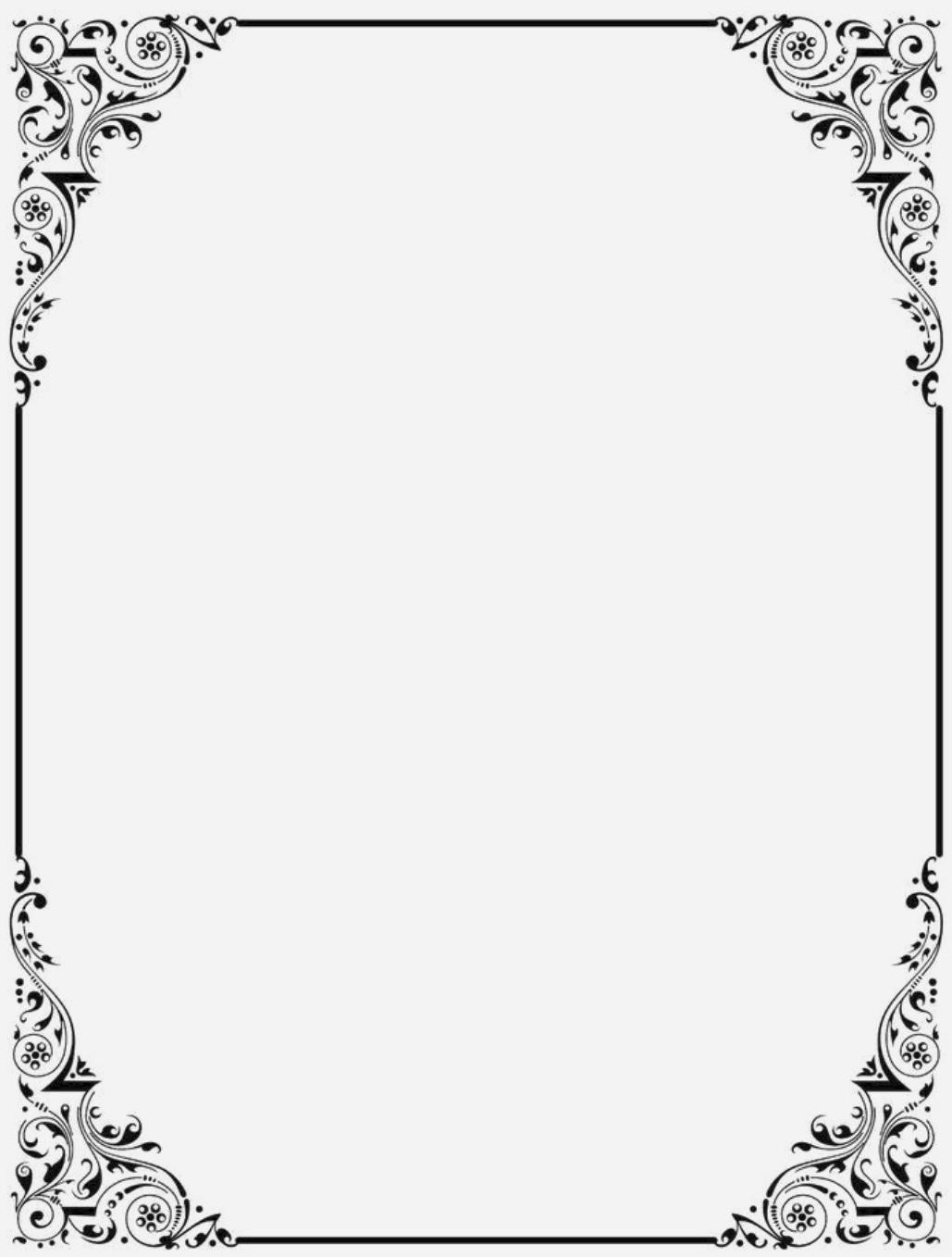
ĐẠI HỌC BÁCH KHOA HÀ NỘI



1 | P a g e

**Trường Điện – Điện tử**

--------

BÁO CÁO: Kỹ thuật lập trình C/C++

# Đề tài : “Hệ thống quản lý bệnh nhân - bệnh án tại bệnh viện.”

*Giảng viên hướng dẫn*: ***Nguyễn Thị Kim Thoa***

*Sinh viên thực hiện*: Bùi Trọng Nam Anh 20233990

Nguyễn Hoàng Anh 20233994

Nguyễn Lê Việt Anh 20233995

Vũ Việt Anh 20233996

Link github của nhóm chúng em: <https://github.com/NamAnh2k5/BTL-Ky-thuat-lap-trinh>

