|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | |  |  |  | | МИНОБРНАУКИ РОССИИ | | | | Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |   Институт Информационных технологий | |
|  | |
| Кафедра Математического обеспечения и стандартизации информационных технологий | |
|  | |
|  | |

|  |  |
| --- | --- |
| **ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 5** | |
| **по дисциплине** | |
| **«**Объектно-ориентированное программирование**»**  **Тема: «Наследование. Полиморфизм»** | |
|  | |
| Выполнил студент группы ИКБО-12-18 | Валяев Д.А. |
| Принял преподаватель | Миронов А.Н. |

|  |  |  |
| --- | --- | --- |
| Лабораторная работа выполнена | «4»мая2019 г. | *(подпись студента)* |
|  |  |  |
| «Зачтено» | «\_\_»\_\_\_\_\_\_\_2019 г. | *(подпись руководителя)* |

Москва 2019

Задание 1. Создание базового класса.

Декомпозиция

Список подзадач задачи:

1. Создать класс Студент
2. Узнать зачислены ли 2 студента в 1 институте
3. Вернуть группу, институт, кафедру по id

**Разработка первой задачи**

1. Постановка задачи

Дано: -

Результат: -

Ограничения: нет

1. Входные и выходные данные

Входные данные –

Выходные данные –

1. Таблица имён

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя** | **Структура** | **Диапазон значений** | **Семантика** |
| dt | сложная | символы | Параметр значение |
| LastName | сложная | символы | Параметр значение |
| id | простая | Целые числа | Параметр значение |
| Name | сложная | символы | Параметр значение |
| Number | простая | Целые числа | Параметр значение |
| gr | простая | символы | Параметр значение |
| d2 | простая | символы | Параметр значение |

1. Кодирование

**Student.h**

#pragma once

#include <iostream>

#include <string.h>

class Student

{

private:

unsigned int id;

unsigned int Number;

char LastName[30];

char Name[30];

char dt[30];

char gr[50];

char d2[40];

public:

// Конструктор без параметров

Student();

// Конструктор с параметрами

Student(unsigned int ids, unsigned int Num, char l[], char Nam[], char d[], char gro[], char d3[]);

// Дестрктор

~Student();

// Методы модификации

void Set\_LastName(const char l[]) { strcpy\_s(LastName, l); }

void Set\_Name(const char Nam[]) { strcpy\_s(Name, Nam); }

void Set\_dt(const char d[]) { strcpy\_s(dt, d); }

void Set\_gr(const char gro[]) { strcpy\_s(gr, gro); }

void Set\_d2(const char d3[]) { strcpy\_s(d2, d3); }

void Set\_id(unsigned int ids) { id = ids; }

void Set\_Number(unsigned int Num) { Number = Num; }

char \*Get\_LastName() { return LastName; }

char \*Get\_Name() { return Name; }

char \*Get\_dt() { return dt; }

char \*Get\_gr() { return gr; }

char \*Get\_d2() { return d2; }

unsigned int Get\_id() { return id; }

unsigned int Get\_Number() { return Number; }

};

**Student.cpp**

#include "pch.h"

#include "Student.h"

Student::Student()

{

id = 0;

Number = 0;

strcpy\_s(LastName, "");

strcpy\_s(Name, "");

strcpy\_s(dt, "");

strcpy\_s(gr, "");

strcpy\_s(d2, "");

}

Student::Student(unsigned int ids, unsigned int Num, char l[], char Nam[], char d[], char gro[], char d3[])

{

id = ids;

Number = Num;

strcpy\_s(LastName, l);

strcpy\_s(Name, Nam);

strcpy\_s(dt, d);

strcpy\_s(gr, gro);

strcpy\_s(d2, d3);

}

Student::~Student()

{

}

#pragma once

#include "newstudent.h"

#include<iostream>

using namespace std;

// фамилия и дата рождения по id

void st1(unsigned int num, Student \*arr, int length);

// зачислены ли 2 студента в 1 дату

bool st(unsigned int num1, unsigned int num2, Student \*arr, int length);

#include "pch.h"

#include "newstudent.h"

bool st(unsigned int num1, unsigned int num2, Student \*arr, int length)

{

bool firstIn = false;

bool secondIn = false;

unsigned int CD1 = num1;

unsigned int CD2 = num2;

unsigned int P1;

unsigned int P2;

for (int i = 0; i < length; i++)

{

if (!firstIn && arr[i].Get\_id() == CD1)

{

P1 = arr[i].Get\_inst();

firstIn = true;

}

if (!secondIn && arr[i].Get\_id() == CD2)

{

P2 = arr[i].Get\_ inst ();

secondIn = true;

}

if (!firstIn && i == length - 1 && arr[i].Get\_id() != CD1)

cout << "Первый студент не найден " << endl;

if (!secondIn && i == length - 1 && arr[i].Get\_id() != CD2)

cout << "Второй студент не найден " << endl;

}

if (P1 == P2)

return true;

return false;

}

#include "pch.h"

