Министерство науки и высшего образования Российской Федерации Федеральное государственное автономное учреждение высшего образования «Пермский национальный исследовательский политехнический университет»

ПНИПУ

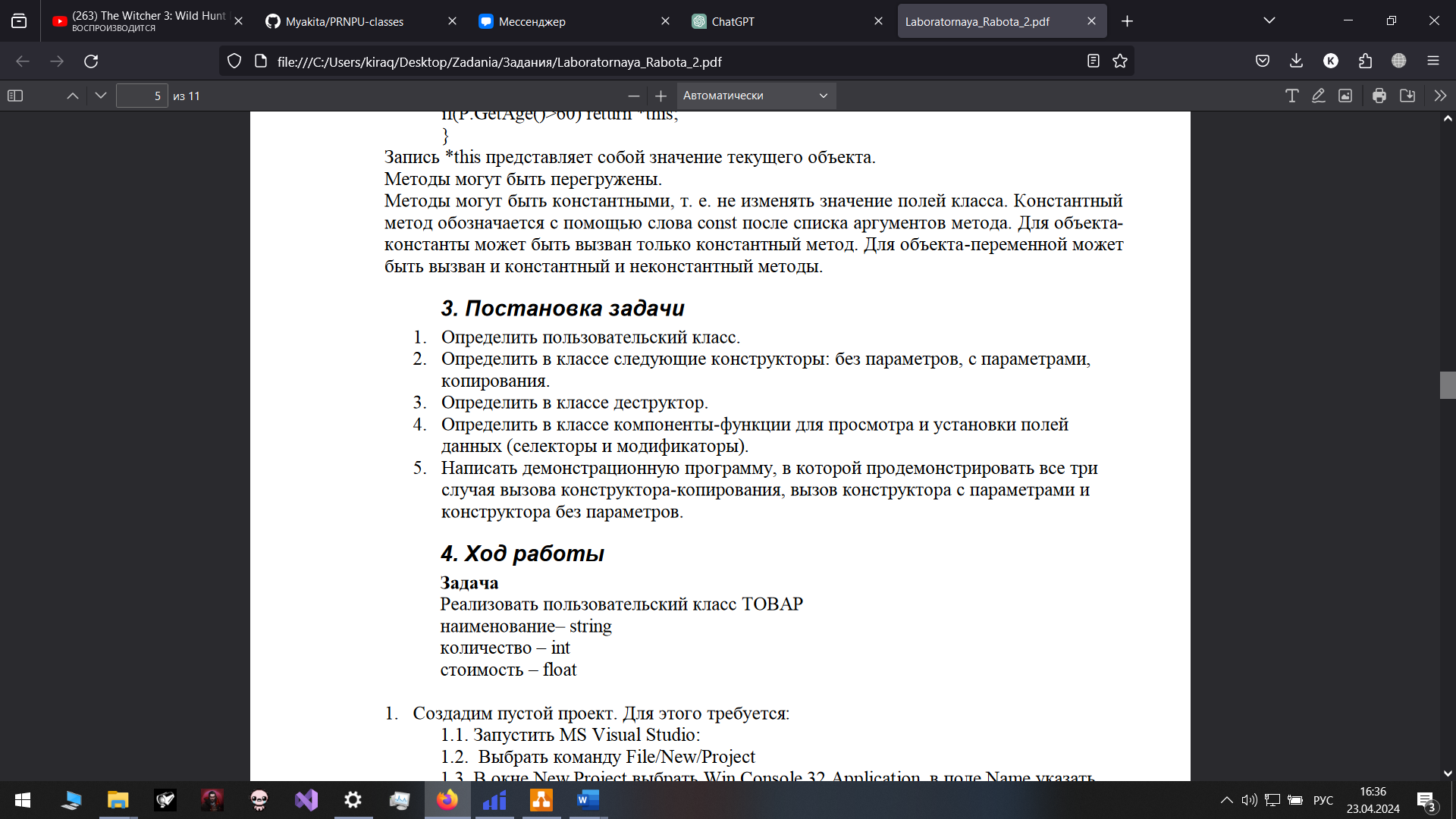
Лабораторная работа  
«Использование конструкторов»

