

Project Report

Hospital Management System

**Submitted to:**

Dr Md Rifat Ahmmad Rashid(PhD)

Assistant Professor of CSE department

of ULAB

**Submitted by:**

Mahazabin Khan id (193014034)

                                             Morium Begum id(201014072)

Rezowana Zayed Zahin id( 201014096)

Md. Mizanur Rahman id(201014061)

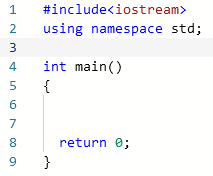
   Md. Yusuf Hossain id(201014089)

**Project Overview:**

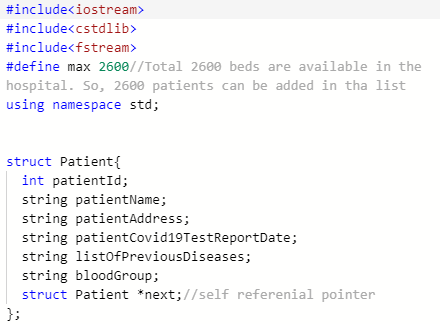
The main object of the C++ project on Hospital Management System is to manage Patient’s information i.e. patient id, patient name, patient address, patient covid-19 test report date, patient previous diseases and blood group which is stored in the patient record book. User can insert these data, search a patient, update patient’s information, see patient list etc. with digital method which is called digital medical record. User can save patient information permanently through file handling. User can see the history about the covid-19 dedicated hospital Dhaka Medical College Hospital (DMC).

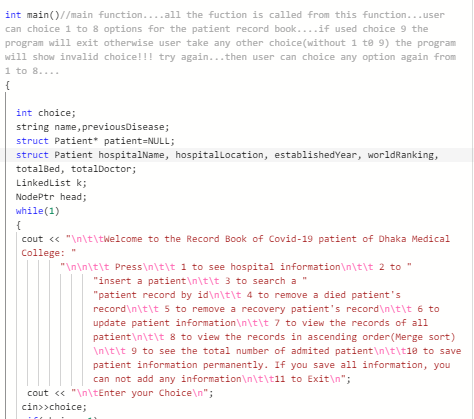
**Data Structure Programming concept:**

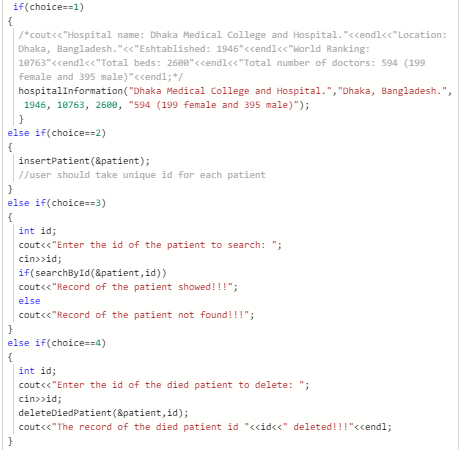
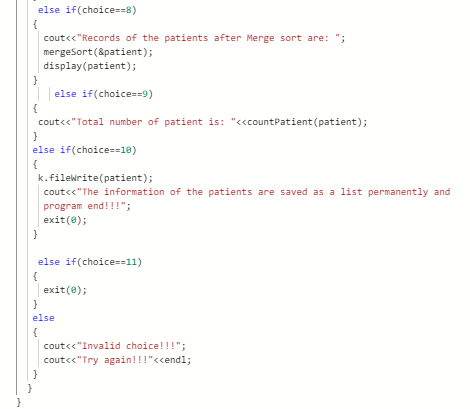
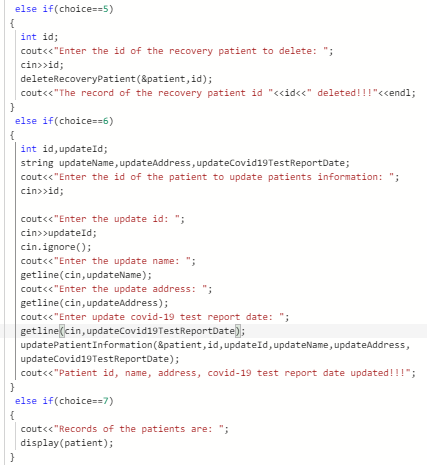
We have used structure, stack, merge sort based on singly linked list with dynamic memory allocation to develop the hospital management system properly. There are a proper flow of all functions and its parameters.

**1**. At first, we have create the basic structure of C++ program. After that we develop the code for getting proper output of tic tac toe game.