Các chức năng của chương trình:

* Thêm thông tin bệnh nhân
* Sửa thông tin bệnh nhân
* Xoá thông tin bệnh nhân
* Hiển thị thông tin của bệnh nhân
* Tìm kiếm thông tin của bệnh nhân (theo mã bênh nhân, tên bệnh nhân)
* Sắp xếp bệnh nhân (theo tên và ID)
* Thông kê thông tin của bệnh nhân (theo độ tuổi, giới tính)
* Lưu lại dữ liệu cho lần chạy kế tiếp

Chi tiết chức năng ứng với các hàm (ngôn ngữ C++):

Hàm tạo mã bệnh nhân theo định danh BNxx

string Patient::generatePatientID(const vector<Patient>& patients) {

int maxID = 0;

for (const auto& patient : patients) {

string id = patient.getPatientID();

if (id.size() > 2 && id.substr(0, 2) == "BN") {

int num = stoi(id.substr(2));

if (num > maxID) {

maxID = num;

}

}

}

//return "BN" + to\_string(maxID + 1);

ostringstream oss;

oss << "BN" << setw(3) << setfill('0') << (maxID + 1); // Đệm số 0 lên 3 chữ số

return oss.str();

}

Hàm addPatient: có chức năng thêm bệnh nhân vào chương trình bao gồm các thông tin <(string)mã bệnh nhân>, <(string)tên>, <(Date)ngày/tháng/năm sinh>, <(string)giới tính>, <(string)địa chỉ>, <(string)số điện thoại>

void Patient::addPatient(vector<Patient>& patients) {

Patient newPatient;

string input;

int day, month, year;

string newID = generatePatientID(patients);

newPatient.setPatientID(newID);

cout << "Tao ma so benh nhan: " << newID << endl;

cout << "Nhap ten benh nhan: ";

getline(cin, input);

newPatient.setPatientName(input);

cout << "Nhap ngay sinh:" << endl;

cout << " Ngay (dd): ";

cin >> day;

cout << " Thang (mm): ";

cin >> month;

cout << " Nam (yyyy): ";

cin >> year;

cin.ignore();

newPatient.setBirthday(Date(day, month, year));

cout << "Nhap gioi tinh: ";

getline(cin, input);

newPatient.setGender(input);

cout << "Nhap dia chi: ";

getline(cin, input);

newPatient.setAddress(input);

cout << "Nhap so dien thoai: ";

getline(cin, input);

newPatient.setPhoneNumber(input);

patients.push\_back(newPatient);

}

Hàm searchPatients: có chức năng tìm kiếm thông tin bệnh nhân theo <mã bệnh nhân>, <tên bệnh nhân>

void Patient::searchPatients(const vector<Patient>& patients, char searchOption) {

    if (searchOption != '1' && searchOption != '2') {

        cout << "Lua chon khong hop le." << endl;

        return;

    }

    if (searchOption == '1') {

        string id;

        cout << "Nhap ma so benh nhan can tim kiem: ";

        cin >> id;

        bool found = false;

        for (const auto& patient : patients) {

            if (patient.getPatientID() == id) {

                found = true;

                cout << "Thong tin benh nhan can tim: " << endl;

                cout << "Ho va ten: " << patient.getPatientName() << endl;

                cout << "Ngay sinh: " << patient.getBirthday().getDay() << "/" << patient.getBirthday().getMonth() << "/" << patient.getBirthday().getYear() << endl;

                cout << "Gioi tinh: " << patient.getGender() << endl;

                cout << "Dia chi: " << patient.getAddress() << endl;

                cout << "So dien thoai: " << patient.getPhoneNumber() << endl;

                cout << "Ma benh nhan: " << patient.getPatientID() << endl;

                break;