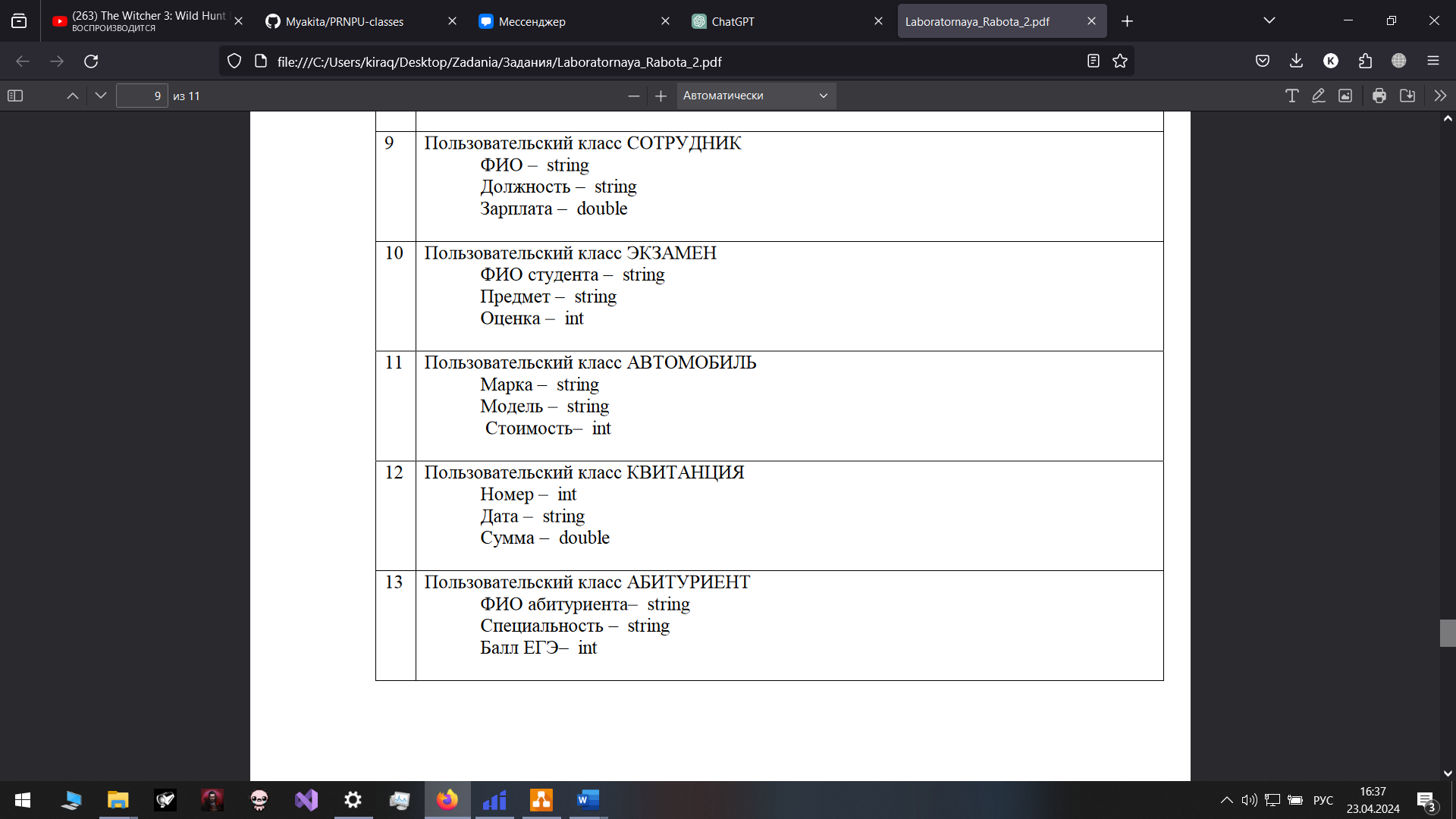
Выполнил:   
студент группы РИС-23-1б   
Мокрушин Никита Дмитриевич

Проверила:   
доцент кафедры ИТАС   
О.А. Полякова

Пермь 2024 г.

Постановка задачи:





Анализ задачи:

* Конструктор по умолчанию automobil(), инициализирующий поля значениями по умолчанию ("noname" для марки и модели, 0 для цены). Выводит информацию о созданном объекте.
* Конструктор с параметрами automobil(string, string, int), принимающий значения марки, модели и цены. Использует переданные значения для инициализации соответствующих полей. Также выводит информацию о созданном объекте.
* Конструктор копирования automobil(const automobil&), который создает копию существующего объекта, инициализируя поля значениями из объекта-аргумента. Также выводит информацию о созданном объекте.
* SetMARKA, SetMODEL, SetPRICE - для установки значений марки, модели и цены.
* GetMARKA, GetMODEL, GetPRICE - для получения значений марки, модели и цены.
* Print - для вывода информации об автомобиле на экран.
* Деструктор ~automobil(), выводящий сообщение о удалении объекта. Создание объектов класса automobil с использованием различных конструкторов и методов для установки значений полей.
* Вывод информации о созданных объектах.

