**CODE:**

**MAP**

**#include<graphics.h>**

**#include<iostream>**

**#include<stdio.h>**

**#include<conio.h>**

**#include<string.h>**

**#include<cstdlib>**

**#include<string>**

**#include<fstream>**

**#include<sstream>**

**#include<algorithm>**

**#include<conio.h>**

**#include<process.h>**

**#include<math.h>**

**using namespace std;**

**void extra(int x1,int y1,int no) //function to check and mark area, with red/green or yellow.**

**{**

**if(no<5)**

**{**

**setfillstyle(1,2);**

**fillellipse(x1,y1,15,15);**

**}**

**else if(no>5 && no<10)**

**{**

**setfillstyle(1,14);**

**fillellipse(x1,y1,15,15);**

**}**

**else**

**{**

**setfillstyle(1,4);**

**fillellipse(x1,y1,15,15);**

**}**

**}**

**int main()**

**{**

**int gd=DETECT,gm;**

**int arr[]={48,75,45,99,51,116,52,136,52,158,49,169,60,185,59,198,38,209,10,231,10,238,17,240,76,245,80,253,74,262,70,276,60,284,47,296,35,306,45,329,45,343,50,356,72,344,82,364,60,387,67,398,88,398,99,404,111,400,124,400,132,403,139,405,137,414,**

**133,422,136,430,133,440,125,443,121,452,131,491,137,510,144,520,155,523,167,522,196,497,180,446,170,437,184,435,219,424,227,417,242,420,258,416,258,436,269,437,266,457,271,473,272,486,272,500,232,501,304,512,317,492,326,497,336,491,340,517,371,505,393,500,389,458,**

**400,418,433,426,461,440,470,416,486,411,499,399,504,382,511,367,522,346,532,340,551,333,551,313,516,314,514,295,487,290,488,276,487,260,513,252,500,230,508,209,518,195,534,197,559,192,566,185,558,181,558,166,534,130,522,116,529,106,515,104,521,78,527,69,490,87,**

**481,62,475,50,434,48,419,9,395,7,380,2,365,66,349,64,334,66,321,73,313,89,300,97,288,108,283,122,276,127,257,119,243,119,241,90,236,85,228,91,221,86,214,97,203,108,220,111,232,119,228,126,215,195,204,136,186,139,170,137,156,129,148,127,137,136,**

**126,143,118,143,114,135,116,122,111,114,102,111,90,100,88,87,86,80,79,74,73,77,61,75,48,75};**

**initgraph(&gd,&gm,"NULL");**

**rectangle(0,0,850,850);**

**drawpoly(145,arr);**

**settextstyle(8,0,2); //change the font of the text.**

**int x[30],y[30];// coordinates of the areas,**

**x[1]=64,y[1]=92,x[2]=79,y[2]=124,x[3]=115,y[3]=218,x[4]=155,y[4]=153;**

**x[5]=159,y[5]=301,x[6]=170,y[6]=420,x[7]=160,y[7]=504;x[8]=284,y[8]=438;**

**x[9]=362,y[9]=414,x[10]=449,y[10]=396,x[11]=488,y[11]=310,x[12]=271,y[12]=373,x[13]=313,y[13]=333;**

**x[14]=276,y[14]=259;**

**x[15]=396,y[15]=34,x[16]=480,y[16]=92,x[17]=459,y[17]=126,x[18]=515,y[18]=175,x[19]=472,y[19]=197;**

**x[20]=454,y[20]=236,x[21]=230,y[21]=106,x[22]=105,y[22]=364,x[23]=368,y[23]=171,x[24]=308,y[24]=479,x[25]=319,y[25]=120;**