            }

        }

        if (!found)

            cout << "Khong tim thay benh nhan voi ma so: " << id << endl;

    } else if (searchOption == '2') {

        string name;

        cout << "Nhap ten benh nhan can tim kiem: ";

        cin.ignore();

        getline(cin, name);

        bool found = false;

        for (const auto& patient : patients) {

            if (patient.getPatientName() == name) {

                found = true;

                cout << "Thong tin benh nhan can tim: " << endl;

                cout << "Ho va ten: " << patient.getPatientName() << endl;

                cout << "Ngay sinh: " << patient.getBirthday().getDay() << "/" << patient.getBirthday().getMonth() << "/" << patient.getBirthday().getYear() << endl;

                cout << "Gioi tinh: " << patient.getGender() << endl;

                cout << "Dia chi: " << patient.getAddress() << endl;

                cout << "So dien thoai: " << patient.getPhoneNumber() << endl;

                cout << "Ma benh nhan: " << patient.getPatientID() << endl;

            }

        }

        if (!found)

            cout << "Khong tim thay benh nhan voi ten: " << name << endl;

    }

}

Hàm displayPatients: có chức năng hiển thị thông tin của tất cả bệnh nhân

oid Patient::displayPatients(const vector<Patient>& patients) {

cout << left << setw(20) << "Ho va ten"

<< setw(15) << "Ngay sinh"

<< setw(10) << "Gioi tinh"

<< right<< setw(25) << "Dia chi"

<< setw(17) << "So dien thoai"

<< setw(18) << "Ma benh nhan" << endl;

cout << setfill('-') << setw(105) << "-" << endl;

cout << setfill(' ');

for (const auto& patient : patients) {

cout << left << setw(20) << patient.getPatientName()

<< setw(2) << patient.getBirthday().getDay() << "/"

<< setw(2) << patient.getBirthday().getMonth() << "/"

<< setw(2) << patient.getBirthday().getYear() << setfill(' ') << setw(11)

<< right << setw(10) << patient.getGender()

<< right << setw(30) << patient.getAddress()

<< right << setw(15) << patient.getPhoneNumber()

<< right<< setw(15) << patient.getPatientID() << endl;

cout << setfill('-') << setw(105) << "-" << endl;

cout << setfill(' ');

}

}

Hàm deletePatient: có chức năng xoá dữ liệu của bệnh nhân

void Patient::deletePatient(vector<Patient>& patients){

    cout<<"\nNhap ID ma ban muon xoa: ";

    string id;

    cin.ignore();

    getline(cin,id);

    bool found=false;

    for(int i=0;i<patients.size();i++){

        if(patients[i].getPatientID() == id){

            patients.erase(patients.begin()+i);

            found=true;

            cout<<"Da xoa thanh cong benh nhan voi ID: "<<id<<endl;

            break;

        }

    }

    if(!found){

        cout<<"Khong tim thay benh nhan co ID: "<<id<<endl;

    }

}

Hàm editPatint: có chức năng sửa thông tin của bệnh nhân

void Patient::editPatient(vector<Patient>& patients){

    cout<<"Nhap ID cua benh nhan ma ban muon sua: ";

    string id;

    cin.ignore();

    getline(cin,id);

    bool found=false;

    for(int i=0;i<patients.size();i++){

        if(patients[i].getPatientID() == id){

            found = true;

            cout << "Chon muc can sua:" << endl;

            cout << "[1]. Ten benh nhan" << endl;

            cout << "[2]. Ngay sinh" << endl;

            cout << "[3]. Gioi tinh" << endl;

            cout << "[4]. Dia chi" << endl;

            cout << "[5]. So dien thoai" << endl;

            cout << "[0]. Quay lai menu" << endl;

            char choice;

            do{

            cin >> choice;

            cin.ignore();

            switch (choice) {

                case '1': {

                    string newName;

                    cout << "Nhap ten benh nhan moi: ";

                    getline(cin, newName);

                    patients[i].setPatientName(newName);

                    cout << "Da sua ten benh nhan thanh cong." << endl;

                    break;

