;

}else if(s.compare("Blue\_area")==0){

return 17;

} else if(s.compare("Jamia\_msjid")==0){

return 18;

} else if(s.compare("pindi\_firebrigade\_station")==0){

return 19;

}

else if(s.compare("quaid\_e\_azam\_hospital")==0){

return 20;

}

return 999;

}

string intconversion(int a){

if(a==0){

return "Shifa hospital";

}else if(a==1){

return "Paf hospital";

}

else if(a==2){

return "Madina market";

}

else if(a==3){

return "Raj shadi hall";

}

else if(a==4){

return "Ali mobile plaza";

}

else if(a==5){

return "ssg-qrf";

}

else if(a==6){

return "pindi cricket staium";

}

else if(a==7){

return "Comsats university islamabad";

}

else if(a==8){

return "Hostel city";

}

else if(a==9){

return "Fire\_brigade\_station";

}

else if(a==10){

return "Islamabad police station";

}

else if(a==11){

return "Medical centre";

}

else if(a==12){

return "sadqabad police station";

}

else if(a==13){

return "Ayub nation park";

}

else if(a==14){

return "ali pur";

}

else if(a==15){

return "rawal hospital";

}

else if(a==16){

return "Bani gala";

}

else if(a==17){

return "Blue area";

}

else if(a==18){

return "Jamia msjid";

}

else if(a==19){

return "pindi\_firebrigade\_station";

}

else if(a==20){

return "quaid e azam hospital";

}

return "";

}

};