**//x[25]=396,y[25]=34;**

**//int x16=284,y16=295,x17=346,y17=242,**

**///x[5]=128,y[5]=185;**

**// x[23]=334,y[23]=230; aissms ioit**

**int radius=12;**

**int sh[30]; // array for storing patient count in particular area.**

**for(int i=0;i<30;i++){**

**sh[i]=0;**

**}**

**int count=0;**

**ifstream corona("Entries.txt"); // open entries file.**

**if(!corona.is\_open())**

**{**

**cout << "Failed to open " << endl;**

**return 0 ;**

**}**

**string line;**

**while (!corona.eof())**

**{**

**while (getline(corona, line))**

**{ count++;**

**stringstream ss(line);**

**getline(ss,line, ',');**

**getline(ss,line, ',');**

**if(line=="balewadi") // match the area name with the count of patient**

**sh[1]++;**

**else if(line=="baner")**

**sh[2]++;**

**else if(line=="pashan")**

**sh[3]++;**

**else if(line=="aundh")**

**sh[4]++;**

**else if(line=="kothrud")**

**sh[5]++;**

**else if(line=="wadgaon")**

**sh[6]++;**

**else if(line=="dhayari")**

**sh[7]++;**

**else if(line=="dhankawadi")**

**sh[8]++;**

**else if(line=="katraj")**

**sh[24]++;**

**else if(line=="kondhava")**

**sh[9]++;**

**else if(line=="mohamadwadi")**

**sh[10]++;**

**else if(line=="hadapsar")**

**sh[11]++;**

**else if(line=="talajai")**

**sh[12]++;**

**else if(line=="swargate")**

**sh[13]++;**

**else if(line=="shivaji nagar")**

**sh[14]++;**

**else if(line=="khadki")**

**sh[25]++;**

**else if(line=="danori")**

**sh[15]++;**

**else if(line=="lohagaon")**

**sh[16]++;**

**else if(line=="viman nagar")**

**sh[17]++;**

**else if(line=="kharadi")**

**sh[18]++;**

**else if(line=="wadgaonsheri")**

**sh[19]++;**

**else if(line=="mundhava")**

**sh[20]++;**

**else if(line=="dapodi")**

**sh[21]++;**

**else if(line=="warje")**

**sh[22]++;**

**else if(line=="yerwada")**

**sh[23]++;**

**}**

**}**

**corona.close();**

**cout<<endl;**

**for(int i=1;i<26;i++) // function to draw circle around the area**

**{**

**if(sh[i]<5)**

**{**

**setfillstyle(1,2);**

**fillellipse(x[i],y[i],15,15);**

**}**

**else if(sh[i]>5 && sh[i]<10)**

**{**

**setfillstyle(1,14);**

**fillellipse(x[i],y[i],15,15);**

**}**

**else**

**{**

**setfillstyle(1,4);**

**fillellipse(x[i],y[i],15,15);**

**}}**

**setfillstyle(1,4);**

**extra(334,230,sh[14]); // mark extra points**

**extra(128,185,sh[4]);**

**outtextxy(64,92,"Balewadi"); // display text on graphics mode**

**outtextxy(79,124," @ Baner");**

**outtextxy(115,218,"pashan");**

**outtextxy(155,153,"\* Aundh");**

**outtextxy(128,185,"& Pune Univ.");**

**outtextxy(159,301,"# Kothrud");**

**outtextxy(170,420," @ Wadgaon");**

**outtextxy(160,504,"Dhayari");**

**outtextxy(284,438,"Dhankawadi");**

**outtextxy(308,479,"@ Katraj");**

**outtextxy(362,414,"Kondhava");**

**outtextxy(449,396,"Mohamadwadi");**

**outtextxy(488,310,"@\*Hadapsar" );**

**outtextxy(271,373,"Talajai");**

**outtextxy(313,333,"@#Swargate");**

**outtextxy(284,295,"PUNE");**

**outtextxy(276,259,"@&Shivaji Nagar");**

**outtextxy(319,120,"#Khadki");**

**outtextxy(396,34," @ Danori");**

**outtextxy(480,92,"Lohagaon");**

**outtextxy(459,126,"@ Viman Nagar");**

**outtextxy(515,175,"Kharadi");**

**outtextxy(472,197,"Wadgaonsheri");**

**outtextxy(456,236,"& Mundhava");**

**outtextxy(230,106,"Dapodi");**

**outtextxy(105,364,"\*Warje");**

**outtextxy(300,230,"#!AISSMS IOIT");**

**outtextxy(368,171,"Yerwada");**

**outtextxy(500,490,"@ = Sahyadri Hospital");**

**outtextxy(500,512,"# = Ruby Hospital");**

**outtextxy(500,536,"\* = Nobel Hospital");**

**outtextxy(500,554,"& = KEM Hospital");**

**outtextxy(500,572,"! = NAIDU Hospital");**

**getch();**

**closegraph();**

**return 0;**

**}**

**MATHS**

**#include<iostream>**

**#include<conio.h>**

**#include<stdio.h>**

**#include<string.h>**

**#include<cstdlib>**

**#include<string>**

**#include<fstream>**

**#include<sstream>**

**#include<algorithm>**

**#include<process.h>**

**#include<iomanip>**

**#include<math.h>**

**#include<sstream>**

**#include <graphics.h>**

**using namespace std;**

**struct node**

**{**

**string date;**

**string id;**

**int active\_cases;**

**int no=0;**

**int death,recover;**

**node \*next;**

**};**

**int decimal\_to\_binary(int n) //function to convert decimal to binary**

**{**

**long long binaryNumber = 0;**

**int remainderr, i = 1, step = 1;**

**while (n!=0)**

**{**

**remainderr = n%2;**

**n = n/2;**

**binaryNumber = binaryNumber + remainderr\*i;**

**i = i\*10;**

**}**

**return (binaryNumber);**

**}**

**int binary\_to\_decimal(long int n) //function to convert binary to decimal**

**{**

**int decimalNumber = 0, i = 0, remainderr;**

**while (n!=0)**

**{**

**remainderr = n%10;**

**n =n/ 10;**

**decimalNumber = decimalNumber + remainderr\*pow(2,i);**

**i++;**

**}**

**return (decimalNumber);**

**}**

**int binary\_addition(long int bn1 ,long int bn2) // fun for binary addition**

**{**

**int i=0, r=0,arr[20];**

**long int sum[20];**

**while (bn1 != 0 || bn2 != 0)**

**{**

**sum[i++] = (int)((bn1 % 10 + bn2 % 10 + r) % 2);**

**r = (int)((bn1 % 10 + bn2 % 10 + r) / 2);**

**bn1 = bn1 / 10;**

**bn2 = bn2 / 10;**

**}**

**if (r != 0) {**

**sum[i++] = r;**

**}**

**int k=--i;**

**int count=0;**

**for(int a=k;a>=0;)**

**{**

**arr[a]=sum[a--];**

**count++;**

**}**

**stringstream ss;**

**for ( int i = count-2; i>=-1; i--)**

**ss<<arr[i];**

**long int result=0;**

**ss>>result;**

**return (result);**

**}**

**class PATIENTS**

**{**

**int active,addition;**

**long int existing\_b , active\_b , new\_existing;**

**public:**

**node \*head,\*temp,\*temp1;**

**int dead ,cured;**

**int len=0;**

**void getdata();**

**int calculation(int );**

**void accept();**

**void acceptg(string);**

**void operation(PATIENTS);**

**void display();**

**void count(PATIENTS);**

**void count1(PATIENTS);**

**void count2(PATIENTS);**

**void final();**

**void length();**

**void getim(int \*a,int \*b, int \*c,int\*d);**

**};**

**void PATIENTS::display()**

**{**

**temp1=head;**

**while(temp1!