МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ПЕРМСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ

ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

(ПНИПУ)

Факультет: Электротехнический

Кафедра: «Информационные технологии и автоматизированные системы» (ИТАС)

Направление: Разработка информационных систем (РИС)

**О Т Ч Ё Т**   
**о лабораторной работе №2**

Дисциплина: «Основы алгоритмизации и программирования»

Тема: «Классы и объекты. Использование конструкторов.»

Выполнил:

Студент группы РИС-24-3б

Суяргулов А.О.

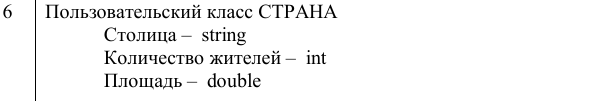
Проверил:

Доцент кафедры ИТАС

Полякова О.А.

Пермь, 2025

ВВЕДЕНИЕ



ОСНОВНАЯ ЧАСТЬ

**Блок-схема:**

**Код:**

#include <iostream>

#include <string>

// Объявляем, что будем использовать пространство имен std (стандартное)

using namespace std;

// Объявляем класс, который называется Country (Страна)

class Country {

// private: означает, что эти переменные можно использовать только внутри класса

private:

string capital; // Столица страны

int population; // Количество жителей в стране

double area; // Площадь страны

// public: означает, что эти функции (методы) можно использовать снаружи класса

public:

// Конструктор (специальная функция, которая вызывается при создании объекта)

// Этот конструктор - без параметров (параметры - это то, что передается в скобках)

Country() {

capital = ""; // Инициализируем столицу пустой строкой

population = 0; // Инициализируем население нулем

area = 0.0; // Инициализируем площадь нулем

cout << "Создана страна (без параметров)!" << endl; // Сообщаем, что конструктор сработал

}

// Конструктор с параметрами (какие-то значения передаются при создании страны)

Country(string cap, int pop, double ar) {

capital = cap; // Столица равна тому, что передали в cap

population = pop; // Население равно тому, что передали в pop

area = ar; // Площадь равна тому, что передали в ar

cout << "Создана страна (с параметрами)!" << endl; // Сообщаем, что конструктор сработал

}

// Конструктор копирования (создаем новую страну, копируя данные из другой страны)

Country(const Country& other) {

capital = other.capital; // Столица новой страны = столице другой страны

population = other.population; // Население новой страны = населению другой страны

area = other.area; // Площадь новой страны = площади другой страны

cout << "Создана копия страны!" << endl; // Сообщаем, что конструктор сработал

}

// Деструктор (специальная функция, которая вызывается, когда объект удаляется)

~Country() {

cout << "Страна уничтожена!" << endl; // Сообщаем, что деструктор сработал

}

// Функции для получения значений (геттеры) - позволяют узнать значения переменных

string getCapital() const {

return capital; // Возвращаем значение столицы

}

int getPopulation() const {

return population; // Возвращаем значение населения

}

double getArea() const {

return area; // Возвращаем значение площади

}

// Функции для установки значений (сеттеры) - позволяют изменить значения переменных

void setCapital(string newCapital) {

capital = newCapital; // Устанавливаем столицу в новое значение

}

void setPopulation(int newPopulation) {

population = newPopulation; // Устанавливаем население в новое значение

}

void setArea(double newArea) {

area = newArea; // Устанавливаем площадь в новое значение

}

// Функция для вывода информации о стране на экран

void printInfo() const {

cout << "Столица: " << capital << endl; // Выводим столицу

cout << "Население: " << population << endl; // Выводим население

cout << "Площадь: " << area << endl; // Выводим площадь

}

};

// Главная функция программы (с нее начинается выполнение)

int main() {

setlocale(LC\_ALL, "ru");

// Создаем объект Country (страну) с помощью конструктора без параметров

Country country1;

cout << "Информация о стране 1 (без параметров):" << endl;

country1.printInfo(); // Выводим информацию о стране

// Создаем объект Country (страну) с помощью конструктора с параметрами

Country country2("Moscow", 144000000, 17100000.0);

cout << "\nИнформация о стране 2 (с параметрами):" << endl;

country2.printInfo(); // Выводим информацию о стране

// Создаем объект Country (страну) с помощью конструктора копирования

Country country3 = country2; // Копируем данные из country2 в country3

cout << "\nИнформация о стране 3 (копия страны 2):" << endl;

country3.printInfo(); // Выводим информацию о стране

// Изменяем значения переменных в объекте country1 с помощью сеттеров

country1.setCapital("Washington");

country1.setPopulation(330000000);

country1.setArea(9800000);

cout << "\nИнформация о стране 1 (после изменений):" << endl;