                }

                case '2': {

                    int day, month, year;

                    cout << "Nhap ngay sinh moi:" << endl;

                    cout << "  Ngay (dd): ";

                    cin >> day;

                    cout << "  Thang (mm): ";

                    cin >> month;

                    cout << "  Nam (yyyy): ";

                    cin >> year;

                    cin.ignore();

                    patients[i].setBirthday(Date(day, month, year));

                    cout << "Da sua ngay sinh thanh cong." << endl;

                    break;

                }

                case '3': {

                    string newGender;

                    cout << "Nhap gioi tinh moi: ";

                    getline(cin, newGender);

                    patients[i].setGender(newGender);

                    cout << "Da sua gioi tinh thanh cong." << endl;

                    break;

                }

                case '4': {

                    string newAddress;

                    cout << "Nhap dia chi moi: ";

                    getline(cin, newAddress);

                    patients[i].setAddress(newAddress);

                    cout << "Da sua dia chi thanh cong." << endl;

                    break;

                }

                case '5': {

                    string newPhone;

                    cout << "Nhap so dien thoai moi: ";

                    getline(cin, newPhone);

                    patients[i].setPhoneNumber(newPhone);

                    cout << "Da sua so dien thoai thanh cong." << endl;

                    break;

                }

                case '0':

                    cout << "Quay lai menu chinh." << endl;

                    break;

                default:

                    cout << "Lua chon khong hop le." << endl;

                    break;

            }

            }while(choice!='0');

        }

    }

    if(!found){

        cout<<"Khong tim thay benh nhan co ID: "<<id<<endl;

    }

}

Hàm sortPatientsByName: có chức năng sắp xếp bệnh nhân theo tên

void Patient::sortPatientsByName(vector<Patient>& patients) {

sort(patients.begin(), patients.end(), [](const Patient& a, const Patient& b) {

// Chuyển tất cả ký tự trong tên thành chữ thường để so sánh

string nameA = a.getPatientName();

string nameB = b.getPatientName();

string temp = ""; //Biến tạm để lưu tên bệnh nhân

int i;

for(i = nameA.size()-1; i > 0; i--) { //Chạy ngược từ ký tự cuối lại để lấy tên của bệnh nhân

if(nameA[i] == ' ') break; //Gặp khoảng trống là dừng

}

// cout << i << endl;

for(int a = i+1; a < nameA.size(); a++) { //Gán từ sau vị trí khoảng trống nãy vừa tìm được đến hết string để lấy tên

temp+= nameA[a];

}

nameA = temp; //Gán temp vào nameA để chạy hàm ở dưới

//Tương tự như trên nhưng với nameB

temp = "";

//int i;

for(i = nameB.size()-1; i > 0; i--) {

if(nameB[i] == ' ') break;

}

// cout << i << endl;

for(int a = i+1; a < nameB.size(); a++) {

temp+= nameB[a];

}

nameB = temp;

transform(nameA.begin(), nameA.end(), nameA.begin(), ::tolower);

transform(nameB.begin(), nameB.end(), nameB.begin(), ::tolower);

return nameA < nameB;

});

displayPatients(patients);

}

Hàm sortPatientsByID: có chức năng sắp xếp bệnh nhân theo ID

void Patient::sortPatientsByID(vector<Patient>& patients) {

sort(patients.begin(), patients.end(), [](const Patient& a, const Patient& b) {

return a.getPatientID() < b.getPatientID();

});

displayPatients(patients);

}

Hàm calculateAge: có chức năng tính tuổi của bệnh nhân

int calculateAge(const Date& birthdate) {

    time\_t t = time(0);

    tm\* now = localtime(&t);

    int currentYear = now->tm\_year + 1900;

    int currentMonth = now->tm\_mon + 1;

    int currentDay = now->tm\_mday;

    int age = currentYear - birthdate.getYear();

    if (currentMonth < birthdate.getMonth() || (currentMonth == birthdate.getMonth() && currentDay < birthdate.getDay())) {

        age--;