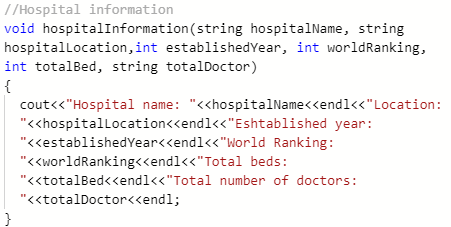
**2.** Then we have created struct name Patient which contains patientId, patientName, , patient address, patient covid-19 test report date, patient previous diseases and blood group and a self-referential pointer named next.



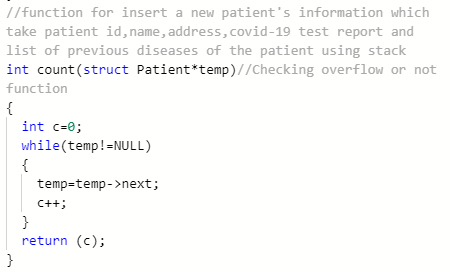
**3.** We have created a main function and all function will call from the main function.

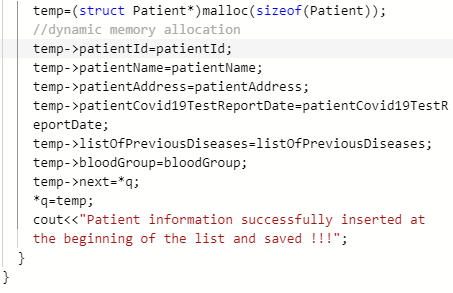
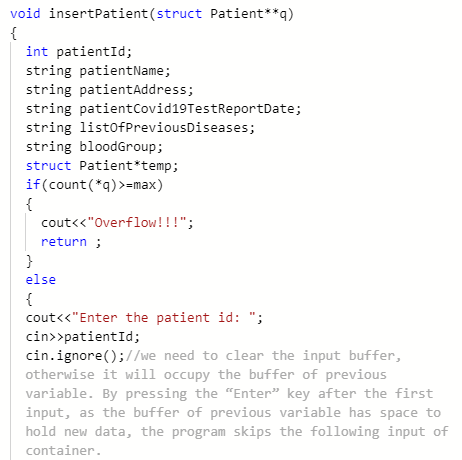
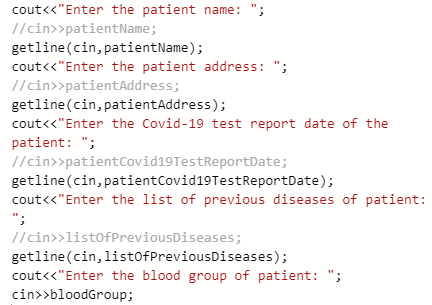


**4.** We have created a void type function to get the information about the Covid-19 dedicated hospital.

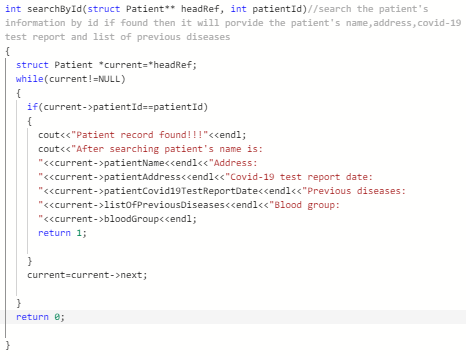
****

**5.** Then we have created a count function which will check the stack is overflow or not. If the bed fill up that means total number of bed 2600 fill up, user cannot include any patient in the hospital or record book any program will show overflow. Because the capacity of the hospital is 2600.

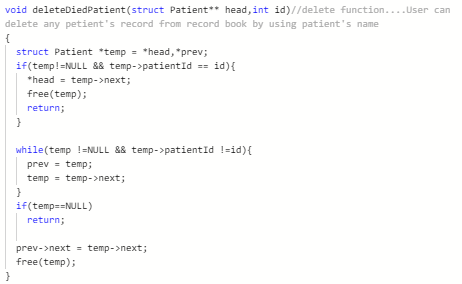


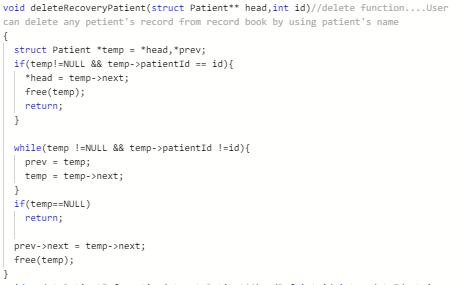
**6.** After that we used an insert function for inserting a patient where user can input of the patient with dynamically and its specialty is user can input with space of all string type data. There we use cin.ignore(). After taking id which is an integer type data because need to clear the input buffer, otherwise it will occupy the buffer of previous variable. By pressing the “Enter” key after the first input, as the buffer of previous variable has space to hold new data, the program skips the following input of container. Then it will show information successfully inserted.

