Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования

Пермский национальный исследовательский политехнический университет

Электротехнический факультет

Кафедра информационных технологий и автоматизированных систем

**ОТЧЕТ**

**Тема:** Лабораторная работа №2 по ООП

Семестр: 2

Выполнил студент ИВТ-23-2б:

Хомутов Сергей Алексеевич

(дата, подпись)

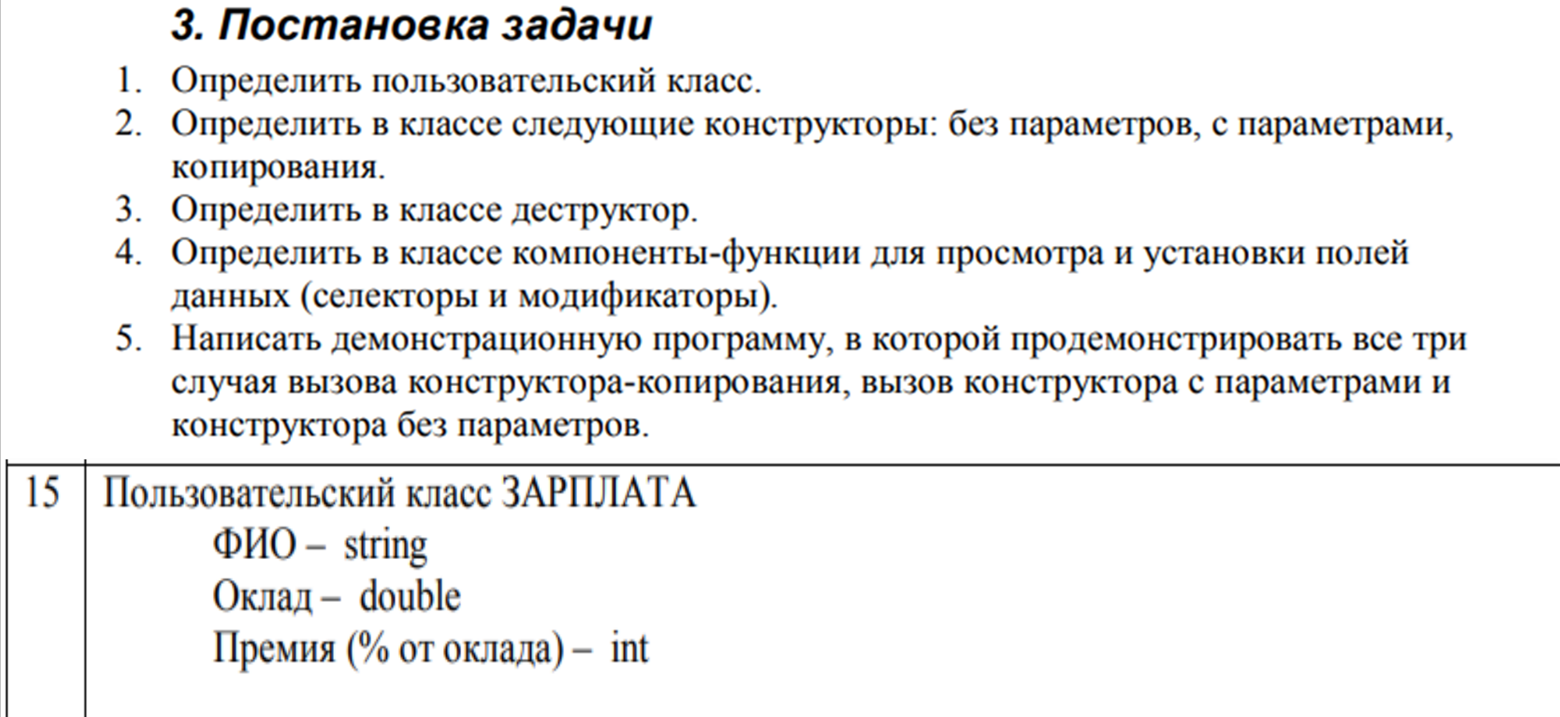
Проверила:

Полякова Ольга Андреевна

(дата, подпись)

Пермь 2024

**Задание**



**Анализ задачи**

1. Метод SetName – установление имени
2. Метод SetReward – установление зарплаты
3. Метод SetPrcnt – установление оклада
4. Метод GetPrcnt – вывод оклада
5. Метод Show – вывод данных

**UML-Диаграмма**

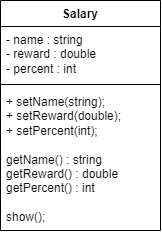
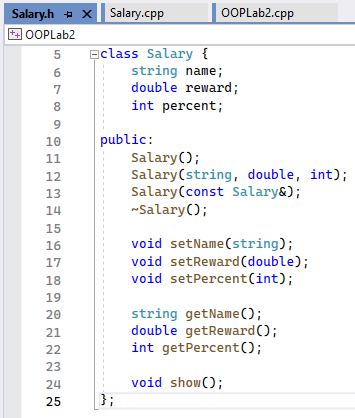


Рисунок 1 – UML-диаграмма класса «Salary»

**Программное решение**





#include <iostream>

#include <string>

#include "Salary.h"

using namespace std;

Salary::Salary() {

cout << "Вызов конструктора без параметров" << '\n';

name = "";

reward = 0;

percent = 0;

}

Salary::Salary(string s, double d, int i) {

cout << "Вызов конструктора с параметрами" << '\n';

name = s; reward = d; percent = i;