=NULL)**

**{**

**cout<<temp1->date<<"-"<<temp1->active\_cases<<"-"<<temp1->no<<"-"<<temp1->death<<"-"<<temp1->recover<<"\n";**

**temp1=temp1->next;**

**}**

**}**

**void PATIENTS::operation(PATIENTS pat)**

**{**

**node \*h1,\*tempv;**

**int set=0;**

**char str[100];**

**h1=pat.head;**

**head=NULL;**

**while(h1!=NULL)**

**{**

**tempv=head;**

**while(tempv!=NULL)**

**{**

**if(tempv->date==h1->date)**

**{**

**set=set+1;**

**break;**

**}**

**else**

**set=0;**

**tempv=tempv->next;**

**}**

**if(set==0)**

**{ strcpy(str,h1->date.c\_str());**

**temp = new node;**

**temp->date=str;**

**temp->next = NULL;**

**if (head == NULL)**

**{**

**head = temp;**

**temp1 = temp;**

**}**

**else**

**{**

**while (temp1->next != NULL)**

**temp1 = temp1->next;**

**temp1->next = temp;**

**}**

**}**

**h1=h1->next;**

**}**

**}**

**void PATIENTS::count(PATIENTS p)**

**{**

**node \*h1;**

**string in\_use="0";**

**int set=0;**

**h1=p.head;**

**temp1=head;**

**while(temp1!=NULL)**

**{**

**h1=p.head;**

**while(h1!=NULL)**

**{**

**if(h1->id!=in\_use)**

**{**

**if(temp1->date==h1->date)**

**set=set+1;**

**}**

**h1=h1->next;**

**}**

**temp1->no=set;**

**temp1=temp1->next;**

**set=0;**

**}**

**}**

**void PATIENTS::count1(PATIENTS p)**

**{**

**node \*h1;**

**int set=0;**

**h1=p.head;**

**temp1=head;**

**while(temp1!=NULL)**

**{**

**h1=p.head;**

**while(h1!=NULL)**

**{**

**if(temp1->date==h1->date)**

**set=set+1;**

**h1=h1->next;**

**}**

**temp1->recover=set;**

**temp1=temp1->next;**

**set=0;**

**}**

**}**

**void PATIENTS::count2(PATIENTS p)**

**{**

**node \*h1;**

**int set=0;**

**int c=0;**

**h1=p.head;**

**temp1=head;**

**while(temp1!=NULL)**

**{**

**h1=p.head;**

**while(h1!=NULL)**

**{**

**if(temp1->date==h1->date)**

**set=set+1;**

**h1=h1->next;**

**}**

**temp1->death=set;**

**temp1=temp1->next;**

**set=0;**

**}**

**}**

**void PATIENTS::accept() // accept entries from txt**

**{**

**string line,useful;**

**char sit[20];**

**ifstream read("Entries.txt");**

**if(!read.is\_open())**

**{**

**cout << "Failed to open " << endl;**

**}**

**head=NULL;**

**while (!read.eof())**

**{**

**while (getline(read, line))**

**{**

**temp = new node;**

**temp->next = NULL;**

**stringstream ss(line);**

**getline(ss, useful, ',');**

**getline(ss,useful, ',');**

**getline(ss,temp->id, ',');**

**getline(ss,useful, ',');**

**getline(ss,useful, ',');**

**getline(ss,useful, ',');**

**getline(ss,temp->date, ',');**

**if (head == NULL)**

**{**

**head = temp;**

**temp1 = temp;**

**}**

**else**

**{**

**while (temp1->next != NULL)**

**temp1 = temp1->next;**

**temp1->next = temp;**

**}**

**}**

**}**

**read.close();**

**}**

**void PATIENTS::acceptg(string name) // accept entries from txt.**

**{**

**string line,useful;**

**char sit[20];**

**ifstream read(name.c\_str());**

**if(!read.is\_open())**

**{**

**cout << "Failed to open " << endl;**

**}**

**head=NULL;**

**while (!read.eof())**

**{**

**while (getline(read, line))**

**{**

**temp = new node;**

**temp->next = NULL;**

**stringstream ss(line);**

**getline(ss, useful, ',');**

**getline(ss,useful, ',');**

**getline(ss,useful, ',');**

**getline(ss,useful, ',');**

**getline(ss,useful, ',');**

**getline(ss,useful, ',');**

**getline(ss,useful, ',');**

**getline(ss,temp->date, ',');**

**if (head == NULL)**

**{**

**head = temp;**

**temp1 = temp;**

**}**

**else**

**{**

**while (temp1->next != NULL)**

**temp1 = temp1->next;**

**temp1->next = temp;**

**}**

**}**

**}**

**read.close();**

**}**

**void PATIENTS::getdata()**

**{**

**active=temp1->no;**

**dead=temp1->death;**

**cured=temp1->recover;**

**}**

**int PATIENTS::calculation(int existing) // converting decimal to binary performing calculation and reconverting it.**

**{**

**active\_b = decimal\_to\_binary (active);**

**existing\_b = decimal\_to\_binary (existing);**

**addition = binary\_addition (existing\_b , active\_b);**

**new\_existing = binary\_to\_decimal (addition);**

**new\_existing = new\_existing - (dead+cured);**

**return (new\_existing);**

**}**

**void PATIENTS::final() // final function for display and all fun calls.**

**{**

**temp1=head;**

**int existing=0 ,list[len]={};**

**int i=0;**

**string ARRAY[len];**

**while(temp1!=NULL && i!=(len+1))**

**{ARRAY[i]=temp1->date;**

**i++;**

**temp1=temp1->next;**

**}**

**temp1=head;**

**for(int i=0;i<len,temp1!=NULL;i++,temp1=temp1->next)**

**{**

**getdata();**

**list[i]=calculation(existing);**

**existing=list[i];**

**}**

**cout<<"\n\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_..\_.\_.\n";**

**cout<<"\n DATE\t\t\t ACTIVE CASES\t\tREGISTERED CASES \t\tDEAD\t\t\tCURED\n";**

**temp1=head;**

**for(int j=0;j<len,temp1!=NULL;j++,temp1=temp1->next)**

**{**

**cout<<ARRAY[j]<<"\t\t\t"<<list[j]<<"\t\t\t "<<temp1->no<<"\t\t\t"<<temp1->death<<"\t\t\t "<<temp1->recover<<endl;**

**}**

**temp1=head;**

**cout<<"\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_..\_.\_.\n";**

**for(int i=0;i<len,temp1!=NULL;i++,temp1=temp1->next)**

**{**

**temp1->active\_cases=list[i];**

**}**

**}**

**void PATIENTS::length()**

**{**

**for(temp1=head;temp1!=NULL;temp1=temp1->next)**

**{**

**len++;**

**}**

**}**

**void PATIENTS::getim(int \*a,int \*b, int \*c,int\*d) // assign no with values.**

**{**

**temp1=head;**

**while(temp1!