#include "newstudent.h"

void Setkaf(unsigned int id, newstudent \*arr, int length) {

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

if (arr[i].Get\_ids() == id) {

char Kaf[30];

cout << "Введите кафедру" << endl; cin >> Kaf;

arr[i].Set\_kaf(Kaf);

}

}

}

void Setinst(unsigned int id, newstudent \*arr, int length) {

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

if (arr[i].Get\_ids() == id) {

char insT[30];

cout << "Введите кафедру" << endl; cin >> insT;

arr[i].Set\_inst(insT);

}

}

}

void st1(unsigned int num, newstudent \*arr, int length)

{

for (int i = 0; i < length; i++)

if (arr[i].Get\_id() == num)

{

cout << arr[i].Get\_gr() << " " << arr[i].Get\_inst() <<" " << arr[i].Get\_kaf()<< endl;

break;

}

}

Класс наследник

#include "Student.h"

class newstudent : public Student

{

private:

char inst[30];

char kaf[30];

public:

newstudent();

~newstudent();

newstudent(unsigned int ids, unsigned int Num, char l[], char Nam[], char d[], char gro[], char d3[], char inst[], char kaf[]);

char Set\_inst(const char in[]) { strcpy\_s(inst, in); }

char Set\_kaf(const char ka[]) { strcpy\_s(kaf, ka); }

char \*Get\_inst() { return inst; }

char \*Get\_kaf() { return kaf; }

};

Реализация методов класса

#include "pch.h"

#include "newstudent.h"

newstudent::newstudent() : Student()

{

}

newstudent::newstudent(unsigned int ids, unsigned int Num, char l[], char Nam[], char d[], char gro[], char d3[], char inst[], char kaf[]):Student(t ids, Num, l, Nam, d, gro, d3,inst, kaf)

{

strcpy\_s(this->inst, inst);

strcpy\_s(this->kaf, kaf);

}

newstudent::~newstudent()

{

}

**Основная программа для тестирования объектов класса**

#include "pch.h"

#include <iostream>

#include <locale.h>

#include "Menu.h"

#include <Windows.h>

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

Menu m;

m.OutPut();

}

**Класс меню**

#pragma once

#include <iostream>

#include "Student.h"

#include "Op.h"

#include "newstudent.h"

using namespace std;

class Menu

{

protected:

int input;

int length = 0;

bool Correct = false;

bool ex = false;

newstudent \*base = new newstudent[length];

public:

// конструктор без параметров

Menu();

// Деструктор

~Menu();

// Вывод пунктов меню

void OutPut();

// Добавление id в базу

void Add(Student \*&arr, int &length, Student \*C);

// Распечатать всех

void prnt(Student \*arr, int length);

// Распечатать пункты меню

void Printr();

};

**Реализация методов вне класса**

#include "pch.h"

#include "Menu.h"

Menu::Menu()

{

}

Menu::~Menu()

{

}

void Menu::Add(Student \*&arr, int &length, Student \*C)

{

Student \*newArr = new Student[length + 1];

for (int i = 0; i < length; i++)

newArr[i] = arr[i];

newArr[length] = \*C;

delete[] arr;

arr = newArr;

length++;

}

void Menu::prnt(Student \* arr, int length)

{

for (int i = 0; i < length; i++)

cout << arr[i].Get\_LastName() << endl;

}

void Menu::Printr()

{

cout << "Выберите действие из предложенных\n";

cout << "1)Добавить студента\n";

cout << "2)Определить зачислены ли 2 студента в 1 дату\n";

cout << "3)Фамилия и дата рождения по номеру зачетки\n";

cout << "4)Вывести всех студентов\n";

cout << "5)Выход\n";

}

void Menu::OutPut()

{

unsigned int C, H;

char l[30];

char Nam[30];

char d[30];

char gro[50];

char d3[40];

unsigned int num1, num2;

Printr();

while(!Correct)

{

cin >> input;

if (input > 5 || input < 1)

cout << "Неккоректный ввод\n";

else

Correct = true;

}

while (!ex)

{

switch (input)

{

case 1:

{

cout << "Введите фамилию "; cin >> l; cout << endl;

cout << "Введите имя "; cin >> Nam; cout << endl;

cout << "Введите id "; cin >> C; cout << endl;

cout << "Введите номер телефона "; cin >> H; cout << endl;

cout << "Введите группу "; cin >> d3; cout << endl;

cout << "Введите дату рождения "; cin >> d; cout << endl;

cout << "Введите дату поступления "; cin >> gro; cout << endl;

Student P(C, H, l, Nam, d, gro, d3);

Add(base, length, &P);

break;

}

case 2:

cout << "Введите id 1\n";

cin >> num1;

cout << "Введите id 2\n";

cin >> num2;

if (st(num1, num2, base, length))

cout << "Студенты зачислены в 1 дату\n";

else

cout << "Студенты не зачислены в 1 дату\n";

break;

case 3:

cout << "Введите id\n";

cin >> input;

st1(input, base, length);

break;

case 4:

if (length == 0)

cout << "База данных пуста\n";

else

prnt(base, length);

break;

case 5:

ex = true;

break;

}

if(!ex)

{

Printr();

cin >> input;

}

}

}