//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// HOSPITAL MANAGEMENT DATABASE

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//Created by Nikolay Zahariev

//Assists hospital staff in creating a database to store information on patients and the kinds of illnesses they are suffering from.

//The user will be able to both input info on new patience that suffer from one of those conditions or will be able to search for already sighed-in patience.

//Furthermore, users can input data (name, ID, specialty) on available doctors.

#include "stdafx.h"

#include <iostream>

#include <conio.h>

#include <stdlib.h>

#include <string>

#include <fstream>

#include <list>

#include <cctype>

using namespace std;

fstream myfile;

// Used to validate user input. This function is called when you want to check if the user

// entered only integers. If he did not then the function will continue to ask for new input

// until the user enters a correct value.

int valid\_input\_int()

{

int intg;

string input;

bool valid;

do

{

getline(cin, input); // get a line of string input

valid = true; // assume it's valid

if (input != "") { // check if the string entered is empty

for (auto &i : input) // check each character in the input string

{

if (!isdigit(i)) // is it an numberical character?

{

valid = false; // if not, mark it as invalid

cout << "Invalid input. Please input only numberical characters: "; // print an error to the user

break; // break out of the for() loop, as we have already established the input is invalid

}

}

}

else {

cout << "Invalid input. Please input only numberical characters: ";

valid = false; // mark input as invalid

}

} while (!valid); // keep going until we get input that is valid

intg = atoi(input.c\_str()); //convert the string that has only digits in it into an integer variable

return intg; // return the integer variable now that it contains only digits

}

// Used to validate user input. This function is called when you want to check if the user

// entered only characters in the range [a-zA-Z]. If he did not then the function will continue to ask for new input

// until the user enters a correct value.

string getLetters()

{

string input;

bool valid;

do

{

getline(cin, input); // get a line of input

valid = true; // assume it's valid

if (input != "") { // check if the string entered is empty

for (auto &i : input) // check each character in the input string

{

if (!isalpha(i)) // is it an alphabetical character?

{

valid = false; // if not, mark it as invalid

cout << "Invalid input. Please input only alphabetical characters: "; // print an error to the user

break; // break out of the for() loop, as we have already established the input is invalid

}

}

}

else {

cout << "Invalid input. Please input only alphabetical characters: ";

valid = false; // mark input as invalid

}

} while (!valid); // keep going until we get input that is valid

return input; // once we have valid input, return it

}

// class that contains functions for inputing and storing patient data

class patient

{

public:

string First\_name;

string Last\_name;

string treating\_doc;

int room\_number;

int pat\_age;

int pat\_id;

int num\_patients;

void patient\_data();

void entry\_del(string);

void search(int);

//virtual functions used in the two derived classes

virtual void input\_data\_broncho() { };

virtual void input\_data\_malaria() { };

};

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// FUNCTIONS TO GET AND SHOW DATA FOR PATIENT

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void patient::patient\_data()

{

myfile << "\n\tPatient Info " << endl;

myfile << " --------------------------------------------" << endl;

cout << endl << endl;

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

cout << " \n\t\tInput New Patient Info\n\n ";

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

cout << "\n ->Enter Patient First Name : ";

First\_name = getLetters();

myfile << "PATIENT -> Name: " << First\_name << " ";

//checks user input and if it was correct then it

//inputs the data entered by the user in the file Patient.txt

cout << "\n ->Enter Patient Last Name : ";

Last\_name = getLetters();

myfile << Last\_name << " | ";

cout << "\n ->Enter Patient ID : ";

pat\_id = valid\_input\_int();

myfile << "ID: " << pat\_id << " | ";

cout << "\n ->Enter Patient age : ";

pat\_age = valid\_input\_int();

myfile << "Age: " << pat\_age << " | ";

cout << "\n ->Enter the treating doctor's last name : ";

treating\_doc = getLetters();

myfile << "Treating Doctor: " << treating\_doc << " | ";

cout << "\n ->Enter Patient Room Number : ";

room\_number = valid\_input\_int();

myfile << "Room Number: " << room\_number << "";

cout << endl;

cout << endl;

};

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// FUNCTIONS TO DELETE RECORD FOR A SPECIFIC PATIENT OR DOCTOR

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void patient::entry\_del(string delete\_pat)

{

string line;

ifstream myfile;

ofstream temp;

myfile.open("Patient.txt"); // the file that stores the original data

temp.open("temp.txt"); // file where we will store the data left after deletion [basicaly a copy and replace]

//loop that traverses the file Patient.txt and copies all data that is not to be deleted to the file temp.txt

for (unsigned int curLine = 0; getline(myfile, line); curLine++)