}

Salary::Salary(const Salary& t) {

cout << "Вызов конструктора копирования" << '\n';

name = t.name;

reward = t.reward;

percent = t.percent;

}

Salary::~Salary() {

cout << "Вызов деструктора" << '\n';

}

void Salary::setName(string name) {

this->name = name;

}

void Salary::setReward(double reward) {

this->reward = reward;

}

void Salary::setPercent(int percent) {

percent = percent;

}

string Salary::getName() {

return name;

}

double Salary::getReward() {

return reward;

}

int Salary::getPercent() {

return percent;

}

void Salary::show() {

cout << "Name: " << name << '\n';

cout << "Reward: " << reward << '\n';

cout << "Percent: " << percent << '\n';

}



#include <iostream>

#include <string>

#include "Salary.h"

using namespace std;

Salary makeStaff();

void print(Salary);

int main() {

system("chcp 1251 > NULL");

Salary s1; // конструктор без параметров

s1.show();

Salary s2("Антонов Дмитрий Сергеевич", 200054, 45); // Конструктор с параметрами

s2.show();

Salary s3 = s2; // Конструктор копирования

print(s3); // конструктор копирования

// Вызов деструктора (так как в параметрах функции не ссылка)

Salary s4 = makeStaff();

s4.show();

// Деструктор вызывается для каждого объекта

return 0;

}

Salary makeStaff() {

string name;

double r;

int i;

cout << "Введите ФИО сотрудника:" << '\n';

getline(cin, name);

cout << "Зарплата:" << '\n';

cin >> r;

cout << "Премия%:" << '\n';

cin >> i;

Salary t(name, r, i); // конструктор с параметрами

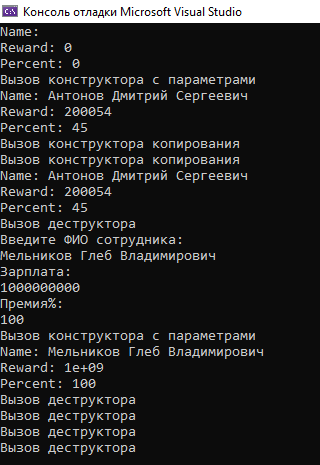
return t; // конструктор копирования, затем деструктор, так как заканчивается время жизни

}

void print(Salary t) {

t.show();

}



**Контрольные вопросы**

1. Для чего нужен конструктор?

Чтобы избежать ошибки, связанные с использованием неинициализированных переменных.

2. Сколько типов конструкторов существует?

3: конструктор с параметрами, без параметров и конструктор копирования

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

Для освобождения памяти, выделенной под объект. Явно используется при выделении динамической памяти.

4-5. Для чего конструктор без параметров? С параметрами? Копирования?

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

- при описании нового объекта с инициализацией другим объектом;

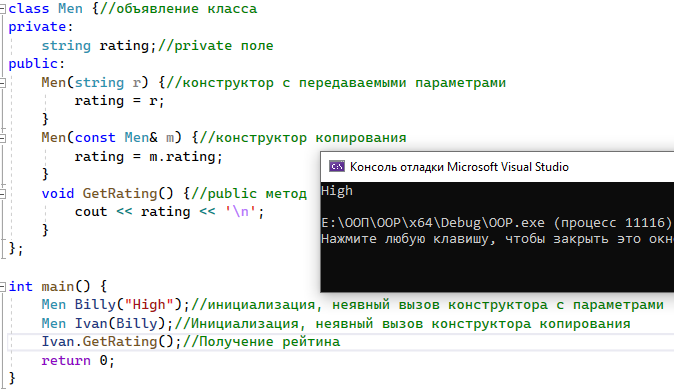
- при передаче объекта в функцию по значению;

- при возврате объекта из функции.

Конструктор без параметров нужен для создания “пустого” экземпляра класса.

Конструктор с параметрами для инициализации объекта с заданными параметрами.

Пример:



6. Перечислите свойства конструкторов

* Конструктор должен иметь то же имя, что и класс
* Класс может иметь несколько конструкторов с разными параметрами (конструктор можно перегружать)
* Конструктор не возвращает значение
* Если конструктор не был указан явно, то компилятор создаст пустой конструктор автоматически

7. Перечислите свойства деструкторов

* Должен иметь то же имя, что и класс, но со знаком тильда (~) перед именем
* Не принимает аргументы
* Не возвращает значение
* Вызывается автоматически, когда объект выходит из области видимости
* Если деструктор не был указан явно, то компилятор создаст пустой

деструктор автоматически

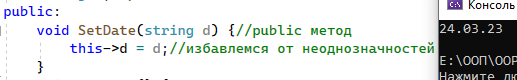
8. К каким атрибутам имеют доступ методы класса

Ко всем.

9. Что представляет собой запись: this

это скрытый константный для адреса указатель, содержащий адрес объекта.

Пример:



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

Для метода, определённого вне класса, нужно прописать пространство имен класса.

11. Какое значение возвращает конструктор?

Никакое.

12. Какие методы создаются по умолчанию?

Конструкторы, деструктор.

13. Какое значение возвращает деструктор?

Никакое.

14. Конструктор без параметров

15. Конструктор без параметров

16. Конструктор с параметрами

17. Конструктор с параметрами, копирования

18. Конструктор с параметрами, без параметров, копирования.

19. Конструктор копирования.

20. С помощью сеттера set\_name(string);