**7.** Then we have designed the program with a search function named searchById where user can search a patient by using id. If the patient admitted in the hospital, it will show the record found and will show the full information of the patient. Otherwise, the function will record not found.



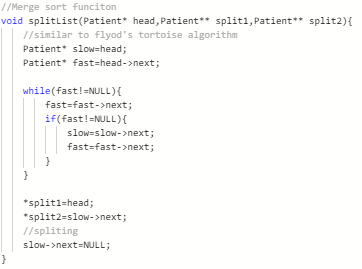
**8.** User can delete the information of died patient by using patient’s id through deleteDiedPatient function.

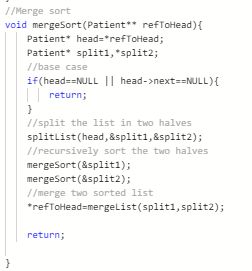


**9.** User also can delete the information of recovery patient by using patient’s id through deleteRecoveryPatient function.

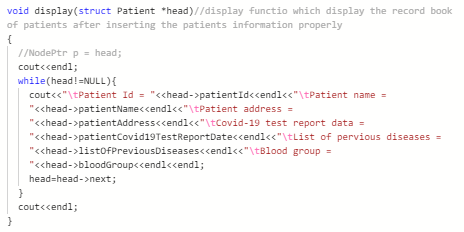
**10.** Then user can update patient’s information. At first user should use patient’s id to find the patient the user can update the information through updatePatientInformation function.



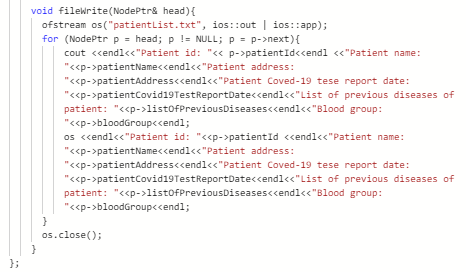
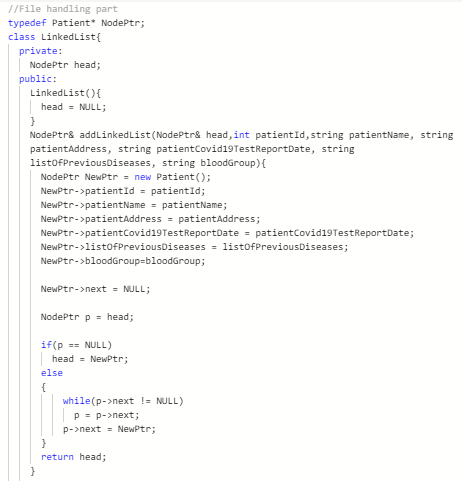
**11.** After that we have designed a merge sort function to merge the patient by using patient’s id according ascending order.

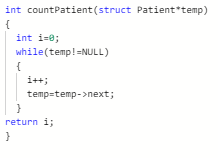


**12.** Then we have used display to function to display the information of the patients which is inserted in the record book.



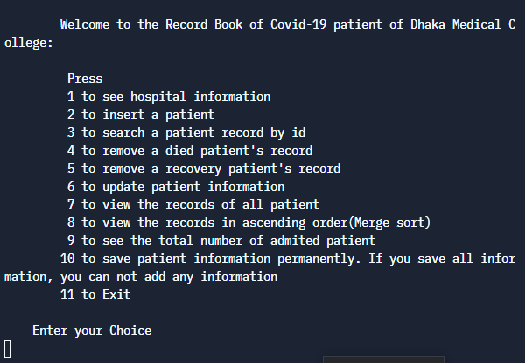
**13.** We have used class and object to save all the information of the patients permanently. User should use it at the end of the program because if user save all the information once then the program will end.



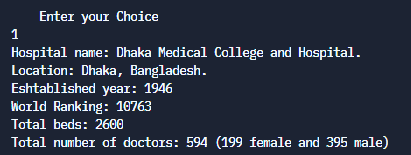
**14.** At the last we have used a count function name countPatient function to get the total number of patient of the record book.