    }

    return age;

}

Hàm statisticsPatients: có chức năng thống kê thông tin của bệnh nhân theo <độ tuổi>, <giới tính>

void Patient::statisticsPatients(const vector<Patient>& patients) {

    int choice;

    cout << "[1]. Thong ke benh nhan theo do tuoi." << endl;

    cout << "[2]. Thong ke benh nhan theo gioi tinh." << endl;

    cout << "Nhap lua chon: ";

    cin >> choice;

    switch (choice) {

        case 1: {

            int under15 = 0, between15And65 = 0, over65 = 0;

            for (const auto& patient : patients) {

                int age = calculateAge(patient.getBirthday());

                if (age < 15) under15++;

                else if (age <= 65) between15And65++;

                else over65++;

            }

            cout << "So luong benh nhan duoi 15 tuoi: " << under15 << endl;

            cout << "So luong benh nhan tu 15 den 65 tuoi: " << between15And65 << endl;

            cout << "So luong benh nhan tren 65 tuoi: " << over65 << endl;

            break;

        }

        case 2: {

            int male = 0, female = 0;

            for (const auto& patient : patients) {

                if (patient.getGender() == "Nam") male++;

                else if (patient.getGender() == "Nu") female++;

            }

            cout << "So luong benh nhan nam: " << male << endl;

            cout << "So luong benh nhan nu: " << female << endl;

            break;

        }

        default:

            cout << "Lua chon khong hop le!" << endl;

            break;

    }

}

Hàm saveFile: có chức năng lưu lại các dữ liệu của chương trình cho lần chạy tiếp theo.

void saveFile(const vector<Patient>& patients) {

    ofstream output;

    output.open("data/dulieu-benhnhan.txt");

    //Dòng đầu tiên sẽ lưu lại số lượng các bệnh án có trong file, sau các dòng sau sẽ là các bệnh án, thuộc tính đầu tiên của mỗi bênh án là mã bệnh án.

    output << patients.size() << "\n";

    //<Mã bệnh án>, <tên bệnh nhân>, <ngày/tháng/năm sinh>, <giới tính>, <địa chỉ>, <số điện thoại>

    for (const auto& patient : patients) {

        output << patient.getPatientID() << ","

             << patient.getPatientName() << ","

             << patient.getBirthday().getDay() << "/"

             << patient.getBirthday().getMonth() << "/"

             << patient.getBirthday().getYear() << ","

             << patient.getGender() << ","

             << patient.getAddress() << ","

             << patient.getPhoneNumber() << endl;

    }

    output.close();

}

Hàm readFile: có chức năng đọc dữ liệu được lưu từ lần chạy trước

void readFile(vector<Patient>& patients) {

    Patient newPatient;

    string inpatient;

    ifstream input;

    input.open("data/dulieu-benhnhan.txt");

    string data; //data là biến lấy dữ liệu của 1 dòng lưu trong file

    int commaCount=0; //Biến này để đếm số dấu ',' từ đó xác dịnh được thuộc tính đang đọc từ file là thuộc tính nào

    const int numberInfor = 6;

    string parsing[numberInfor]={"","","","","",""}; //Mảng này để lưu các thuộc tính đọc ra từ file (kích thước của bảng = số lượng các dữ liệu)

    int time[3] = {1,1,1}; //Mảng lưu thời gian được tách ra từ file

    int i=0; //i là biến chạy tăng dần để đọc từng kí tự data

    getline(input, data);

    cout << "So luong benh nhan: " << stoi(data) << endl;;

    data = "";

    while(!input.eof()) {

        getline(input,data);

        cout << data << endl;

        cout << endl;

        i=0;

        commaCount = 0;

        while(data[i] != '\0') {

            if(data[i] != ',' && data[i] != '\0') { //Kiểm tra nếu không phải dấu ',' thì chèn ký tự vào cuối chuỗi

            parsing[commaCount]+=data[i];