{

if (line.find(delete\_pat) != string::npos)

{

}

else

{

temp << line << endl;

}

}

cout << " --Patient entry deleted.";

myfile.close();

temp.close();

remove("Patient.txt"); //removes the original file

rename("temp.txt", "Patient.txt"); //renames the file with the updated info as the old file basically taking its place

};

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// FUNCTIONS TO DISPLAY RECORD OF A SPECIFIC PATIENT OR DOCTOR

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void patient::search(int pattern)

{

ifstream myfile;

myfile.open("Patient.txt");

string line;

string pattern\_str;

pattern\_str = to\_string(pattern);

//loop that traverses the file until it finds the desired info and then outputs the info to the console

for (unsigned int curLine = 0; getline(myfile, line); curLine++)

{

if (line.find(pattern\_str) != string::npos)

{

cout << " --Data on person/room in question: " << line << endl;

}

else

{

}

}

myfile.close();

};

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// FUNCTIONS TO GET AND SHOW DATA FOR WARD

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

class Ward\_info

{

public:

string doc\_first\_name;

string doc\_last\_name;

int doc\_id;

string doc\_specialty;

int rooms\_malaria;

int rooms\_general;

void ward\_data\_doctors();

void ward\_data\_rooms();

};

void Ward\_info::ward\_data\_doctors()

{

int num\_doctors = 0;

myfile << " \n\tAvailable Doctors: " << endl;

myfile << " --------------------------------------------" << endl;

//loop that allows user to input as many doctors as he desires

cout << " -How many doctors to input: ";

num\_doctors = valid\_input\_int();

cout << endl;

while (num\_doctors > 0)

{

cout << "\n ->Enter doctor First Name : ";

doc\_first\_name = getLetters();

myfile << "DOCTOR -> Name: " << doc\_first\_name << " ";

//inputs the data entered by the user in the file Patient.txt

cout << "\n ->Enter doctor Last Name : ";

doc\_last\_name = getLetters();

myfile << doc\_last\_name << " | ";

cout << "\n ->Enter doctor ID : ";

doc\_id = valid\_input\_int();

myfile << "ID: " << doc\_id << " | ";

cout << "\n ->Enter doctor specialty : ";

doc\_specialty = getLetters();

myfile << "Specialty: " << doc\_specialty;

cout << endl;

myfile << endl;

cout << "--------------------------------------------" << endl;

num\_doctors--;

}

myfile << endl << endl;

cout << endl;

};

void Ward\_info::ward\_data\_rooms()

{

int num\_mal\_rooms = 0;

int num\_bro\_rooms = 0;

//loop that allows user to input as many rooms for malaria patients as he desires

cout << " -How many rooms for malaria patients to input: ";

num\_mal\_rooms = valid\_input\_int();

cout << endl;

myfile << " Malaria rooms numbers: " << endl;

while (num\_mal\_rooms > 0)

{

cout << " Enter Malaria room numbers: ";

rooms\_malaria = valid\_input\_int();

myfile << " ->Room number : " << rooms\_malaria << endl;

num\_mal\_rooms--;

}

myfile << endl << endl;

cout << endl;

//loop that allows user to input as many rooms for regular uninfectuous patients as he desires

cout << " -How many rooms for broncho patients to input: ";

num\_bro\_rooms = valid\_input\_int();

cout << endl;

myfile << " Multipurpose rooms numbers: " << endl;

while (num\_bro\_rooms > 0)

{

cout << " Enter multipurpose room number: ";

rooms\_general = valid\_input\_int();

myfile << " ->Room number : " << rooms\_general << endl;

num\_bro\_rooms--;

};

myfile << endl << endl;

};

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// FUNCTIONS TO GET AND SHOW DATA FOR CANCER PATIENTS

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

class Cancer :public patient

{

public:

//virtual functions used in the two derived classes

virtual void input\_data\_treatable() { };

virtual void input\_data\_unreatable() { };

};

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// FUNCTIONS TO GET AND SHOW DATA FOR BRONCHO PATIENTS

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

class Broncho :public patient

{

public:

char treatment\_broncho;

void input\_data\_broncho();

};

void Broncho::input\_data\_broncho()

{

//inputs the treatment for each patient into the file with the help of a switch statement

cout << " ->Enter the type of treatment required for the patient in question: " << endl;

cout << " Rest [R], Medicine [M], Surgery [S], None [N]: ";

while (1) {

cin >> treatment\_broncho; cout << endl;

switch (treatment\_broncho)