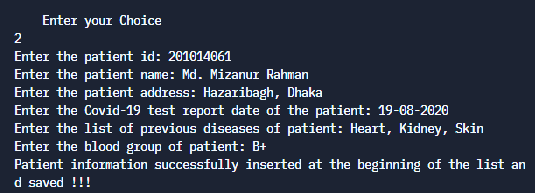
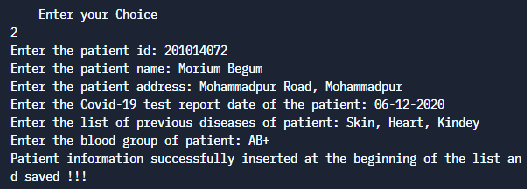
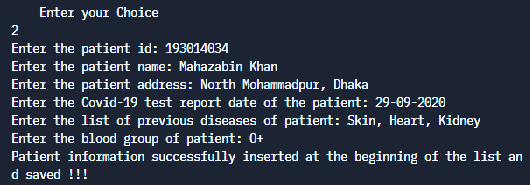
**Output:**

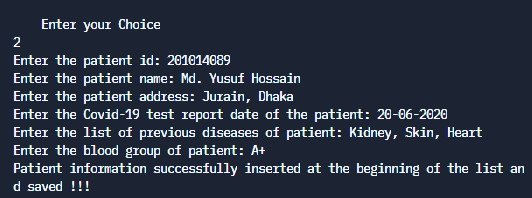
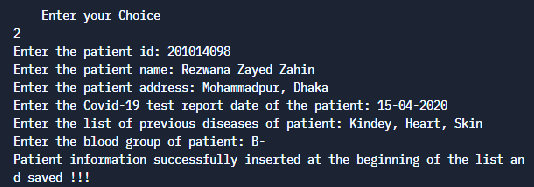
**1.** If we want to see the output of the program where used will get a choice list. User can take any choice from the choice list. If user take an invalid choice, program will show invalid choice!!! Try again!!!

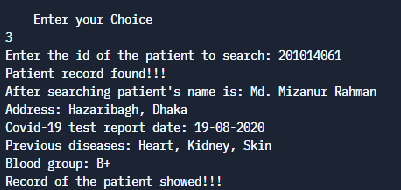


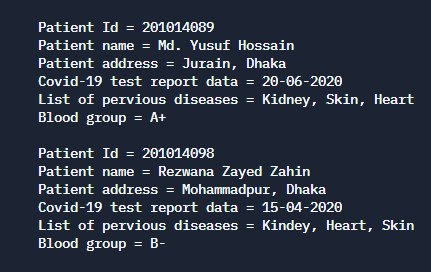
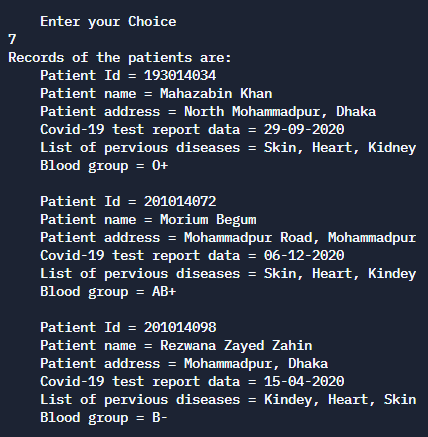
**2.** If user take choice 1, user will see the hospital information.

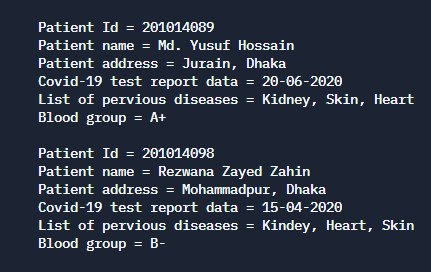
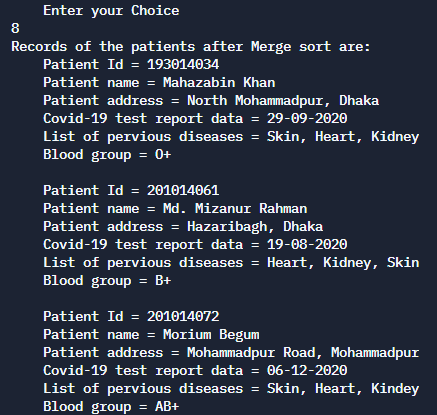


**3.** If user take choice 2, user can insert a new patient in the hospital which will save in at the beginning of the record list.

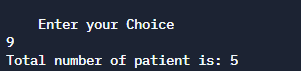


**4.** If user take choice 3, user can search a patient by using patient’s id. Then user will see all the information of the patient.

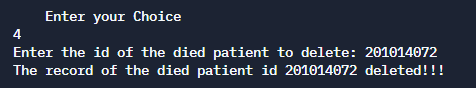
******5.** If user take choice 7, user will all information of all patient.

