МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

ПЕРМСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ  
ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Кафедра информационные технологии и автоматизированные системы

**Дисциплина Информатика**

**Использование конструкторов**

Выполнил студент ИВТ-22-2б:

Мифтахов Марат Ринатович

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Проверил доцент кафедры ИТАС:

Полякова Ольга Андреевна

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Пермь, 2023

**Постановка задачи**

1. Реализовать определение нового класса.
2. Определить конструкторы.
3. Определить деструктор.
4. Определить сеттеры и геттеры.
5. Для демонстрации работы с объектами написать главную функцию.

**Задание**

Пользовательский класс зарплата:

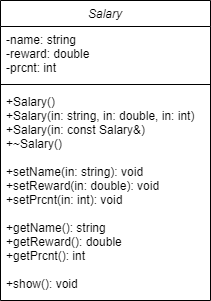
-ФИО: string,

-Оклад: double,

-Премия: int.

**Диаграмма класса**

Ниже представлена UML-диаграмма класса Salary.



**Код программы**

В файле Salary.h:

#pragma once

#include <string>

using namespace std;

class Salary {

string name;

double reward;

int prcnt;

public:

Salary();

Salary(string, double, int);

Salary(const Salary&);

~Salary();

// Модификаторы

void setName(string);

void setReward(double);

void setPrcnt(int);

// Селекторы

string getName();

double getReward();

int getPrcnt();

void show();

};

В файле Salary.cpp:

#include <iostream>

#include <string>

#include <Salary.h>

using namespace std;

Salary::Salary() {

cout << "Вызов конструктора без параметров" << endl;

name = "";

reward = 0;

prcnt = 0;

}

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

cout << "Вызов конструктора с параметрами" << endl;

name = s; reward = d; prcnt = i;

}

Salary::Salary(const Salary& t) {

cout << "Вызов конструктора копирования" << endl;

name = t.name;

reward = t.reward;

prcnt = t.prcnt;

}

Salary::~Salary() {

cout << "Вызов деструктора" << endl;

cout << name << endl;

}

void Salary::setName(string name) {

this->name = name;

}

void Salary::setReward(double reward) {

this->reward = reward;

}

void Salary::setPrcnt(int prcnt) {

prcnt = prcnt;

}

string Salary::getName() {

return name;

}

double Salary::getReward() {

return reward;

}

int Salary::getPrcnt() {

return prcnt;

}

void Salary::show() {

cout << "Name: " << name << endl;

cout << "Reward: " << reward << endl;

cout << "Percent: " << prcnt << endl;

}

В файле main.cpp:

#include <iostream>

#include <Windows.h>

#include <string>

#include <Salary.h>

using namespace std;

Salary makeStaff();

void print(Salary);

int main() {

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

Salary s1; // конструктор без параметров

s1.show();

Salary s2("Графинин Граф Евграфович", 200000, 34); // Конструктор с параметрами

s2.show();

Salary s3 = s2; // Конструктор копирования

print(s3); // конструктор копирования

// Вызов деструктора (так как в пар-рах 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 имя?" << endl;

getline(cin, name);

cout << "Зарплата?" << endl;

cin >> r;

cout << "Премия (%)?" << endl;

cin >> i;

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

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

}

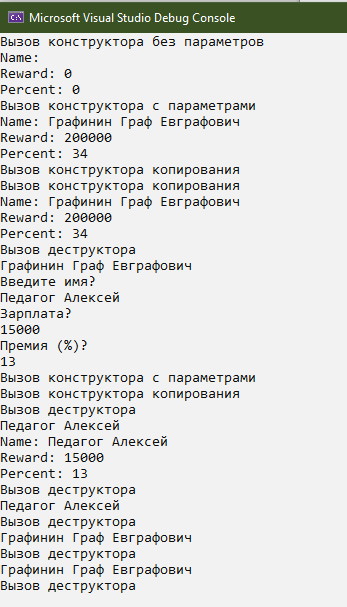
void print(Salary t) {

t.show();

}

**Вывод программы**

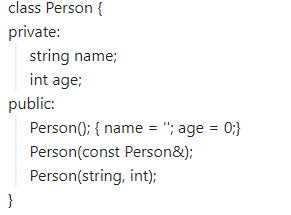
На рисунке 2 представлен вывод программы на консоль.

Рисунок 2 - Вывод программы

**Ответы на контрольные вопросы**

1. Для чего нужен конструктор?

Ответ: для инициализации значений экземпляра при создании объекта.



1. Сколько типов конструкторов существует в С++?

Ответ: 3

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

Ответ: для очистки динамической памяти, если таковая имелась в классе

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

Ответ: Конструкторы созданы для задания значений полям, поэтому:

первый задает значения null и 0, второй задает значения параметров,

третий копирует значения из одного экземпляра в другой.

1. В каких случаях вызывается конструктор копирования?

Ответ: в случаях копирования одного объекта в другой

1. Перечислить свойства конструкторов.

Ответ: не возвращает значение, не наследуется,

1. Перечислить свойства деструкторов.

Ответ: не возвращает значение, не наследуется,

1. К каким атрибутам имеют доступ методы класса?

Ответ: ко всем.

1. Что представляет собой указатель this?

Ответ: указатель на данный объект класса.

1. Какая разница между методами определенными внутри класса и вне класса?

Ответ: во втором случае нужно предоставить пространство имен класса

1. Какое значение возвращает конструктор?

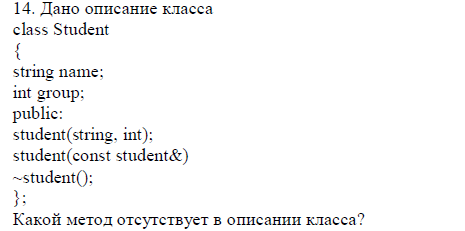
Ответ: никакое.

1. Какие методы создаются по умолчанию?

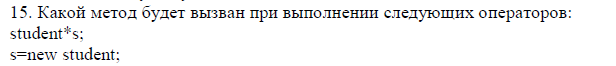
Ответ: конструкторы, деструктор.

1. Какое значение возвращает деструктор?

Ответ: никакое.

\_

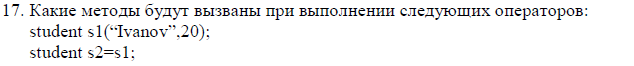
Ответ: конструктор без параметров

\_

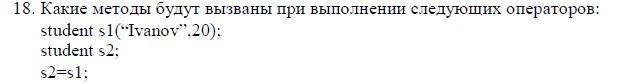
Ответ: конструктор без параметров

\_

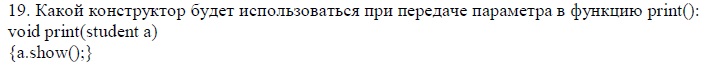
Ответ: конструктор с параметрами

\_

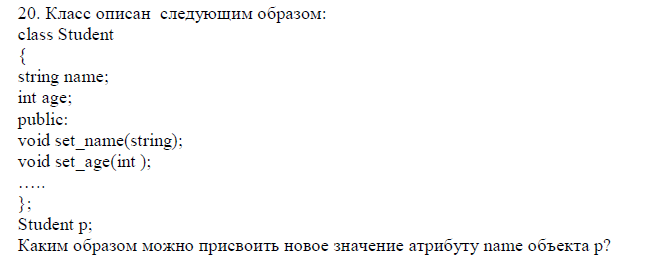
Ответ: конструктор с параметрами, копирования

\_

Ответ: конструктор с параметрами, без параметров

\_

Ответ: конструктор копирования

\_

Ответ: через сеттер