=NULL)**

**{**

**\*a+=temp1->no;**

**\*b=temp1->active\_cases;**

**\*c+=temp1->recover;**

**\*d+=temp1->death;**

**temp1=temp1->next;**

**}**

**}**

**int main()**

**{**

**string name;**

**PATIENTS pat,pat1;**

**pat.accept();**

**pat1.operation(pat);**

**pat1.count(pat);**

**name="recover.txt";**

**pat.acceptg(name);**

**pat1.count1(pat);**

**name="death.txt";**

**pat.acceptg(name);**

**pat1.count2(pat);**

**pat1.length();**

**pat1.final();**

**int A=0,B=0,C=0,D=0;**

**int \*a,\*b,\*c,\*d;**

**int t,m;**

**a=&A;**

**b=&B;**

**c=&C;**

**d=&D;**

**pat1.getim(a,b,c,d);**

**t=100-B;**

**m=t;**

**if(m<0)**

**{**

**t=0;**

**m=(-1)\*m;**

**}**

**else**

**{**

**m=0;**

**}**

**int gd=DETECT,gm;**

**initgraph(&gd,&gm,NULL);**

**char msg[128],SETA[15],SETB[15],SETC[15],SETD[15],HBR[15],PUIH[15];**

**sprintf(HBR, "=%d", t); // write char string at specified point.**

**outtextxy(388,484, HBR);**

**sprintf(PUIH, "=%d", m);**

**outtextxy(485,509, PUIH);**

**sprintf(SETA, "%d", A);**

**outtextxy(190,376, SETA);**

**sprintf(SETB, "%d", B);**

**outtextxy(190,394, SETB);**

**sprintf(SETC, "%d", C);**

**outtextxy(190,412, SETC);**

**sprintf(SETD, "%d", D);**

**outtextxy(190,430, SETD);**

**rectangle(150,100,450,200);**

**setfillstyle(3, 2);**

**fillellipse(200,150,25,25);**

**setfillstyle(3, 3);**

**fillellipse(300,150,25,25);**

**setfillstyle(3, 4);**

**fillellipse(400,150,25,25);**

**settextstyle(8,0,2);**

**outtextxy(450,75,"A");**

**outtextxy(200,102,"B");**

**outtextxy(300,102,"C");**

**outtextxy(400,102,"D");**

**outtextxy(100,268,"SET A= REGISTERED CASES WHICH IS A UNIVERSAL SET");**

**outtextxy(100,286,"SET B= ACTIVE CASES ");**

**outtextxy(100,304,"SET C= RECOVERED CASES");**

**outtextxy(100,322,"SET D= DEATH CASES");**

**outtextxy(100,340,"SET A = SET B U SET C U SET D");**

**outtextxy(100,358," |B| = |A| - (|C| + |D|)");**

**outtextxy(100,376,"SET A=");**

**outtextxy(190,376,SETA);**

**outtextxy(100,394,"SET B=");**

**outtextxy(190,394,SETB);**

**outtextxy(100,412,"SET C=");**

**outtextxy(190,412,SETC);**

**outtextxy(100,430,"SET D=");**

**outtextxy(190,430,SETD);**

**outtextxy(100,448,"Alloted Hospital Beds =100");**

**outtextxy(100,466,"Patiens active =");**

**outtextxy(292,466,SETB);**

**outtextxy(100,484,"Hospital Beds Remaining =");**

**outtextxy(388,484,HBR);**

**outtextxy(100,509,"PATIENTS UNALLOCATED IN HOSPITAL ");**

**outtextxy(485,509,PUIH);**

**getch();**

**return 0;**

**}**

**OOP**

**#include<iostream>**

**#include<conio.h>**

**#include<stdio.h>**

**#include<string.h>**

**#include<cstdlib>**

**#include<string>**

**#include<fstream>**

**#include<sstream>**

**#include<algorithm>**

**#include<process.h>**

**#include<iomanip>**

**#include<math.h>**

**using namespace std;**

**// CREATING all the structures for active, dead and recovered patients**

**struct node**

**{**

**string name,addr,sex,age,health;**

**double id;**

**string temp;**

**string datee;**

**node \*next;**

**}\*head,\*temp,\*temp1;**

**struct del**

**{**

**string name,addr,sex,age,health;**

**double id;**

**string datee,ddatee;**

**string temp;**

**del \*next;**

**}\*start=NULL,\*temporary,\*use;**

**struct rec**

**{**

**string name,addr,sex,age,health;**

**double id;**

**string datee,ddatee;**

**string temp;**

**rec \*next;**

**}\*go,\*normal,\*creat;**

**class sll**

**{**

**public:**

**string line;**

**void create();**

**void display();**

**void search();**

**void write();**

**void deleting();**

**void recover();**

**void accept();**

**void acceptd();**

**void acceptr();**

**void writer();**

**void writed();**

**};**

**void sll::accept() //accept for accepting previous entries**

**{**

**int count;**

**char sit[20];**

**ifstream read("Entries.txt");**

**if(!read.is\_open())**

**{**

**cout << "Failed to open " << endl;**

**}**

**head=NULL;**

**while (!read.eof())**

**{**

**while (getline(read, line))**

**{**

**count++;**

**temp = new node;**

**temp->next = NULL;**

**stringstream ss(line);**

**getline(ss, temp->name, ',');**

**getline(ss,temp->addr, ',');**

**getline(ss,temp->temp, ',');**

**strcpy(sit ,temp->temp.c\_str());**

**temp-> id = atoi(sit );**

**getline(ss,temp->age, ',');**

**getline(ss,temp->sex, ',');**

**getline(ss,temp->health, ',');**

**getline(ss,temp->datee, ',');**

**cout<<endl;**

**if (head == NULL)**

**{**

**head = temp;**

**temp1 = temp;**

**}**

**else**

**{**

**while (temp1->next != NULL)**

**temp1 = temp1->next;**

**temp1->next = temp;**

**}**

**}**

**}**

**read.close();**

**}**

**void sll::acceptr() //accept for accepting previous entries**

**{**

**{**

**char cit[20];**

**ifstream dea("recover.txt");**

**if(!dea.is\_open())**

**{**

**cout << "Failed to open " << endl;**

**}**

**start=NULL;**

**while (!dea.eof())**

**{**

**while (getline(dea, line))**

**{**

**creat = new rec;**

**creat->next = NULL;**

**stringstream ss(line);**

**getline(ss, creat->name, ',');**

**getline(ss,creat->addr, ',');**

**getline(ss,creat->temp, ',');**

**strcpy(cit ,creat->temp.c\_str());**

**creat-> id = atoi(cit );**

**getline(ss,creat->age, ',');**

**getline(ss,creat->sex, ',');**

**getline(ss,creat->health, ',');**

**getline(ss,creat->datee, ',');**

**getline(ss,creat->ddatee, ',');**

**if (start == NULL)**

**{**

**go = creat;**

**normal = creat;**

**}**

**else**

**{**

**while (normal->next != NULL)**

**normal = normal->next;**

**normal->next = creat;**

**}**

**}**

**}**

**dea.close();**

**}**

**}**

**void sll::writed() //writing into the file**

**{**

**ofstream arn;**

**arn.open("death.txt");**

**if(!arn.is\_open())**

**exit(EXIT\_FAILURE);**

**for(temporary=start;temporary!=NULL;temporary=temporary->next)**

**{arn<<temporary->name<<","<<temporary->addr<<","<<temporary->id<<","<<temporary->age<<","<<temporary->sex<<","<<temporary->health<<","<<temporary->datee<<","<<temporary->ddatee<<endl;}**