******6.** If user take choice 8, user will all information of all patient in ascending order where have used merge sort.

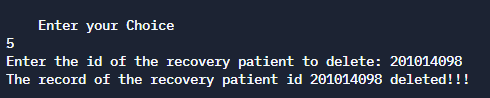
**7.** If user take choice 9, user will see how many patients are admitted in the hospital.

****

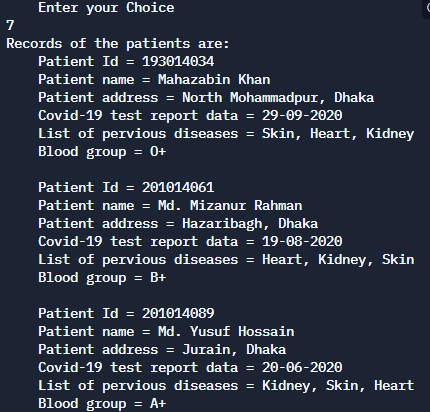
**8.** If user take choice 4, user can delete the information of died patient. So, user should user patient id to find the died patient information.



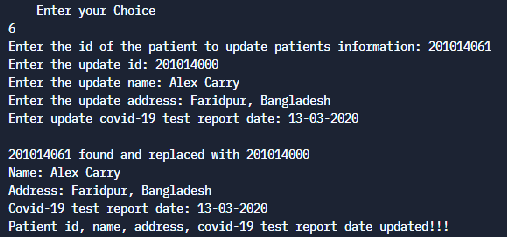
**9.** If user take choice 5, user can delete the information of recovery patient. So, there also user should patient id to find the recovery patient information.



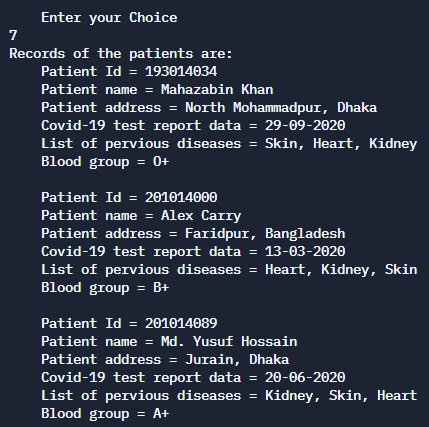
**10.** Then user can see updated record book after deleting died and recovery patient.



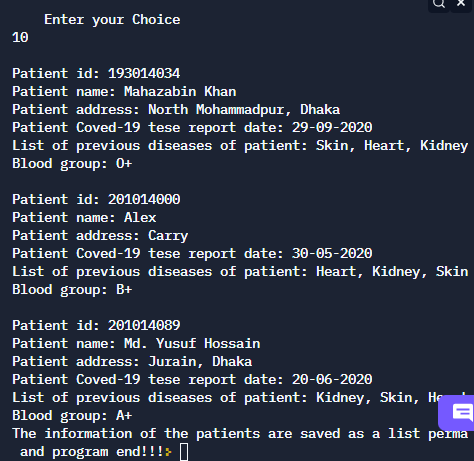
**11.** If user take choice 6, user can update patient’s information. So, user should use patient id to find the old information. After finding user can input update information of the patient.



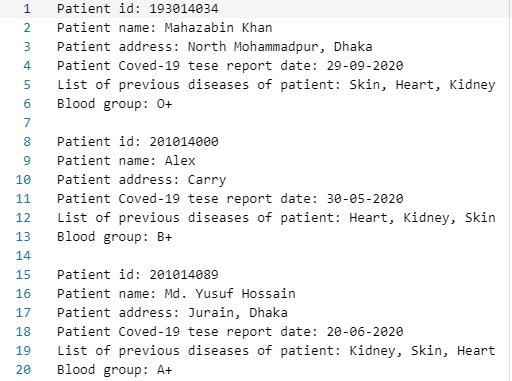
Then user will see the update record book by taking the choice 7. There user will the information is updated.



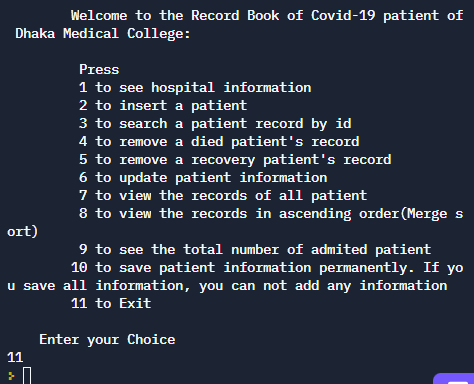
**12.** If user take choice 10, the information of all patient will save in the permanent storage and program will show the information of all patient are saved as a list and program end. User can use it one time. If user choice it once, the program will end.