{

case 'R':

myfile << " | Condition: Broncho | Treatment: Rest" << endl;

return;

break;

case 'M':

myfile << " | Condition: Broncho | Treatment: Medicine" << endl;

return;

break;

case 'S':

myfile << " | Condition: Broncho | Treatment: Surgery" << endl;

return;

break;

case 'N':

myfile << " | Condition: Broncho | Treatment: None" << endl;

return;

default:

cout << "Wrong input. Try again: ";

continue;

break;

}

}

};

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// FUNCTIONS TO GET AND SHOW DATA FOR MALARIA PATIENT

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

class Malaria :public patient

{

public:

char treatment\_malarie;

void input\_data\_malaria();

};

void Malaria::input\_data\_malaria()

{

//inputs the treatment for each patient into the file with the help of a switch statement

cout << " ->Enter the type of treatment required for the patient in question: " << endl;

cout << " Rest [R], Medicine [M], Surgery [S], None [N]: ";

while (1) {

cin >> treatment\_malarie; cout << endl;

switch (treatment\_malarie)

{

case 'R':

myfile << " | Condition: Malaria | Treatment: Rest" << endl;

return;

break;

case 'M':

myfile << " | Condition: Malaria | Treatment: Medicine" << endl;

return;

break;

case 'S':

myfile << " | Condition: Malaria | Treatment: Surgery" << endl;

return;

break;

case 'N':

myfile << " | Condition: Malaria | Treatment: None" << endl;

return;

break;

default:

cout << "Wrong input. Try again: ";

continue;

break;

}

}

};

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// FUNCTIONS TO GET AND SHOW DATA FOR PATIENT WITH TREATABLE CANCER

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

class Cancer\_Treatable :public Cancer

{

public:

char treatment\_treatable;

void input\_data\_treatable();

};

void Cancer\_Treatable::input\_data\_treatable()

{

//inputs the treatment for each patient into the file with the help of a switch statement

cout << " ->Enter the type of treatment required for the patient in question: " << endl;

cout << " Chemeotherapy [C], Medicine [M], Surgery [S], None [N]: ";

while (1) {

cin >> treatment\_treatable; cout << endl;

switch (treatment\_treatable)

{

case 'C':

myfile << " | Condition: Treatable Cancer | Treatment: Chemeotherapy" << endl;

return;

break;

case 'M':

myfile << " | Condition: Treatable Cancer | Treatment: Medicine" << endl;

return;

break;

case 'S':

myfile << " | Condition: Treatable Cancer | Treatment: Surgery" << endl;

return;

break;

case 'N':

myfile << " | Condition: Treatable Cancer | Treatment: None" << endl;

return;

break;

default:

cout << "Wrong input. Try again: ";

continue;

break;

}

}

};

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// FUNCTIONS TO GET AND SHOW DATA FOR PATIENT WITH UNTREATABLE CANCER

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

class Cancer\_Untreatable :public Cancer

{

public:

char treatment\_unreatable;

void input\_data\_unreatable();

};

void Cancer\_Untreatable::input\_data\_unreatable()

{

//inputs the treatment for each patient into the file with the help of a switch statement

cout << " ->Enter the type of treatment required for the patient in question: " << endl;

cout << " Chemeotherapy [C], Medicine [M], Surgery [S], None [N]: ";

while (1) {

cin >> treatment\_unreatable; cout << endl;

switch (treatment\_unreatable)

{

case 'C':

myfile << " | Condition: Untreatable Cancer | Treatment: Chemeotherapy" << endl;

return;

break;

case 'M':

myfile << " | Condition: Untreatable Cancer | Treatment: Medicine" << endl;

return;

break;

case 'S':

myfile << " | Condition: Untreatable Cancer | Treatment: Surgery" << endl;

return;

break;

case 'N':

myfile << " | Condition: Untreatable Cancer | Treatment: None" << endl;

return;

break;

default:

cout << "Wrong input. Try again: ";

continue;

break;

}

}

};

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// FUNCTION TO SHOW ALL INPUTED DATA

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void display\_all()

{

//opens file, traverses it, and displayes all lines of text that are in the file

ifstream file("Patient.txt");

string content;

while (getline(file, content))

{

cout << content << endl;

};

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// FUNCTIONS TO DELETE ENTIRE DATABASE

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void del\_whole()

{

string confirm;

cout << "WARNING: this option will wipe all data in the database." << endl;

cout << "To continue with deleting press 'Y'. To cancel deletion press any other alphabetical character." << endl;

cout << "Choice: ";

confirm = getLetters();

if (confirm == "Y")