Код:

#include<iostream>

#include<string>

using namespace std;

class automobil {

private:

string Marka;

string Model;

int price;

public:

automobil() {

Marka = "noname";

Model = "noname";

price = 0;

cout << "Вызовконструктораспараметрами: " << this->Marka << " " << this->Model << " " << this->price << endl;

}

automobil(string Marka, string Model, int price) {

this->Marka = Marka;

this->Model = Model;

this->price = price;

cout << "Вызовконструктораспараметрами: " << this->Marka << " " << this->Model << " "<< this->price << endl;

}

automobil(const automobil& d) {

Marka = d.Marka;

Model = d.Model;

price = d.price;

cout << "Вызовконструктораспараметрами: "<< this->Marka << " " << this->Model << " " << this->price << endl;

}

void SetMARKA(string name) {

Marka = name;

}

void SetMODEL(string model) {

Model = model;

}

void SetPRICE(double price) {

price = price;

}

string GetMARKA() {

return Marka;

}

string GetMODEL() {

return Model;

}

int GetPRICE() {

return price;

}

void Print() {

cout << "Марка: " << Marka << endl << "Модель: " << Model << endl << "Стоимость: " << price << endl;

}

~automobil() {

cout << "Удаление коснтруктора" << endl;

}

};

int main() {

system("chcp 1251>NULL");

automobil h1;

automobil h2("Молодых Никита Андреевич", "Бог", 77);

automobil h3(h2);

automobil h4;

h4.SetMARKA("Сингур Иван Сергеевич");

h4.SetMODEL("Чушпан");

h4.SetPRICE(48);

cout << endl;

cout << "Марка: " << h4.GetMARKA() << endl;

cout << "Модель: " << h4.GetMODEL() << endl;

cout << "Стоимость: " << h4.GetPRICE() << endl;

cout << endl;

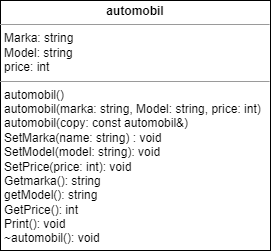
//h4.Print();

cout << endl;

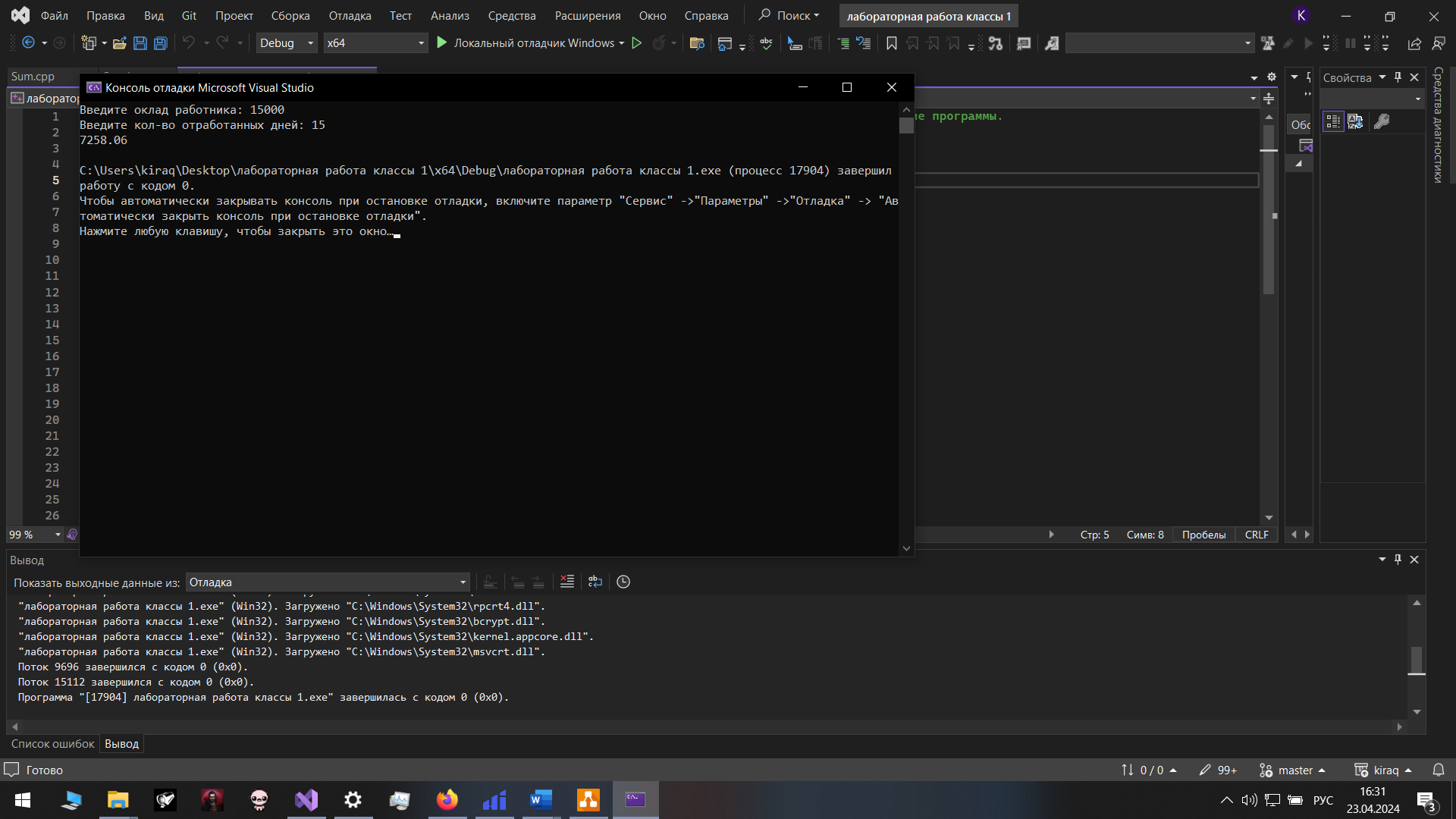
return 0;

