САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ

УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ

Практическая работа №12

«Реализация наследования»

Выполнил: Мороз И.О.

Группа К3120

Проверил: Осипов Н.А.

Санкт-Петербург

2018 г.

# Упражнение 1.

1. Был создан заголовочный файл human.h, описывающий родительский класс

#pragma once

#include <string>

#include <sstream>

using namespace std;

class human {

public:

human(string last\_name, string name, string second\_name)

{

this->last\_name = last\_name;

this->name = name;

this->second\_name = second\_name;

}

string get\_full\_name()

{

ostringstream full\_name;

full\_name << this->last\_name << " "

<< this->name << " "

<< this->second\_name;

return full\_name.str();

}

private:

string name;

string last\_name;

string second\_name;

};

1. Был создан класс Student, наследуемый от human

#pragma once

#include "human.h"

#include <string>

#include <vector>

using namespace std;

class Student : public human {

public:

Student(string last\_name, string name, string second\_name,

vector<int> scores) : human(last\_name, name, second\_name) {

this->scores = scores;

}

float get\_average\_score()

{

unsigned int count\_scores = this->scores.size();

unsigned int sum\_scores = 0;

float average\_score;

for (unsigned int i = 0; i < count\_scores; ++i) {

sum\_scores += this->scores[i];

}

average\_score = (float)sum\_scores / (float)count\_scores;

return average\_score;

}

private:

vector<int> scores;

};

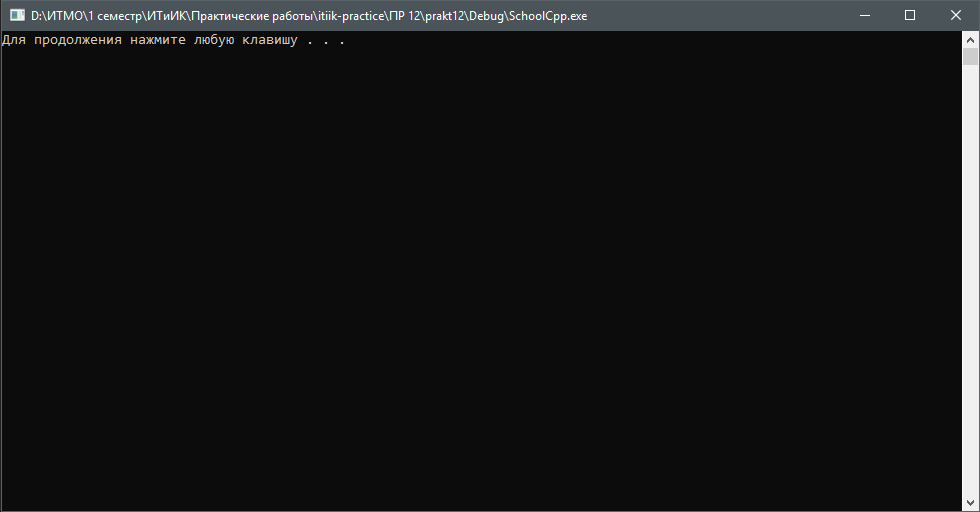


Рисунок 1 – проверка работоспособности программы

# Упражнение 2.

1. В файле main.cpp был создан объект класса Student. Ему были присвоены ФИО и оценки. Вывод на экран осуществлялся при помощи методов класса human get\_full\_name() и класса Student get\_average\_score()

#include <iostream>

#include <vector>

#include "human.h"

#include "Student.h"

using namespace std;

int main()

{

vector<int> scores;

scores.push\_back(5);

scores.push\_back(3);

scores.push\_back(4);

scores.push\_back(4);

scores.push\_back(5);

scores.push\_back(3);

scores.push\_back(3);

scores.push\_back(3);

scores.push\_back(3);

Student \*stud = new Student("Moroz", "Ilya", "Olegovich", scores);

cout << stud->get\_full\_name() << endl;

cout << "Average score: " << stud->get\_average\_score() << endl;

system("pause");

return 0;