{

myfile.close(); //close the current database file

remove("Patient.txt"); //removes the original file

myfile.open("Patient.txt", std::fstream::in | std::fstream::out | std::fstream::app); //create a new database with the same name

cout << "Deletion complete." << endl;

}

else

{

cout << "Deletion canceled" << endl;

}

}

void main()

{

myfile.open("Patient.txt", std::fstream::in | std::fstream::out | std::fstream::app);

//creating the objects of the classes

Ward\_info w;

patient p;

Cancer c;

Cancer\_Treatable t;

Cancer\_Untreatable u;

Broncho b;

Malaria m;

//pointers to the addresses of the classes that are needed to use the virtual functions

Cancer \*can1 = &t;

Cancer \*can2 = &u;

patient \*pat1 = &c;

patient \*pat2 = &b;

patient \*pat3 = &m;

int choice;

char type\_cancer;

char condition;

int search;

string entry\_del;

char repeat;

bool loop\_control;

int num\_patients;

//first calls this function from class Ward\_info in order for this to be the first data inputed into the file

//inputs the data on the hospital word into a text file

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

cout << " \t\tCreate New Ward Database " << endl;

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

myfile << "\tWard Database" << endl;

myfile << " --------------------------------------------" << endl;

cout << "------------------------------------------------" << endl;

//loop that allows the user to return to the menu multiple times as long as he continues to enter ‘Y’ when prompted

do

{

//menu that uses a switch statement

cout << "\tChoose one of the following options:" << endl;

cout << "(recommended that you start with inputing new doctors and rooms)" << endl << endl;

cout << "-> 1. Create new Doctor entry." << endl;

cout << "-> 2. Enter new rooms." << endl;

cout << "-> 3. Create new patient entry." << endl;

cout << "-> 4. Search for a specific entry by room number, doctor last name, or patient last name." << endl;

cout << "-> 5. Delete a patient entry." << endl;

cout << "-> 6. Delete Entire Database." << endl << endl;

cout << "-> 7. Show entire database." << endl;

cout << "-> 8. Exit program." << endl << endl;

cout << " Your choice: ";

choice = valid\_input\_int();

switch (choice)

{

case 1:

w.ward\_data\_doctors();

break;

case 2:

w.ward\_data\_rooms();

break;

case 3:

{

cout << " -How many patients to input: ";

num\_patients = valid\_input\_int();

cout << endl;

//loop that repeats based on the number of times a user wants to input records

while (num\_patients > 0) {

p.patient\_data(); //calls function that accepts input for patient: his name, ID, etc.

cout << " What is the patient suffering from: " << endl << " Cancer [C], Bronchopneumonia [B], Malaria [M]? ";

loop\_control = true;

while (loop\_control == true) {

cin >> condition; cout << endl;

if (condition == 'C' || condition == 'B' || condition == 'M') {

//nested switch statement that allows the user to choose which type of cancer the patient is suffering from

switch (condition)

{

case 'C':

cout << " --Is the cancer treatable [T] or untreatable [U]? ";

cin >> type\_cancer;

cout << endl;

if (type\_cancer = 'T')

can1->input\_data\_treatable();

else

can1->input\_data\_unreatable();

break;

case 'B':

pat2->input\_data\_broncho();

break;

case 'M':

pat3->input\_data\_malaria();

break;

}

loop\_control = false;

}

else {

cout << "Wrong input. Try again: ";

}

}

num\_patients--;

}

}

break;

case 4:

cout << " ->Enter ID of person or number of room: ";

search = valid\_input\_int(); cout << endl;

p.search(search);

break;

case 5:

//first we need to close the file otherwise we would not be able to delete it

//because of permission issues. After we call the delete function we open the file again,

//but now it has one entry less due to us deleting the wanted patient data

cout << " ->Enter ID of the patient or doctor whose records to delete: ";

cin >> entry\_del;

myfile.close();

p.entry\_del(entry\_del);

myfile.open("Patient.txt", std::fstream::in | std::fstream::out | std::fstream::app);

break;

case 6:

del\_whole();

break;

case 7:

display\_all();

break;

case 8:

cout << "\n \*\* Exiting program. \*\*" << endl << endl;

system("PAUSE");

return;

break;

default:

cout << "Invalid entry. Aborting program.";

system("PAUSE");

continue;

break;

};

cout << "\n\n\tInput any alphanumeric character to go back to the main menu or type 'E' to exit? ";

cin >> repeat;

cout << " ----------------------------------------------------" << endl << endl;

} while (repeat != 'E');

myfile.close();

}