User will see a permanent record in the storage where user saved the information.



**13.** If user take choice 11, the program will end.



**Contribution:**

Repl link: <https://replit.com/@MdMizanurRahman/Data-structure-Lab-Project-group-4>

Source code:

#include<iostream>

#include<cstdlib>

#include<fstream>

#define max 2600//Total 2600 beds are available in the hospital. So, 2600 patients can be added in tha list

using namespace std;

struct Patient{

int patientId;

string patientName;

string patientAddress;

string patientCovid19TestReportDate;

string listOfPreviousDiseases;

string bloodGroup;

struct Patient \*next;//self referenial pointer

};

//Hospital information

void hospitalInformation(string hospitalName, string hospitalLocation,int establishedYear, int worldRanking, int totalBed, string totalDoctor)

{

cout<<"Hospital name: "<<hospitalName<<endl<<"Location: "<<hospitalLocation<<endl<<"Eshtablished year: "<<establishedYear<<endl<<"World Ranking: "<<worldRanking<<endl<<"Total beds: "<<totalBed<<endl<<"Total number of doctors: "<<totalDoctor<<endl;

}

//function for insert a new patient's information which take patient id,name,address,covid-19 test report and list of previous diseases of the patient using stack

int count(struct Patient\*temp)//Checking overflow or not function

{

int c=0;

while(temp!=NULL)

{

temp=temp->next;

c++;

}

return (c);

}

void insertPatient(struct Patient\*\*q)

{

int patientId;

string patientName;

string patientAddress;

string patientCovid19TestReportDate;

string listOfPreviousDiseases;

string bloodGroup;

struct Patient\*temp;

if(count(\*q)>=max)

{

cout<<"Overflow!!!";

return ;

}

else

{

cout<<"Enter the patient id: ";

cin>>patientId;

cin.ignore();//we need to clear the input buffer, otherwise it will occupy the buffer of previous variable. By pressing the “Enter” key after the first input, as the buffer of previous variable has space to hold new data, the program skips the following input of container.

cout<<"Enter the patient name: ";

//cin>>patientName;

getline(cin,patientName);

cout<<"Enter the patient address: ";

//cin>>patientAddress;

getline(cin,patientAddress);

cout<<"Enter the Covid-19 test report date of the patient: ";

//cin>>patientCovid19TestReportDate;

getline(cin,patientCovid19TestReportDate);

cout<<"Enter the list of previous diseases of patient: ";

//cin>>listOfPreviousDiseases;

getline(cin,listOfPreviousDiseases);

cout<<"Enter the blood group of patient: ";

cin>>bloodGroup;

temp=(struct Patient\*)malloc(sizeof(Patient));//dynamic memory allocation

temp->patientId=patientId;

temp->patientName=patientName;

temp->patientAddress=patientAddress;

temp->patientCovid19TestReportDate=patientCovid19TestReportDate;

temp->listOfPreviousDiseases=listOfPreviousDiseases;

temp->bloodGroup=bloodGroup;

temp->next=\*q;

\*q=temp;

cout<<"Patient information successfully inserted at the beginning of the list and saved !!!";

}

}

int searchById(struct Patient\*\* headRef, int patientId)//search the patient's information by id if found then it will porvide the patient's name,address,covid-19 test report and list of previous diseases

{

struct Patient \*current=\*headRef;

while(current!=NULL)

{

if(current->patientId==patientId)

{

cout<<"Patient record found!!!"<<endl;

cout<<"After searching patient's name is: "<<current->patientName<<endl<<"Address: "<<current->patientAddress<<endl<<"Covid-19 test report date: "<<current->patientCovid19TestReportDate<<endl<<"Previous diseases: "<<current->listOfPreviousDiseases<<endl<<"Blood group: "<<current->bloodGroup<<endl;

return 1;

}

current=current->next;

}

return 0;

}

void deleteDiedPatient(struct Patient\*\* head,int id)//delete function....User can delete any petient's record from record book by using patient's name

{

struct Patient \*temp = \*head,\*prev;

if(temp!=NULL && temp->patientId == id){

\*head = temp->next;

free(temp);

return;

}

while(temp !=NULL && temp->patientId !=id){

prev = temp;

temp = temp->next;

}

if(temp==NULL)

return;

prev->next = temp->next;

free(temp);

}

void deleteRecoveryPatient(struct Patient\*\* head,int id)//delete function....User can delete any petient's record from record book by using patient's name

