Пермский Национальный Исследовательский  
Политехнический Университет

**Лабораторная работа № 2**

Информатика

за 2 семестр

Вариант № 1

Выполнил:

Студент группы

РИС 20-1-бз

Акиев В.В.

20-ЭТФ-641

Проверила:

Доцент кафедры ИТАС

Полякова О.А.

2021

**Классы и объекты. Использование конструкторов.**

1. **Цель задания**.

1.1 Создание консольного приложения, состоящего из нескольких файлов в системе программирования VisualStudio.

1.2. Создание объектов с использованием конструкторов.

1. **Задание.**

Пользовательский класс СТУДЕНТ

ФИО – string

Группа – string

Средний балл – int

1. **Файл Student.h**

#include <iostream>

#include <string>

using namespace std;

class Student

{

string FIO;

string group;

int averageScore;

public:

Student(); //конструктор без параметров

Student(string, string, int); // Конструктор с параметрами

Student(const Student&); // Конструктор копиррования

~Student(); // Деструктор

string get\_FIO(); // Селектор

void set\_FIO(string); // Модификатор

string get\_group(); // Селектор

void set\_group(string); // Модификатор

int get\_averageScore(); // Селектор

void set\_averageScore(int); // Модификатор

void show(); // Просмотр атрибутов

};

**файл Student.cpp**#include <iostream>

#include <string>

#include "Student.h"

using namespace std;

Student::Student() // Конструктор без параметров

{

FIO = "Акиев Вячеслав Викторович";

group = "РИС-20-1бз";

averageScore = 3;

cout << "Конструктор без параметров" << endl;

}

Student::Student(string N, string K, int S) // Конструктор с параметрами

{

FIO = N;

group = K;

averageScore = S;

cout << "\nКонструктор с параметрами" << endl;

}

Student::Student(const Student& t)

{

FIO = t.FIO;

group = t.group;

averageScore = t.averageScore;

cout << "Конструктор копирования" << endl;

}

Student::~Student() // Деструктор

{

cout << "\nДеструктор";

cout << endl;

}

string Student::get\_FIO() // Селекторы

{

return FIO;

}

string Student::get\_group()

{

return group;

}

int Student::get\_averageScore()

{

return averageScore;

}

void Student::set\_FIO(string N) // Модификаторы

{

FIO = N;

}

void Student::set\_group(string K)

{

group = K;

}

void Student::set\_averageScore(int S)

{

averageScore = S;

}

void Student::show()

{

cout << "ФИО: " << FIO << endl;

cout << "Группа: " << group << endl;

cout << "Средний балл: " << averageScore << endl;

}

1. **Файл Лабораторная\_2.cpp**

#include <iostream>

#include <string>

#include "Student.h"

using namespace std;

Student make\_student() // Функция для возврата обьекта как результат

{

string s;

string i;

int d;

cout << "Введите ФИО: ";

cin >> s;

cout << "Введите группу: ";

cin >> i;

cout << "Введите средний балл: ";

cin >> d;

Student t(s, i, d);

return t;

}

void print\_student(Student a) // Функция для передачи обьекта как параметра

{

a.show();

}

int main()

{

setlocale(LC\_ALL, "Rus");

Student t1; //конструктор без параметров

t1.show();

cout << endl;

Student t2("Акиев Вячеслав Викторович", "РИС-20-1бз", 3); // конструктор с параметрами

t2.show();

cout << endl;

Student t3 = t2; // конструктор копирования

t3.show();

t3.set\_FIO("Акиев Вячеслав Викторович"); // конструктор копирования

t3.set\_group("РИС-20-1бз");

t3.set\_averageScore(3);

cout << endl;

print\_student(t3); //конструктор копирования

cout << endl;

t1 = make\_student(); //конструктор копирования

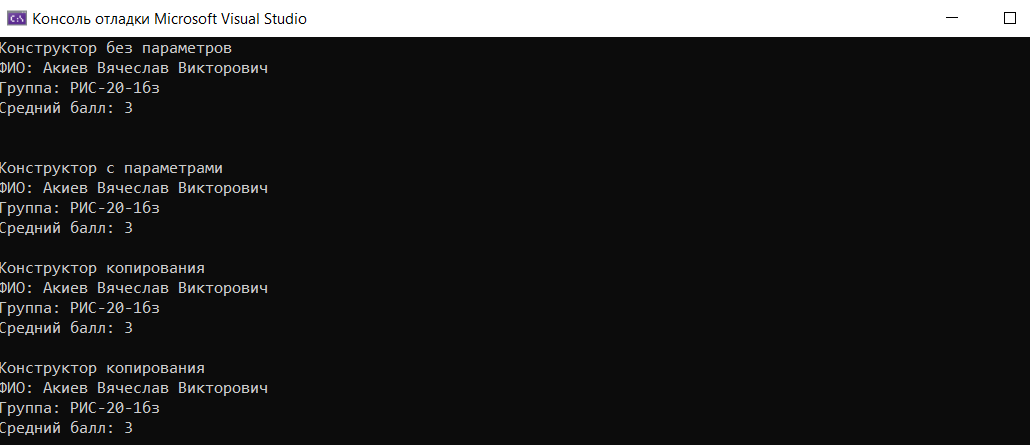
cout << endl;

t1.show();

return 0;

