Пензенский государственный университет

Кафедра «Вычислительная техника»

**Отчёт**

По лабораторной работе №2

по дисциплине: «Объектно - ориентированное программирование»

на тему «Изучение механизмов перегрузки операций и дружественных функций в языке С++»

Выполнили студенты группы 19ВВ2:

Муромский Д.А.

Кобзев М.И.

Горбунов Д.А.

Приняли:

Евсеева Ю.И.

Гудков А.А.

Пенза 2021

**Цель работы:** Изучить механизмы перегрузки операций и дружественных функций в языке С++.

**Задания**

Модифицировать программу из лабораторной работы № 1 следующим образом:

1. В базовом разрабатываемом классе перегрузить операции равенства (==), неравенства (!=) и присваивания (=);
2. Разработать новый класс Observer, который реализует дружественный метод GetInfo по отношению к базовому классу. Метод GetInfo должен выводить в формате целостной строки сведения из private-полей базового класса.

**Практическая часть**

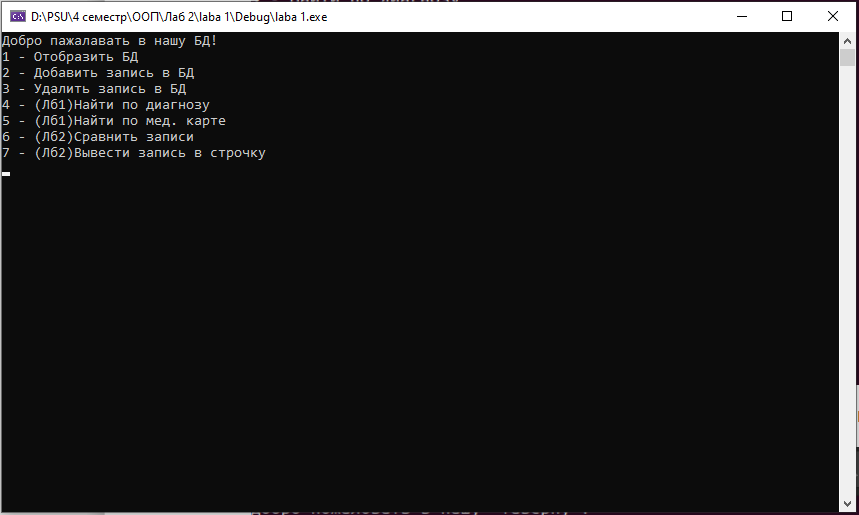


Рисунок 1 - Добавлено 2 новых пункта в меню (для реализации задания Лабораторной работы 2).

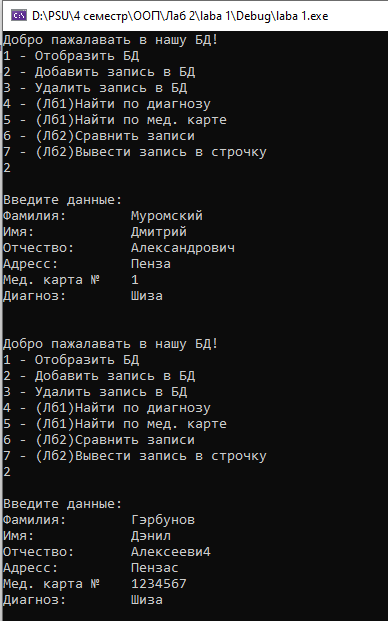


Рисунок 2 - Добавляем новые записи в БД

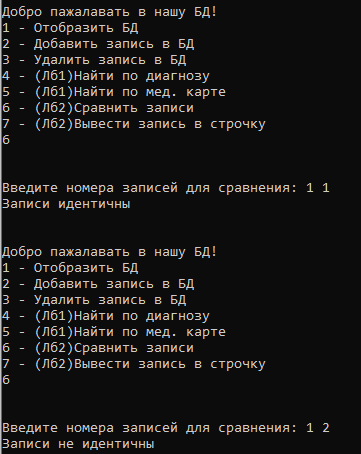


Рисунок 3 - Результат работы функции сравнения записей

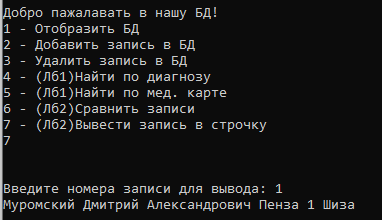


Рисунок 4 - Результат работы функции вывода в строку

**Листинг class.h**

#include <vector>

#include <string>

using namespace std;