{

struct Patient \*temp = \*head,\*prev;

if(temp!=NULL && temp->patientId == id){

\*head = temp->next;

free(temp);

return;

}

while(temp !=NULL && temp->patientId !=id){

prev = temp;

temp = temp->next;

}

if(temp==NULL)

return;

prev->next = temp->next;

free(temp);

}

void updatePatientInformation(struct Patient\*\*headRef,int id,int updateId,string updateName,string updateAddress,string updateCovid19TestReportDate)//update funcition by address...user can update any patient's address by using the previous address

{

if(\*headRef==NULL)

{

cout<<"Patient record file is empty!!!";

}

struct Patient \*current=\*headRef;

while(current!=NULL)

{

if(current->patientId==id)

{

current->patientId=updateId;

current->patientName=updateName;

current->patientAddress=updateAddress;

current->patientCovid19TestReportDate=updateCovid19TestReportDate;

cout<<endl<<id<<" found and replaced with "<<updateId<<endl<<"Name: "<<updateName<<endl<<"Address: "<<updateAddress<<endl<<"Covid-19 test report date: "<<updateCovid19TestReportDate<<endl;

}

current=current->next;

}

}

//Merge sort function for getting the patient information in accending order

Patient\* mergeList(Patient\* split1,Patient\* split2){

//merging two sorted list

Patient\* newhead=NULL;

//base cases

if(split1==NULL)

return split2;

if(split2==NULL)

return split1;

//recursively merge

if(split1->patientId<=split2->patientId){

newhead=split1;

newhead->next=mergeList(split1->next,split2);

}

else{

newhead=split2;

newhead->next=mergeList(split1,split2->next);

}

return newhead;

}

//Merge sort funciton

void splitList(Patient\* head,Patient\*\* split1,Patient\*\* split2){

//similar to flyod's tortoise algorithm

Patient\* slow=head;

Patient\* fast=head->next;

while(fast!=NULL){

fast=fast->next;

if(fast!=NULL){

slow=slow->next;

fast=fast->next;

}

}

\*split1=head;

\*split2=slow->next;

//spliting

slow->next=NULL;

}

//Merge sort

void mergeSort(Patient\*\* refToHead){

Patient\* head=\*refToHead;

Patient\* split1,\*split2;

//base case

if(head==NULL || head->next==NULL){

return;

}

//split the list in two halves

splitList(head,&split1,&split2);

//recursively sort the two halves

mergeSort(&split1);

mergeSort(&split2);

//merge two sorted list

\*refToHead=mergeList(split1,split2);

return;

}

void display(struct Patient \*head)//display functio which display the record book of patients after inserting the patients information properly

{

//NodePtr p = head;

cout<<endl;

while(head!=NULL){

cout<<"\tPatient Id = "<<head->patientId<<endl<<"\tPatient name = "<<head->patientName<<endl<<"\tPatient address = "<<head->patientAddress<<endl<<"\tCovid-19 test report data = "<<head->patientCovid19TestReportDate<<endl<<"\tList of pervious diseases = "<<head->listOfPreviousDiseases<<endl<<"\tBlood group = "<<head->bloodGroup<<endl<<endl;

head=head->next;

}

cout<<endl;

}

//File handling part

typedef Patient\* NodePtr;

class LinkedList{

private:

NodePtr head;

public:

LinkedList(){

head = NULL;

}

NodePtr& addLinkedList(NodePtr& head,int patientId,string patientName, string patientAddress, string patientCovid19TestReportDate, string listOfPreviousDiseases, string bloodGroup){

NodePtr NewPtr = new Patient();

NewPtr->patientId = patientId;

NewPtr->patientName = patientName;

NewPtr->patientAddress = patientAddress;

NewPtr->patientCovid19TestReportDate = patientCovid19TestReportDate;

NewPtr->listOfPreviousDiseases = listOfPreviousDiseases;

NewPtr->bloodGroup=bloodGroup;

NewPtr->next = NULL;

NodePtr p = head;

if(p == NULL)

head = NewPtr;

else

{

while(p->next != NULL)

p = p->next;

p->next = NewPtr;

}

return head;