**arn.close();**

**}**

**void sll::writer() //writing into the file**

**{**

**ofstream arn;**

**arn.open("recover.txt");**

**if(!arn.is\_open())**

**exit(EXIT\_FAILURE);**

**for(normal=go;normal!=NULL;normal=normal->next)**

**{arn<<normal->name<<","<<normal->addr<<","<<normal->id<<","<<normal->age<<","<<normal->sex<<","<<normal->health<<","<<normal->datee<<","<<normal->ddatee<<endl;}**

**arn.close();**

**}**

**void sll::acceptd() //accept for accepting previous entries**

**{**

**char cit[20];**

**ifstream dea("death.txt");**

**if(!dea.is\_open())**

**{**

**cout << "Failed to open " << endl;**

**}**

**start=NULL;**

**while (!dea.eof())**

**{**

**while (getline(dea, line))**

**{**

**use = new del;**

**use->next = NULL;**

**stringstream ss(line);**

**getline(ss, use->name, ',');**

**getline(ss,use->addr, ',');**

**getline(ss,use->temp, ',');**

**strcpy(cit ,use->temp.c\_str());**

**use-> id = atoi(cit );**

**getline(ss,use->age, ',');**

**getline(ss,use->sex, ',');**

**getline(ss,use->health, ',');**

**getline(ss,use->datee, ',');**

**getline(ss,use->ddatee, ',');**

**if (start == NULL)**

**{**

**start = use;**

**temporary = use;**

**}**

**else**

**{**

**while (temporary->next != NULL)**

**temporary = temporary->next;**

**temporary->next = use;**

**}**

**}**

**}**

**dea.close();**

**}**

**void sll::write() //writing into the file**

**{**

**ofstream corona;**

**corona.open("Entries.txt");**

**if(!corona.is\_open())**

**exit(EXIT\_FAILURE);**

**for(temp1=head;temp1!=NULL;temp1=temp1->next)**

**{corona<<temp1->name<<","<<temp1->addr<<","<<temp1->id<<","<<temp1->age<<","<<temp1->sex<<","<<temp1->health<<","<<temp1->datee<<endl;}**

**corona.close();**

**}**

**void sll::create() // new entries**

**{**

**string dt;**

**cin.ignore();**

**cout<<"Enter date: ";**

**cin>>dt;**

**char name[100];**

**char ch;**

**do**

**{**

**temp=new node;**

**cin.ignore();**

**cout<<"Enter Patients name: ";**

**getline(cin,temp->name);**

**cout<<"Enter Patients address: ";**

**getline(cin,temp->addr);**

**cout<<"Enter Patients id: ";**

**cin>>temp->id;**

**cin.ignore();**

**cout<<"Enter Patient age: ";**

**getline(cin,temp->age);**

**cout<<"Enter Patients gender: ";**

**getline(cin,temp->sex);**

**cout<<"Patients Health Status: ";**

**getline(cin,temp->health);**

**strcpy(name,dt.c\_str());**

**temp->datee=std::string(name);**

**//strcpy(temp->datee,dt.c\_str());**

**cout<<endl;**

**temp->next=NULL;**

**temp1=head;**

**if(head==NULL)**

**{**

**head=temp;**

**temp1=head;**

**}**

**else**

**{**

**while(temp1->next!=NULL)**

**temp1=temp1->next;**

**temp1->next=temp;**

**}**

**cout<<"Do u wants to add more Patients: ";**

**cin>>ch;**

**cout<<endl;**

**}while(ch=='y');**

**}**

**void sll::display()**

**{**

**int count=0;**

**cout<<" #########################################################################"<<endl;**

**cout<<" !\*!\*!# # # # ## # ###### ### # # ##### !\*!\*!"<<endl;**

**cout<<" !\*!\*! # # # # # ##### # # # # # !\*!\*!"<<endl;**

**cout<<" !\*!\*! # # # #### # # ####### # # # ### !\*!\*!"<<endl;**

**cout<<" !\*!\*! # # # # # # # ##### # # # !\*!\*!"<<endl;**

**cout<<" !\*!\*! # # # # ## ###### ###### ### # # ##### !\*!\*!"<<endl;**

**cout<<" #########################################################################"<<endl;**

**cout<<" #########################################################################"<<endl;**

**cout<<"\n";**

**cout<<"############################################################################################################"<<endl;**

**cout<<"# SR.# PATIENT # PATIENT # PATIENT # GENDER # PATIENT # HEALTH # DATE OF #"<<endl;**

**cout<<"# NO.# NAME # ADDRESS # ID # [M/F/T] # AGE # STATUS # ADMISSION #"<<endl;**

**cout<<"############################################################################################################"<<endl;**

**temp1=head;**

**while(temp1!=NULL)**

**{count++;**

**cout<<"# "<<setw(2)<<count<<" #"<<setw(19)<<temp1->name<<setw(4)<<"#"<<setw(18)<<temp1->addr<<setw(3)<<"#"<<setw(8)<<temp1->id<<setw(2)<<"#"<<setw(9)<<temp1->sex<<setw(2)<<"#"<<setw(7)<<temp1->age <<setw(6)<<"#"<<setw(8)<<temp1->health<<setw(3)<<"#"<<setw(11)<<temp1->datee<<setw(2)<<"#"<<"\n";**

**temp1=temp1->next;**

**}**

**cout<<"############################################################################################################"<<endl;**