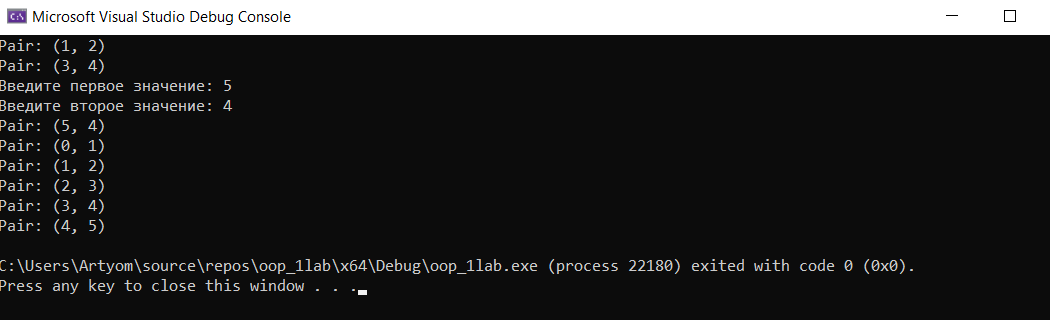
country1.printInfo(); // Выводим информацию о стране

// Возвращаем 0, чтобы сообщить, что программа завершилась успешно

return 0;

}

**Результат работы кода:**

****

**Контрольные вопросы:**

1. Для чего нужен конструктор?

Конструктор нужен для инициализации объекта класса при его создании. Он выделяет память и присваивает начальные значения полям объекта.

2. Сколько типов конструкторов существует в C++?

В C++ существуют следующие типы конструкторов:

• Конструктор по умолчанию

• Конструктор с параметрами.

• Конструктор копирования.

• Конструктор перемещения

3. Для чего используется деструктор? В каких случаях деструктор описывается явно?

Деструктор используется для освобождения памяти и других ресурсов, выделенных объектом класса при его существовании. Деструктор описывается явно, когда необходимо выполнить какие-либо специфические действия перед уничтожением объекта, например, освободить память, закрыть файлы и т.п.

4. Для чего используется конструктор без параметров? Конструктор с параметрами? Конструктор копирования?

• Конструктор без параметров используется для создания объекта без каких-либо начальных значений.

• Конструктор с параметрами используется для создания объекта и инициализации его полей определенными значениями, переданными в качестве аргументов.

• Конструктор копирования используется для создания нового объекта, который является копией существующего объекта.

5. В каких случаях вызывается конструктор копирования?

Конструктор копирования вызывается в следующих случаях:

• При создании нового объекта и инициализации его существующим объектом того же класса (например, Student s2 = s1;).

• При передаче объекта в функцию по значению

• При возврате объекта из функции по значению

6. Перечислить свойства конструкторов.

• Имя конструктора совпадает с именем класса.

• Конструктор не имеет возвращаемого значения

• Конструктор может иметь параметры.

• Класс может иметь несколько конструкторов

• Если конструктор не определен явно, компилятор создает конструктор по умолчанию

7. Перечислить свойства деструкторов.

• Имя деструктора совпадает с именем класса, но начинается с символа тильды (~).

• Деструктор не имеет параметров.

• Деструктор не имеет возвращаемого значения.

• В классе может быть только один деструктор.

• Если деструктор не определен явно, компилятор создает деструктор по умолчанию.

8. К каким атрибутам имеют доступ методы класса?

Методы класса имеют доступ ко всем атрибутам (членам) класса: как к публичным (public), так и к приватным (private) и защищенным (protected).

9. Что представляет собой указатель this?

Указатель this - это неявный указатель, который передается каждому нестатическому методу класса. Он указывает на объект, для которого был вызван этот метод. Он позволяет получить доступ к членам объекта изнутри метода.

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

Методы, определенные внутри класса, автоматически считаются inline (если компилятор сочтет это оптимальным). Методы, определенные вне класса, должны быть явно объявлены как inline, если это требуется. Разница в области видимости и возможности оптимизации inline функцией.

11. Какое значение возвращает конструктор?

Конструктор не возвращает никакого значения (даже void не указывается). Его задача - инициализировать объект, а не возвращать значение.

12. Какие методы создаются по умолчанию?

Компилятор может автоматически сгенерировать:

• Конструктор по умолчанию

• Конструктор копирования

• Оператор присваивания копированием

• Деструктор

• Конструктор перемещения

• Оператор присваивания перемещением

13. Какое значение возвращает деструктор?

Деструктор не возвращает никакого значения

14. Дано описание класса Student ...

• Какой метод отсутствует в описании класса?

В описании класса отсутствует оператор присваивания копированием (operator=). Также, начиная с C++11, отсутствует конструктор перемещения и оператор присваивания перемещением.

15. Какой метод будет вызван при выполнении следующих операторов: Student\* s; s = new Student;

Будет вызван конструктор по умолчанию (Student()).

16. Какой метод будет вызван при выполнении следующих операторов: Student s("Ivanov", 20);

Будет вызван конструктор Student(string, int).

17. Какие методы будут вызваны при выполнении следующих операторов: Student s1("Ivanov", 20); Student s2 = s1;

• Student s1("Ivanov", 20);: Вызов конструктора Student(string, int).

• Student s2 = s1;: Вызов конструктора копирования Student(const Student&).

18. Какие методы будут вызваны при выполнении следующих операторов: Student s1("Ivanov", 20); Student s2; s2 = s1;

• Student s1("Ivanov", 20);: Вызов конструктора Student(string, int).

• Student s2;: Вызов конструктора по умолчанию Student().

• s2 = s1;: Вызов оператора присваивания копированием).**Ссылка на GitHub:**