Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«**Пермский национальный исследовательский политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**ОТЧЕТ**

Дисциплина: «Основы алгоритмизации и программирования»

# Лабораторная работа №2

# « Классы и объекты. Использование конструкторов »

# Семестр 2

Выполнил работу

Студент группы РИС-22-1Б

Юхновец В.Г.

Проверил

Доцент кафедры ИТАС

Полякова О.А.

Г. Пермь-2023

**Постановка задачи**

1. Определить пользовательский класс.

2. Определить в классе следующие конструкторы: без параметров, с параметрами, копирования.

3. Определить в классе деструктор.

4. Определить в классе компоненты-функции для просмотра и установки полей данных (селекторы и модификаторы).

5. Написать демонстрационную программу, в которой продемонстрировать все три случая вызова конструктора-копирования, вызов конструктора с параметрами и конструктора без параметров.

**Вариант 15**

Пользовательский класс ЗАРПЛАТА:  
ФИО – string   
Оклад – double   
Премия (% от оклада) – int

**Ответы на контрольные вопросы**

1. **Для чего нужен конструктор?** Конструктор в языке программирования С++ нужен для инициализации объектов класса.
2. **Сколько типов конструкторов существует в С++?** В языке С++ существует три типа конструкторов: конструктор без параметров (default constructor), конструктор с параметрами (parameterized constructor) и конструктор копирования (copy constructor).
3. **Для чего используется деструктор? В каких случаях деструктор описывается явно?** Деструктор в С++ используется для освобождения памяти, занятой объектом класса. Деструктор описывается явно в тех случаях, когда объект класса содержит динамические данные, которые должны быть освобождены при уничтожении объекта.
4. **Для чего используется конструктор без параметров? Конструктор с параметрами? Конструктор копирования?** Конструктор без параметров используется для создания объекта класса с значениями по умолчанию. Конструктор с параметрами используется для создания объекта класса с заданными значениями атрибутов. Конструктор копирования используется для создания копии объекта класса.
5. **В каких случаях вызывается конструктор копирования?** Конструктор копирования вызывается в тех случаях, когда объект класса передается в качестве аргумента в функцию по значению, возвращается из функции объект класса по значению или происходит инициализация одного объекта класса другим объектом класса.
6. **Перечислить свойства конструкторов.** Свойства конструкторов:

* Имя конструктора совпадает с именем класса
* Не имеют возвращаемого значения
* Могут быть перегружены

1. **Перечислить свойства деструкторов.** Свойства деструкторов:

* Имя деструктора совпадает с именем класса с добавлением символа ~ в начале
* Не имеют параметров и возвращаемого значения
* Могут быть перегружены
* Вызываются автоматически при уничтожении объекта

1. **К каким атрибутам имеют доступ методы класса?** Методы класса имеют доступ к атрибутам объекта класса.
2. **Что представляет собой указатель this?** Указатель this представляет адрес объекта, для которого был вызван метод.
3. **Какая разница между методами определенными внутри класса и вне класса?** Методы, определенные внутри класса, имеют доступ к приватным и защищенным атрибутам класса, а методы, определенные вне класса, могут иметь доступ только к публичным атрибутам класса.
4. **13. 14. 15. 16. 17. 18. 19. 20.**
5. **Какое значение возвращает конструктор?** Конструктор не возвращает никакого значения.
6. **Какие методы создаются по умолчанию?** Методы, создаваемые по умолчанию: конструктор без параметров, конструктор копирования, деструктор, оператор присваивания и оператор сравнения.
7. **Какое значение возвращает деструктор?** Деструктор не возвращает никакого значения.
8. **Дано описание класса class Student { string name; int group; public: student(string, int); student(const student&) ~student(); }; Какой метод отсутствует в описании класса?** В описании класса отсутствует определение метода конструктора по умолчанию.
9. **Какой метод будет вызван при выполнении следующих операторов: student\*s; s=new student;** Будет вызван конструктор без параметров.
10. **Какой метод будет вызван при выполнении следующих операторов: student s(“Ivanov”,20);** Будет вызван конструктор с параметрами, переданными в качестве аргументов.
11. **Какие методы будут вызваны при выполнении следующих операторов: student s1(“Ivanov”,20); student s2=s1;** Будет вызван конструктор копирования.
12. **Какие методы будут вызваны при выполнении следующих операторов: student s1(“Ivanov”,20); student s2; s2=s1;** Будет вызван конструктор с параметрами затем будет вызван конструктор копирования для создания объекта s2 на основе объекта s1, а затем будет вызван оператор присваивания для копирования значений полей объекта s1 в объект s2.
13. **Какой конструктор будет использоваться при передаче параметра в функцию print(): void print(student a) {a.show();}** Будет вызван конструктор копирования, так как объект a передается в функцию print() по значению, а не по ссылке.
14. **Класс описан следующим образом: class Student { string name; int age; public: void set\_name(string); void set\_age(int ); ….. }; Student p; Каким образом можно присвоить новое значение атрибуту name объекта р?** Для того чтобы присвоить новое значение атрибуту name объекта р, можно использовать метод set\_name(), например: p.set\_name("New name"), а также метод set\_age.