**}**

**void sll::search() // search patient on the basis of his/her name**

**{**

**char find[100];**

**int count=0;**

**int set =0;**

**cout<<"\n Enter Patient NAME you want to search: ";**

**cin>>find;**

**cout<<"##############################################################################################################"<<endl;**

**cout<<"# SR. # PATIENT # PATIENT # PATIENT # GENDER # PATIENT # HEALTH # DATE OF #"<<endl;**

**cout<<"# NO. # NAME # ADDRESS # ID # [M/F/T] # AGE # STATUS # ADMISSION #"<<endl;**

**cout<<"##############################################################################################################"<<endl;**

**temp=head;**

**while(temp!=NULL)**

**{**

**std::size\_t found = temp->name.find(find);**

**if(found!=std::string::npos)**

**{count++;**

**cout<<"# "<<setw(2)<<count<<" #"<<setw(19)<<temp->name<<setw(4)<<"#"<<setw(18)<<temp->addr<<setw(3)<<"#"<<setw(8)<<temp->id<<setw(2)<<"#"<<setw(9)<<temp->sex<<setw(2)<<"#"<<setw(7)<<temp->age <<setw(6)<<"#"<<setw(8)<<temp->health<<setw(3)<<"#"<<setw(11)<<temp->datee<<setw(2)<<"#"<<"\n";**

**set=1;**

**}**

**temp=temp->next;**

**}**

**if(set!=1)**

**cout<<"\t\t\t"<<"PATIENT NOT FOUND"<<endl;**

**cout<<"##############################################################################################################"<<endl;**