}

void fileWrite(NodePtr& head){

ofstream os("patientList.txt", ios::out | ios::app);

for (NodePtr p = head; p != NULL; p = p->next){

cout <<endl<<"Patient id: "<< p->patientId<<endl <<"Patient name: "<<p->patientName<<endl<<"Patient address: "<<p->patientAddress<<endl<<"Patient Coved-19 tese report date: "<<p->patientCovid19TestReportDate<<endl<<"List of previous diseases of patient: "<<p->listOfPreviousDiseases<<endl<<"Blood group: "<<p->bloodGroup<<endl;

os <<endl<<"Patient id: "<<p->patientId <<endl<<"Patient name: "<<p->patientName<<endl<<"Patient address: "<<p->patientAddress<<endl<<"Patient Coved-19 tese report date: "<<p->patientCovid19TestReportDate<<endl<<"List of previous diseases of patient: "<<p->listOfPreviousDiseases<<endl<<"Blood group: "<<p->bloodGroup<<endl;

}

os.close();

}

};

int countPatient(struct Patient\*temp)

{

int i=0;

while(temp!=NULL)

{

i++;

temp=temp->next;

}

return i;

}

int main()//main function....all the fuction is called from this function...user can choice 1 to 8 options for the patient record book....if used choice 9 the program will exit otherwise user take any other choice(without 1 t0 9) the program will show invalid choice!!! try again...then user can choice any option again from 1 to 8....

{

int choice;

string name,previousDisease;

struct Patient\* patient=NULL;

struct Patient hospitalName, hospitalLocation, establishedYear, worldRanking, totalBed, totalDoctor;

LinkedList k;

NodePtr head;

while(1)

{

cout << "\n\t\tWelcome to the Record Book of Covid-19 patient of Dhaka Medical College: "

"\n\n\t\t Press\n\t\t 1 to see hospital information\n\t\t 2 to "

"insert a patient\n\t\t 3 to search a "

"patient record by id\n\t\t 4 to remove a died patient's record\n\t\t 5 to remove a recovery patient's record\n\t\t 6 to update patient information\n\t\t 7 to view the records of all patient\n\t\t 8 to view the records in ascending order(Merge sort)\n\t\t 9 to see the total number of admited patient\n\t\t10 to save patient information permanently. If you save all information, you can not add any information\n\t\t11 to Exit\n";

cout << "\n\tEnter your Choice\n";

cin>>choice;

if(choice==1)

{

/\*cout<<"Hospital name: Dhaka Medical College and Hospital."<<endl<<"Location: Dhaka, Bangladesh."<<"Eshtablished: 1946"<<endl<<"World Ranking: 10763"<<endl<<"Total beds: 2600"<<endl<<"Total number of doctors: 594 (199 female and 395 male)"<<endl;\*/

hospitalInformation("Dhaka Medical College and Hospital.","Dhaka, Bangladesh.", 1946, 10763, 2600, "594 (199 female and 395 male)");

}

else if(choice==2)

{

insertPatient(&patient);

//user should take unique id for each patient

}

else if(choice==3)

{

int id;

cout<<"Enter the id of the patient to search: ";

cin>>id;

if(searchById(&patient,id))

cout<<"Record of the patient showed!!!";

else

cout<<"Record of the patient not found!!!";

}

else if(choice==4)

{

int id;

cout<<"Enter the id of the died patient to delete: ";

cin>>id;

deleteDiedPatient(&patient,id);

cout<<"The record of the died patient id "<<id<<" deleted!!!"<<endl;

}

else if(choice==5)

{

int id;

cout<<"Enter the id of the recovery patient to delete: ";

cin>>id;

deleteRecoveryPatient(&patient,id);

cout<<"The record of the recovery patient id "<<id<<" deleted!!!"<<endl;

}

else if(choice==6)

{

int id,updateId;

string updateName,updateAddress,updateCovid19TestReportDate;

cout<<"Enter the id of the patient to update patients information: ";

cin>>id;

cout<<"Enter the update id: ";

cin>>updateId;

cin.ignore();

cout<<"Enter the update name: ";

getline(cin,updateName);

cout<<"Enter the update address: ";

getline(cin,updateAddress);

cout<<"Enter update covid-19 test report date: ";

getline(cin,updateCovid19TestReportDate);

updatePatientInformation(&patient,id,updateId,updateName,updateAddress,updateCovid19TestReportDate);

cout<<"Patient id, name, address, covid-19 test report date updated!!!";

}

else if(choice==7)

{

cout<<"Records of the patients are: ";

display(patient);

}

else if(choice==8)

{

cout<<"Records of the patients after Merge sort are: ";

mergeSort(&patient);

display(patient);

}

else if(choice==9)

{

cout<<"Total number of patient is: "<<countPatient(patient);

}

else if(choice==10)

{

k.fileWrite(patient);

cout<<"The information of the patients are saved as a list permanently and program end!!!";

exit(0);

}

else if(choice==11)

{

exit(0);

}

else

{

cout<<"Invalid choice!!!";

cout<<"Try again!!!"<<endl;

}

}

}