}

1. **Результат работы программы**

****

1. **Контрольные вопросы.**
2. **Для чего нужен конструктор?**

Одна из ключевых особенностей ООП — [инкапсуляция](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D0%BA%D0%B0%D0%BF%D1%81%D1%83%D0%BB%D1%8F%D1%86%D0%B8%D1%8F_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)): внутренние поля объекта напрямую недоступны, и пользователь может работать с объектом только как с единым целым, через открытые (public) методы. Каждый метод, в идеале, должен быть устроен так, чтобы объект, находящийся в «допустимом» состоянии (то есть когда выполняется [инвариант класса](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D0%B2%D0%B0%D1%80%D0%B8%D0%B0%D0%BD%D1%82_%D0%BA%D0%BB%D0%B0%D1%81%D1%81%D0%B0)), после вызова метода также оказался в допустимом состоянии. И первая задача конструктора — перевести поля объекта в такое состояние.

1. **Сколько типов конструктов существует в C++?**

1 Конструктор с параметрами

2. Конструктор без параметров

3. Конструктор копирования

1. **Для чего используется деструктор. В каких случаях деструктор описывается явно?**

Деструктор применяется для освобождения ресурсов, выделенных конструктором объекту. Описывать деструктор явным образом требуется в случае, когда объект содержит указатель память, выделяемую динамически.

1. **Для чего используется конструктор без параметров? Конструктор с параметрами?**

**Конструктор копирования?**

Конструктор с параметрами, используется для инициализации объекта требуемыми значениями.

Конструктор без параметров используется для создания «пустого» объекта.

Конструктор копирования – это специальный вид конструктора, получающий в качестве единственного параметра указатель на объект этого же класса.

1. **В каких случаях вызывается конструктор копирования?**

Конструктор копирования вызывается в тех случаях, когда новый объект создается путем копирования существующего:

- при описании нового объекта с инициализацией другим объектом;

- при передаче объекта в функцию по значению;

- при возврате объекта из функции.

1. **Перечислить свойства конструкторов.**
2. Конструктор не возвращает значение.
3. Конструктор не наследуется.
4. Конструкторы нельзя описывать с модификаторами const, virtual и static.
5. Конструктор копирования принимает объект того же класса по ссылке
6. Конструкторы глобальных объектов вызываются до функции main.

**7.Перечислите свойства деструктора.**

- не имеет аргументов и возвращаемого значения;

- не наследуется;

- не может быть объявлен как constили static;

- может быть виртуальным.

**8. К каким атрибутам имеют доступ методы класса?**

Методы класса имеют неограниченный доступ ко всем элементам класса, независимо от спецификаторов доступа и порядка объявления методов в классе.

**9. Что представляет собой указатель this?**

Ключевое слово this представляет собой неявно определенный указатель на сам объект. С его помощью метод класса определяет, с данными какого объекта ему предстоит работать.

**10. Какая разница между методами определенными внутри класса и вне касса?**

Определение метода внутри класса ничем не отличается от определения обычной функции. По умолчанию такой метод считается встроенной функцией (inline). Если метод определяется вне функции, то принадлежность метода классу указывается с помощью имени класса: Имя\_класса::Имя\_метода.

**11. Какое значение возвращает конструктор?**

Конструктор не возвращает значение.

**12. Какие методы создаются по умолчанию?**

В каждом классе должен быть хотя бы один метод, который предназначен для инициализации объекта. Его имя совпадает с именем класса, и он вызывается автоматически при инициализации объекта.

**13. Какое значение возвращает деструктор?**

Деструктор не возвращает значение.

**14. Дано описание класса**

ClassStudent

{

String name;

Int group;

Public:

Student(string, int);

Student(const student&)

~student();

};

Отсутствует конструктор без параметров.

**15. Какой метод будет вызван при выполнении следующих операторов**:

Student\*S

S=new student;

Указатель на пустой объект.

**16. Какой метод будет вызван при выполнении следующих операторов:**

Students(«Ivanov», 20);

Конструктор с параметрами.

**17. Какой метод будет вызван при выполнении следующих операторов:**

Students1(«Ivanov», 20); - Конструктор с параметрами

Students2=s1 - Конструктор копирования

**18. Какой метод будет вызван при выполнении следующих операторов:**

Students1(«Ivanov», 20); - Конструктор с параметрами

Students2

s2=s1 - Конструктор копирования

**19. Какой конструктор будет использоваться при передаче параметров в функцию print():**

Void print(student a)

{

a.show();

}

Ответ: конструктор без копирования.

**20. Класс описан следующим образом:**

class Student

{

String name;

Int age;

public:

void set\_name(string);

void\_set\_name(int);

};

Student p;

**Каким образом можно присвоить новое значение атрибуту name объекта p?**

Ответ: p.name="new\_name";