}

UML Диаграммы:



Вывод:



программа работает корректно и выдаёт требуемый результат.  
Ответы на вопросы:

1. Конструктор в языке программирования C++ используется для инициализации объектов класса при их создании. Он выполняет необходимые действия по установке начальных значений переменных класса или выполнению других операций, необходимых для корректной работы объекта.
2. В C++ существует три типа конструкторов:
   * Конструктор без параметров (по умолчанию).
   * Конструктор с параметрами.
   * Конструктор копирования.
3. Деструктор в C++ используется для освобождения ресурсов, выделенных объекту класса, перед его уничтожением. Он вызывается при завершении срока жизни объекта и позволяет выполнить необходимые операции по очистке ресурсов. Деструктор описывается явно, когда объект класса использует динамическое выделение памяти или другие ресурсы, которые требуют освобождения.
   * Конструктор без параметров используется для создания объекта без необходимости передачи начальных значений.
   * Конструктор с параметрами используется для создания объекта и передачи начальных значений его атрибутам.
   * Конструктор копирования используется для создания копии объекта.
4. Конструктор копирования вызывается при создании копии объекта, например, при передаче объекта в функцию по значению или при возвращении объекта из функции.
5. Свойства конструкторов:
   * Имя конструктора совпадает с именем класса.
   * Не имеют возвращаемого значения.
   * Могут быть перегружены.
6. Свойства деструкторов:
   * Имя деструктора совпадает с именем класса, но с префиксом ~.
   * Не имеют параметров.
   * Не имеют возвращаемого значения.
7. Методы класса имеют доступ ко всем атрибутам класса, включая приватные.
8. Указатель this представляет адрес текущего объекта и используется для доступа к его атрибутам и методам внутри методов класса.
9. Методы, определенные внутри класса, автоматически являются методами класса, тогда как методы, определенные вне класса, должны быть явно связаны с классом с использованием оператора области разрешения ::.
10. Конструктор не возвращает значение, он инициализирует объект.
11. Методы, создаваемые по умолчанию, включают в себя конструктор по умолчанию, конструктор копирования, оператор присваивания и деструктор.
12. Деструктор также не возвращает значение, он просто освобождает ресурсы.
13. В описании класса отсутствует конструктор без параметров.
14. При выполнении оператора student \*s; s = new student; вызывается конструктор без параметров.
15. При выполнении оператора student s("Ivanov", 20); вызывается конструктор с параметрами.
16. При выполнении оператора student s1("Ivanov", 20); student s2 = s1; вызывается конструктор копирования.
17. При выполнении оператора student s1("Ivanov", 20); student s2; s2 = s1; вызывается конструктор с параметрами для s1 и оператор присваивания для s2.
18. При передаче параметра в функцию print(), будет использоваться конструктор копирования, чтобы создать копию объекта student.
19. Для присвоения нового значения атрибуту name объекта p класса Student, используется метод set\_name():