**Диаграмма класса**

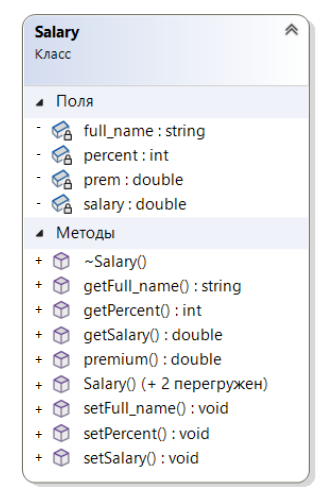


Рисунок 1 – класс Salary

**Описание класса**

#pragma once

#include <iostream>

#include <string>

using namespace std;

class Salary

{

friend ostream& operator<<(ostream& os, const Salary& salary);

private:

string full\_name = "Null";

double salary = 0;

int percent = 0;

double prem = 0;

public:

Salary();

Salary(string full\_name, double salary, int percent);

Salary(Salary const& object);

void setFull\_name(string full\_name);

void setSalary(double salary);

string getFull\_name();

double getSalary();

int getPercent();

void setPercent(int percent);

double premium(); //процент от оклада

~Salary();

};

**Определение методов**

#include "Header.h"

Salary::Salary()

{

cout << "Вызвался конструктор по умолчанию для объекта: " << this << endl;

full\_name = "NULL";

salary = 0;

percent = 0;

prem = 0;

}

Salary::Salary(string full\_name, double salary, int percent)

{

cout << "Вызвался конструктор с параметрами для объекта: " << this << endl;

this->full\_name = full\_name;

this->salary = salary;

this->percent = percent;

premium();

}

Salary::Salary(const Salary& object)

{

cout << "Вызвался конструктор копирования для объекта: " << this << endl;

cout << "Копирование объекта: " << &object << endl;

this->full\_name = object.full\_name;

this->salary = object.salary;

this->percent = object.percent;

this->prem = object.prem;

}

void Salary::setFull\_name(string full\_name)

{

this->full\_name = full\_name;

}

void Salary::setSalary(double salary)

{

this->salary = salary;

}

string Salary::getFull\_name()

{

return full\_name;

}

double Salary::getSalary()

{

return salary;

}

int Salary::getPercent()

{

return percent;

}

void Salary::setPercent(int percent)

{

this->percent = percent;

}

double Salary::premium()

{

prem = (salary / 100) \* percent;

return prem;

}

Salary::~Salary()

{

}

**Результаты работы программы**

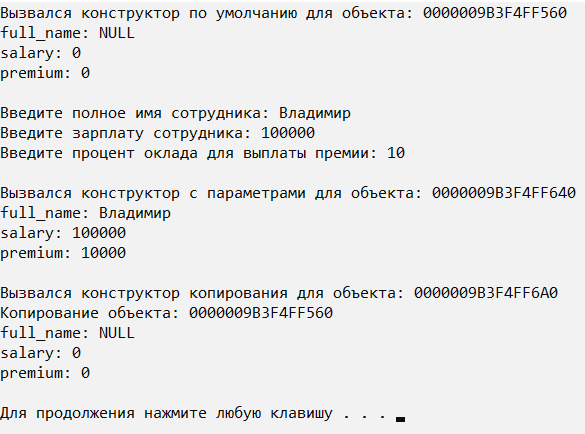


Рис.2 – результат работы программы