}

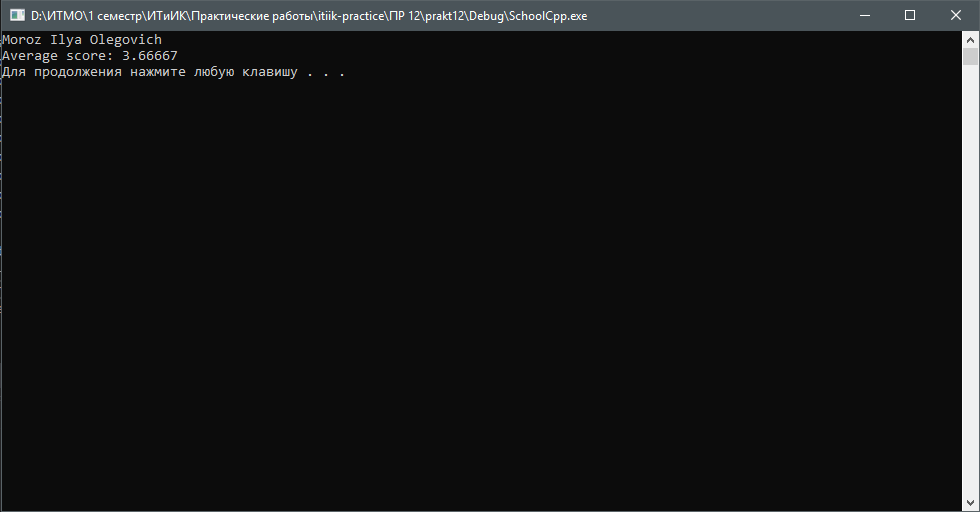


Рисунок 2 – результат работы программы

# Упражнение 3.

1. Был создан класс Teacher, наследуемый от human

#include "human.h"

#include <string>

using namespace std;

class Teacher : public human {

public:

Teacher(string last\_name, string name, string second\_name, unsigned int work\_time) : human(last\_name, name, second\_name) {

this->work\_time = work\_time;

}

unsigned int get\_work\_time()

{

return this->work\_time;

}

private:

unsigned int work\_time;

};

1. В файле main.cpp был создан объект класса Teacher. Работа с ним была реализована при помощи наследуемого метода get\_full\_name() и метода get\_work\_time()

unsigned int teacher\_work\_time = 40;

Teacher \*tch = new Teacher("Kupina", "Anna", "Pavlovna", teacher\_work\_time);

cout << tch->get\_full\_name() << endl;

cout << "Work time: " << tch->get\_work\_time() << endl;

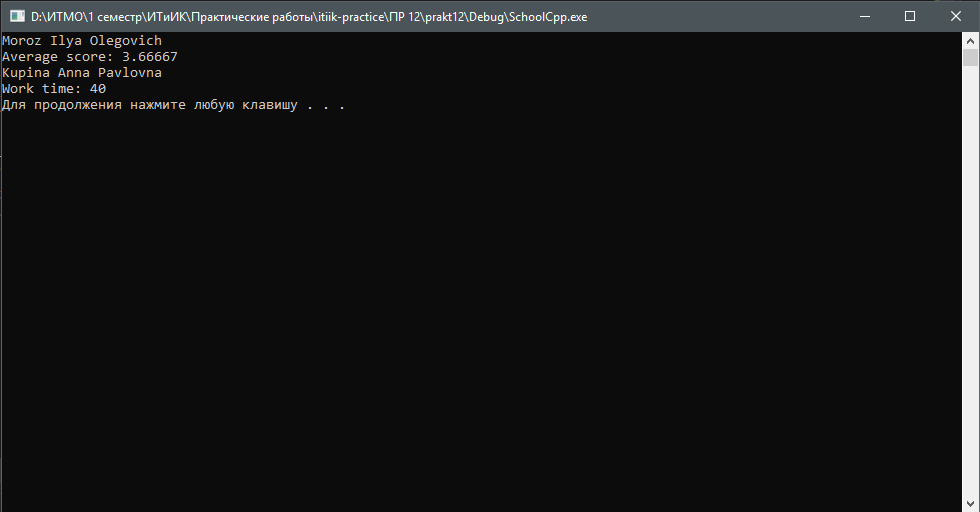


Рисунок 3 – результат работы программы

# Вывод

В ходе данной практической работы были рассмотрены возможности наследования в языке C++. Был создан родительский класс human, от которого наследовались классы Student и Teacher. При этом при работе с объектами этих классов были использованы конструктор и метод родительского класса.

# Контрольные вопросы

1. Можно ли вызвать метод родительского класса в переопределённом методе наследника?

Да.

void get\_info() {

human::get\_info();

cout << 1;

}

1. Что такое виртуальные методы?

Методы, обязательные дла переопределения в классе наследника.