class patient {

public:

friend class Observer;

string get\_last\_name() { return last\_name; }

string get\_name() { return name; }

string get\_papich() { return papich; }

string get\_address() { return address; }

long long int get\_number() { return number; }

string get\_shiza() { return shiza; }

void set\_last\_name(string a) { last\_name = a;}

void set\_name(string a) { name = a; }

void set\_papich(string a) { papich = a; }

void set\_address(string a) { address = a; }

void set\_number(long long int a) { number = a; }

void set\_shiza(string a) { shiza = a; };

friend bool operator== (const patient& object1, const patient& object2);

friend bool operator!= (const patient& object1, const patient& object2);

patient& operator= (const patient& object);

patient()

{

}

patient(string a, string b, string c, string d, long long int e, string f)

: last\_name(a), name(b), papich(c), address(d), number(e), shiza(f)

{

}

patient(const patient& object)

: last\_name(object.last\_name),

name(object.name),

papich(object.papich),

address(object.address),

number(object.number),

shiza(object.shiza)

{

}

~patient() {

}

private:

string last\_name;

string name;

string papich;

string address;

long long int number = 0;

string shiza;

};

class Observer {

public:

string GetInfo(patient& object);

};

**Листинг class.cpp**

#include "class.h"

bool operator== (const patient& object1, const patient& object2)

{

return (object1.last\_name == object2.last\_name &&

object1.name == object2.name &&

object1.papich == object2.papich &&

object1.address == object2.address &&

object1.number == object2.number &&

object1.shiza == object2.shiza

);

}

bool operator!= (const patient& object1, const patient& object2)

{

return !(object1 == object2);

}

patient& patient::operator= (const patient& object){

last\_name = object.last\_name;

name = object.name;

papich = object.papich;

address = object.address;

number = object.number;

shiza = object.shiza;

return \*this;

}

string Observer::GetInfo(patient& object) {

return (object.last\_name + " " +

object.name + " " +

object.papich + " " +

object.address + " " +

to\_string(object.number) + " " +

object.shiza);

}

**Листинг func.cpp**

#include "Func.h"

#include <iostream>

#include <windows.h>

vector<patient> list;

void record(int id, patient& object) {…}

void vivod() {…}

void add\_record() {…}

void delete\_record() {…}

void seach\_shiza\_record() {…}

void seach\_number\_record() {…}

void sravnenie() {

cout << endl;

if (list.empty()) {

cout << "Сравнивать нечего, БД пуста" << endl;

}

else {

cout << "Введите номера записей для сравнения: ";

int x, y;

cin >> x >> y;

if (x <= list.size() && y <= list.size()) {

patient temp\_x = list[x - 1]; // присваивание

patient temp\_y = list[y - 1];

if (temp\_x == temp\_y) { // сравнение

cout << "Записи идентичны" << endl;

}

if (temp\_x != temp\_y) { //сравнение с отрицанием

cout << "Записи не идентичны" << endl;

}

}

}

cout << endl;

cout << endl;

mainmenu();

}

void vivod\_str() {

cout << endl;

Observer temp;

if (list.empty()) {

cout << "Выводить нечего, БД пуста" << endl;

}

else {

cout << "Введите номера записи для вывода: ";

int x;

cin >> x;

cout << temp.GetInfo(list[x-1]);

}

cout << endl;

cout << endl;

mainmenu();

}

void mainmenu() {

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

cout << "Добро пажалавать в нашу БД!" << endl;

cout << "1 - Отобразить БД" << endl;

cout << "2 - Добавить запись в БД" << endl;

cout << "3 - Удалить запись в БД" << endl;

cout << "4 - (Лб1)Найти по диагнозу" << endl;

cout << "5 - (Лб1)Найти по мед. карте" << endl;

cout << "6 - (Лб2)Сравнить записи" << endl;

cout << "7 - (Лб2)Вывести запись в строчку" << endl;

int number;

cin >> number;

cout << endl;

switch (number) {

case 1:

vivod();

break;

case 2:

add\_record();

break;

case 3:

delete\_record();

break;

case 4:

seach\_shiza\_record();

break;

case 5:

seach\_number\_record();

break;

case 6:

sravnenie();

break;

case 7:

vivod\_str();

break;

}

}

**Вывод:** Изучили механизмы перегрузки операций и дружественных функций в языке С++.