**cout<<"\n";**

**}**

**void sll::deleting() //dead patients calculation**

**{**

**string dt;**

**cin.ignore();**

**cout<<"Enter Todays date: ";**

**cin>>dt;**

**char name[100];**

**double no;**

**char ch;**

**node \*vtemp;**

**cin.ignore();**

**do{cout<<"\nEnter ID of the person decieved: ";**

**cin>>no;**

**cout<<endl;**

**temp1=head;**

**while(temp1->id!=no)**

**{**

**vtemp=temp1;**

**temp1=temp1->next;**

**}**

**if(temp1==head)**

**{**

**use =new del;**

**use->name=temp1->name;**

**use->addr=temp1->addr;**

**use->id=temp1->id;**

**use->age=temp1->age;**

**use->sex=temp1->sex;**

**use->health=temp1->health;**

**use->datee=temp1->datee;**

**strcpy(name,dt.c\_str());**

**use->ddatee=std::string(name);**

**use->next=NULL;**

**if(start==NULL)**

**{**

**start=use;**

**temporary=start;**

**}**

**else**

**{**

**while(temporary->next!=NULL)**

**temporary=temporary->next;**

**temporary->next=use;**

**}**

**temp1=temp1->next;**

**delete head;**

**head=temp1;**

**}**

**else if(temp1->next==NULL)**

**{**

**use =new del;**

**use->name=temp1->name;**

**use->addr=temp1->addr;**

**use->id=temp1->id;**

**use->age=temp1->age;**

**use->sex=temp1->sex;**

**use->health=temp1->health;**

**use->datee=temp1->datee;**

**strcpy(name,dt.c\_str());**

**use->ddatee=std::string(name);**

**use->next=NULL;**

**if(start==NULL)**

**{**

**start=use;**

**temporary=start;**

**}**

**else**

**{**

**while(temporary->next!=NULL)**

**temporary=temporary->next;**

**temporary->next=use;**

**}**

**vtemp->next=NULL;**

**delete temp1;**

**}**

**else**

**{**

**use =new del;**

**use->name=temp1->name;**

**use->addr=temp1->addr;**

**use->id=temp1->id;**

**use->age=temp1->age;**

**use->sex=temp1->sex;**

**use->health=temp1->health;**

**use->datee=temp1->datee;**

**strcpy(name,dt.c\_str());**

**use->ddatee=std::string(name);**

**use->next=NULL;**

**if(start==NULL)**

**{**

**start=use;**

**temporary=start;**

**}**

**else**

**{**

**while(temporary->next!=NULL)**

**temporary=temporary->next;**

**temporary->next=use;**

**}**

**vtemp->next=temp1->next;**

**delete temp1;**

**}**

**cout<<"Enter \'y\' if you Want to delete more Patients: ";**

**cin>>ch;}while(ch=='y');**

**}**

**void sll::recover() //recover patients calculation**

**{**

**string dt;**

**cin.ignore();**

**cout<<"Enter Todays date: ";**

**cin>>dt;**

**char name[100];**

**double no;**

**char h;**

**node \*vtemp;**

**do{**

**cout<<"\nEnter id of the person who is recovered: ";**

**cin>>no;**

**cout<<endl;**

**temp1=head;**

**while(temp1->id!=no)**

**{**

**vtemp=temp1;**

**temp1=temp1->next;**

**}**

**if(temp1==head)**

**{**

**creat = new rec;**

**creat->name=temp1->name;**

**creat->addr=temp1->addr;**

**creat->id=temp1->id;**

**creat->age=temp1->age;**

**creat->sex=temp1->sex;**

**creat->health=temp1->health;**

**creat->datee=temp1->datee;**

**strcpy(name,dt.c\_str());**

**creat->ddatee=std::string(name);**

**creat->next=NULL;**

**if(go==NULL)**

**{**

**go=creat;**

**normal=go;**

**}**

**else**

**{**

**while(normal->next!=NULL)**

**normal=normal->next;**

**normal->next=creat;**

**}**

**temp1=temp1->next;**

**delete head;**

**head=temp1;**

**}**

**else if(temp1->next==NULL)**

**{**

**creat = new rec;**

**creat->name=temp1->name;**

**creat->addr=temp1->addr;**

**creat->id=temp1->id;**

**creat->age=temp1->age;**

**creat->sex=temp1->sex;**

**creat->health=temp1->health;**

**creat->datee=temp1->datee;**

**strcpy(name,dt.c\_str());**

**creat->ddatee=std::string(name);**

**creat->next=NULL;**

**if(go==NULL)**

**{**

**go=creat;**

**normal=go;**

**}**

**else**

**{**

**while(normal->next!=NULL)**

**normal=normal->next;**

**normal->next=creat;**

**}**

**vtemp->next=NULL;**

**delete temp1;**

**}**

**else**

**{**

**creat = new rec;**

**creat->name=temp1->name;**

**creat->addr=temp1->addr;**

**creat->id=temp1->id;**

**creat->age=temp1->age;**

**creat->sex=temp1->sex;**

**creat->health=temp1->health;**

**creat->datee=temp1->datee;**

**strcpy(name,dt.c\_str());**

**creat->ddatee=std::string(name);**

**creat->next=NULL;**

**if(go==NULL)**

**{**

**go=creat;**

**normal=go;**

**}**

**else**

**{**

**while(normal->next!=NULL)**

**normal=normal->next;**

**normal->next=creat;**

**}**

**vtemp->next=temp1->next;**

**delete temp1;**

**}**

**cout<<"Enter \'y\' if you Want to delete more Patients: ";**

**cin>>h;}while(h=='y');**

**}**

**int main()**

**{**

**sll s;**

**int value,d;**

**int ch;**

**s.accept();**

**s.display();**

**s.acceptd();**

**s.acceptr();**

**do**

**{**

**cout<<" 1.add a record\n 2.Modify a record\n 3.search\n 4.exit "<<endl;**

**cout<<" Enter your choice: ";**

**cin>>ch;**

**cout<<endl;**

**switch(ch)**

**{**

**case 1:**

**{**

**s.create();**

**s.display();**

**break;**

**}**

**case 2:**

**{**

**cout<<"For person who is decieved press 0 or For a discharge press 1: ";**

**cin>>d;**

**if(d==0)**

**{**

**s.deleting();**

**s.display();**

**}**

**else**

**{**

**s.recover();**

**s.display();**

**}**

**break;**

**}**

**case 3:**

**{**

**s.search();**

**break;**

**}**

**case 4:**

**{**

**s.write ();**

**s.writed();**

**s.writer();**

**exit(0);**

**}**

**default:**

**cout<<"enter choice between 1....4 only!"<<endl;**

**}**

**}while(ch!=4);**

**getch();**

**return 0;**

**}**

**DELD**

**#include<iostream>**

**#include<conio.h>**

**#include<stdio.h>**

**#include<string.h>**

**#include<cstdlib>**

**#include<string>**

**#include<fstream>**

**#include<sstream>**

**#include<algorithm>**

**#include<process.h>**

**#include<iomanip>**

**#include<math.h>**

**#include<sstream>**

**using namespace std;**

**struct node**

**{**

**string date;**

**string id;**

**int active\_cases;**

**int no=0;**

**int death,recover;**

**node \*next;**

**};**

**int decimal\_to\_binary(int n)**

**{**

**long long binaryNumber = 0;**

**int remainderr, i = 1, step = 1;**

**while (n!=0)**

**{**

**remainderr = n%2;**

**n = n/2;**

**binaryNumber = binaryNumber + remainderr\*i;**

**i = i\*10;**

**}**

**return (binaryNumber);**

**}**

**int binary\_to\_decimal(long int n)**

**{**

**int decimalNumber = 0, i = 0, remainderr;**

**while (n!