            }

            i++;

            if(data[i] == ',' || data[i] == '\0') { //Nếu kí tự hiện tại là dấu ',' hoặc rỗng '\0' thì hiển thị chuỗi sau đó reset chuỗi

                if(commaCount == 2) { //thay đổi trị số này ứng với vị trí của ngày tháng năm được lưu trong file, CHƯƠNG TRÌNH BỊ TREO NẾU TRỊ SỐ SAI

                    //cout << stoi(parsing[commaCount]) << endl; //chuyển đổi từ string về int để xử lý năm sinh (lưu ý: phải xác định chính xác vị trí chứa năm sinh, nếu xác định sai thì sẽ bị treo)

                    int y=0;

                    string temp;

                    int slashCount = 0; //Biến đếm số gạch chéo để phân tách ngày tháng năm

                    while(parsing[commaCount][y] != '\0') {

                        temp += parsing[commaCount][y];

                        if(parsing[commaCount][y] == '/') {

                            time[slashCount] = stoi(temp);

                            //cout << time[slashCount] << endl;

                            temp = "";

                            slashCount++;

                        }

                        y++;

                    }

                    time[slashCount] = stoi(temp);

                    //cout << time[2] << endl;

                }

                //cout << parsing[commaCount] << endl;

                //parsing[commaCount] = "";

                commaCount++;

            }

        }

        if(parsing[0] != "") {

            newPatient.setPatientID(parsing[0]);

            cout << newPatient.getPatientID() << endl;

            newPatient.setPatientName(parsing[1]);

            cout << newPatient.getPatientName() << endl;

            newPatient.setBirthday(Date(time[0], time[1], time[2]));

            newPatient.setGender(parsing[3]);

            cout << newPatient.getGender() << endl;

            newPatient.setAddress(parsing[4]);

            cout << newPatient.getAddress() << endl;

            newPatient.setPhoneNumber(parsing[5]);

            cout << newPatient.getPhoneNumber() << endl;

            patients.push\_back(newPatient);

        }

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

            parsing[i] = "";

        }

    }

    input.close();

}

Hàm menu: có chức năng cho người dùng các tuỳ chọn

void menu() {

vector<Patient> patients;

readFile(patients);

char number;

do {

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

cout << "=============QUAN LI BENH NHAN=============" << endl;

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

cout << "[1]. Them thong tin benh nhan." << endl;

cout << "[2]. Hien thi danh sach benh nhan." << endl;

cout << "[3]. Sua thong tin benh nhan." << endl;

cout << "[4]. Xoa thong tin benh nhan." << endl;

cout << "[5]. Tim kiem ho so benh nhan." << endl;

cout << "[6]. Sap xep ho so benh nhan." << endl;

cout << "[7]. Thong ke ho so benh nhan." << endl;

cout << "[0]. Thoat chuong trinh" << endl;

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

cout << "Nhap lua chon cua ban: ";

cin >> number;

cin.ignore();

switch (number) {

case '1':

system("cls");

Patient::addPatient(patients);

break;

case '2':

system("cls");

Patient::displayPatients(patients);

break;

case '3':

system("cls");

Patient::editPatient(patients);

break;

case '4':

system("cls");

Patient::deletePatient(patients);

break;

case '5':

system("cls");

char searchOption;

cout << "Tim kiem theo (1) ID hoac (2) Ten? ";

cin >> searchOption;

Patient::searchPatients(patients, searchOption);

break;

case '6':

cout << "Chon kieu sap xep (1: Ten, 2: ID): ";

int sortOption;

cin >> sortOption;

if (sortOption == 1) {

Patient::sortPatientsByName(patients);

} else if (sortOption == 2) {

Patient::sortPatientsByID(patients);

} else {

cout << "Lua chon khong hop le." << endl;

}

break;

case '7':

system("cls");

Patient::statisticsPatients(patients);

break;

case '0':

cout << "Thoat chuong trinh." << endl;

saveFile(patients);

cout << "Cam on ban da su dung chuong trinh!";

break;

default:

system("cls");

cout << "Lua chon khong hop le. Vui long chon lai!" << endl;

break;

}

} while (number != '0');

}