=0)**

**{**

**remainderr = n%10;**

**n =n/ 10;**

**decimalNumber = decimalNumber + remainderr\*pow(2,i);**

**i++;**

**}**

**return (decimalNumber);**

**}**

**int binary\_addition(long int bn1 ,long int bn2)**

**{**

**int i=0, r=0,arr[20];**

**long int sum[20];**

**while (bn1 != 0 || bn2 != 0)**

**{**

**sum[i++] = (int)((bn1 % 10 + bn2 % 10 + r) % 2);**

**r = (int)((bn1 % 10 + bn2 % 10 + r) / 2);**

**bn1 = bn1 / 10;**

**bn2 = bn2 / 10;**

**}**

**if (r != 0) {**

**sum[i++] = r;**

**}**

**int k=--i;**

**int count=0;**

**for(int a=k;a>=0;)**

**{**

**arr[a]=sum[a--];**

**count++;**

**}**

**stringstream ss;**

**for ( int i = count-2; i>=-1; i--)**

**ss<<arr[i];**

**long int result=0;**

**ss>>result;**

**return (result);**

**}**

**class PATIENTS**

**{**

**int active,addition;**

**long int existing\_b , active\_b , new\_existing;**

**public:**

**node \*head,\*temp,\*temp1;**

**int dead ,cured;**

**int len=0;**

**void getdata();**

**int calculation(int );**

**void write();**

**void accept();**

**void acceptg(string);**

**void operation(PATIENTS);**

**void display();**

**void count(PATIENTS);**

**void count1(PATIENTS);**

**void count2(PATIENTS);**

**void final();**

**void length();**

**};**

**void PATIENTS::display()**

**{**

**temp1=head;**

**while(temp1!=NULL)**

**{**

**cout<<temp1->date<<"-"<<temp1->active\_cases<<"-"<<temp1->no<<"-"<<temp1->death<<"-"<<temp1->recover<<"\n";**

**temp1=temp1->next;**

**}**

**}**

**void PATIENTS::operation(PATIENTS pat)**

**{**

**node \*h1,\*tempv;**

**int set=0;**

**char str[100];**

**h1=pat.head;**

**head=NULL;**

**while(h1!=NULL)**

**{**

**tempv=head;**

**while(tempv!=NULL)**

**{**

**if(tempv->date==h1->date)**

**{**

**set=set+1;**

**break;**

**}**

**else**

**set=0;**

**tempv=tempv->next;**

**}**

**if(set==0)**

**{ strcpy(str,h1->date.c\_str());**

**temp = new node;**

**temp->date=str;**

**temp->next = NULL;**

**if (head == NULL)**

**{**

**head = temp;**

**temp1 = temp;**

**}**

**else**

**{**

**while (temp1->next != NULL)**

**temp1 = temp1->next;**

**temp1->next = temp;**

**}**

**}**

**h1=h1->next;**

**}**

**}**

**void PATIENTS::count(PATIENTS p)**

**{**

**node \*h1;**

**string in\_use="0";**

**int set=0;**

**h1=p.head;**

**temp1=head;**

**while(temp1!=NULL)**

**{**

**h1=p.head;**

**while(h1!=NULL)**

**{**

**if(h1->id!=in\_use)**

**{**

**if(temp1->date==h1->date)**

**set=set+1;**

**}**

**h1=h1->next;**

**}**

**temp1->no=set;**

**temp1=temp1->next;**

**set=0;**

**}**

**}**

**void PATIENTS::count1(PATIENTS p)**

**{**

**node \*h1;**

**int set=0;**

**h1=p.head;**

**temp1=head;**

**while(temp1!=NULL)**

**{**

**h1=p.head;**

**while(h1!=NULL)**

**{**

**if(temp1->date==h1->date)**

**set=set+1;**

**h1=h1->next;**

**}**

**temp1->recover=set;**

**temp1=temp1->next;**

**set=0;**

**}**

**}**

**void PATIENTS::count2(PATIENTS p)**

**{**

**node \*h1;**

**int set=0;**

**int c=0;**

**h1=p.head;**

**temp1=head;**

**while(temp1!=NULL)**

**{**

**h1=p.head;**

**while(h1!=NULL)**

**{**

**if(temp1->date==h1->date)**

**set=set+1;**

**h1=h1->next;**

**}**

**temp1->death=set;**

**temp1=temp1->next;**

**set=0;**

**}**

**}**

**void PATIENTS::write()**

**{**

**ofstream corona;**

**corona.open("patients\_alloted.txt");**

**if(!corona.is\_open())**

**exit(EXIT\_FAILURE);**

**corona<<" DATE\t\t\t ACTIVE CASES\t\tREGISTERED CASES \t\tDEAD\t\t\tCURED\n";;**

**for(temp1=head;temp1!=NULL;temp1=temp1->next)**

**{**

**corona<<temp1->date<<"\t\t\t "<<temp1->active\_cases<<"\t\t\t "<<temp1->no<<"\t\t\t "<<temp1->death<<"\t\t\t "<<temp1->recover<<endl;**

**}**

**corona.close();**

**}**

**void PATIENTS::accept()**

**{**

**string line,useful;**

**char sit[20];**

**ifstream read("Entries.txt");**

**if(!read.is\_open())**

**{**

**cout << "Failed to open " << endl;**

**}**

**head=NULL;**

**while (!read.eof())**

**{**

**while (getline(read, line))**

**{**

**temp = new node;**

**temp->next = NULL;**

**stringstream ss(line);**

**getline(ss, useful, ',');**

**getline(ss,useful, ',');**

**getline(ss,temp->id, ',');**

**getline(ss,useful, ',');**

**getline(ss,useful, ',');**

**getline(ss,useful, ',');**

**getline(ss,temp->date, ',');**

**if (head == NULL)**

**{**

**head = temp;**

**temp1 = temp;**

**}**

**else**

**{**

**while (temp1->next != NULL)**

**temp1 = temp1->next;**

**temp1->next = temp;**

**}**

**}**

**}**

**read.close();**

**}**

**void PATIENTS::acceptg(string name)**

**{**

**string line,useful;**

**char sit[20];**

**ifstream read(name.c\_str());**

**if(!read.is\_open())**

**{**

**cout << "Failed to open " << endl;**

**}**

**head=NULL;**

**while (!read.eof())**

**{**

**while (getline(read, line))**

**{**

**temp = new node;**

**temp->next = NULL;**

**stringstream ss(line);**

**getline(ss, useful, ',');**

**getline(ss,useful, ',');**

**getline(ss,useful, ',');**

**getline(ss,useful, ',');**

**getline(ss,useful, ',');**

**getline(ss,useful, ',');**

**getline(ss,useful, ',');**

**getline(ss,temp->date, ',');**

**if (head == NULL)**

**{**

**head = temp;**

**temp1 = temp;**

**}**

**else**

**{**

**while (temp1->next != NULL)**

**temp1 = temp1->next;**

**temp1->next = temp;**

**}**

**}**

**}**

**read.close();**

**}**

**void PATIENTS::getdata()**

**{**

**active=temp1->no;**

**dead=temp1->death;**

**cured=temp1->recover;**

**}**

**int PATIENTS::calculation(int existing)**

**{**

**active\_b = decimal\_to\_binary (active);**

**existing\_b = decimal\_to\_binary (existing);**

**addition = binary\_addition (existing\_b , active\_b);**

**new\_existing = binary\_to\_decimal (addition);**

**new\_existing = new\_existing - (dead+cured);**

**return (new\_existing);**

**}**

**void PATIENTS::final()**

**{**

**temp1=head;**

**int existing=0 ,list[len]={};**

**int i=0;**

**string ARRAY[len];**

**while(temp1!=NULL && i!=(len+1))**

**{ARRAY[i]=temp1->date;**

**i++;**

**temp1=temp1->next;**

**}**

**temp1=head;**

**for(int i=0;i<len,temp1!=NULL;i++,temp1=temp1->next)**

**{**

**getdata();**

**list[i]=calculation(existing);**

**existing=list[i];**

**}**

**cout<<"\n\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_..\_.\_.\n";**

**cout<<"\n DATE\t\t\t ACTIVE CASES\t\tREGISTERED CASES \t\tDEAD\t\t\tCURED\n";**

**temp1=head;**

**for(int j=0;j<len,temp1!=NULL;j++,temp1=temp1->next)**

**{**

**cout<<ARRAY[j]<<"\t\t\t"<<list[j]<<"\t\t\t "<<temp1->no<<"\t\t\t"<<temp1->death<<"\t\t\t "<<temp1->recover<<endl;**

**}**

**temp1=head;**

**cout<<"\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_.\_..\_.\_.\n";**

**for(int i=0;i<len,temp1!=NULL;i++,temp1=temp1->next)**

**{**

**temp1->active\_cases=list[i];**

**}**

**}**

**void PATIENTS::length()**

**{**

**for(temp1=head;temp1!=NULL;temp1=temp1->next)**

**{**

**len++;**

**}**

**}**

**int main()**

**{**

**string name;**

**PATIENTS pat,pat1;**

**pat.accept();**

**pat1.operation(pat);**

**pat1.count(pat);**

**name="recover.txt";**

**pat.acceptg(name);**

**pat1.count1(pat);**

**name="death.txt";**

**pat.acceptg(name);**

**pat1.count2(pat);**

**pat1.length();**

**pat1.final();**

**//pat1.display();**

**pat1.write();**

**getch();**

**return 0;**

**}**