Федеральное государственное автономное образовательное учреждение высшего образования «Пермский национальный исследовательский

политехнический университет»

Лабораторная работа №2

«Классы и объекты. Использование конструкторов»

Выполнил:

студент первого курса

ЭТФ группы РИС-23-3б

Коротаев Александр Дмитриевич

Проверила:

Доцент кафедры ИТАС О. А. Полякова

Пермь, 2024

Классы и объекты. Использование конструкторов

**Цель задания**

1. Создание консольного приложения, состоящего из нескольких файлов в системе программирования Visual Studio.
2. Создание объектов с использованием конструкторов.

**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.

**Задание**

Пользовательский класс ЗАРПЛАТА ФИО – string

Оклад – double

Премия (% от оклада) – int

**Анализ задачи**

1. В функции main() создаются четыре объекта класса Pay и вызываются их методы Print().
2. В функции make\_pay() создается объект класса Pay с помощью конструктора, который принимает три аргумента: имя, оклад и премию. Эти данные вводятся пользователем с помощью функций getline() и cin.
3. В классе Salary определены три метода: Print(), set\_Name() и

set\_Salary(). Метод Print() выводит информацию об объекте Salary, а методы set\_Name() и set\_Salary() устанавливают значения соответствующих полей объекта.

1. Класс Salary используется для хранения информац о сотруднике,

включая его имя, оклад и премию. В классе Salary определены три конструктора: конструктор по умолчанию, конструктор с параметрами и конструктор копирования. Конструктор по умолчанию устанавливает значения полей объекта на определенные значения, конструктор с параметрами позволяет задать значения полей при создании объекта, а конструктор копирования создает копию существующего объекта.

1. Также в классе определены методы для извлечения и установки значений полей, а также метод Print(), который выводит информацию о сотруднике.
2. Заголовочный файл salary.h, который содержит объявление класса Salary. Класс Salary предназначен для хранения информации о сотруднике, включая его имя, оклад и премию. В классе Salary определены следующие методы:

* конструктор по умолчанию, который не принимает параметров и устанавливает значения полей объекта на определенные значения;
* конструктор с параметрами, который позволяет задать значения полей при создании объекта;
* конструктор копирования, который создает копию существующего объекта;
* деструктор, который освобождает ресурсы, связанные с объектом;
* методы для извлечения и установки значений полей;
* метод Print(), который выводит информацию о сотруднике.

**Код**

Файл ООП2.cpp

#include <iostream>

#include <string>

using namespace std;

#include "salary.h"

Salary make\_salary()

{

string name; double casing; int bonus;

cout << "Введите имя: ";

cin.seekg(cin.eof());

getline(cin, name);

cout << "Введите оклад: ";

cin >> casing;

cout << "Введите премию в %: ";

cin >> bonus;

Salary salary(name, casing, bonus);

return salary;

}

int main()

{

system("chcp 1251>null");

Salary man1;

man1.print();

Salary man2("Ктото Ктотович Ктотович", 100, 10);

man2.print();

Salary man3;

man3.setName("Этот Этович Этович");

man3.setSalary(200);

man3.setBonus(20);

cout << "ФИО: " << man3.getName() << "\nОклад: " << man3.getSalary() << "\nБонус: " << man3.getBonus() << "%\n\n";

Salary man4 = make\_salary();

man4.print();

return 0;

}

Salary::Salary()

{

cout << "Конструктор без параметров для объекта " << this << endl;

name = "никто";

casing = 0;

bonus = 0;

}

Salary::Salary(string name, double casing, int bonus)

{

cout << "Конструктор с параметрами для объекта " << this << endl;

this->name = name;

this->casing = casing;

this->bonus = bonus;

}

Salary::Salary(const Salary& other)

{

cout << "Конструктор копирования для объекта " << this << endl;

this->name = other.name;

this->casing = other.casing;

this->bonus = other.bonus;

}

Salary::~Salary()

{

cout << "Вызов деструктора для объекта " << this << endl;

}

void Salary::setName(string name)

{

this->name = name;

}

void Salary::setSalary(double casing)

{

this->casing = casing;

}

void Salary::setBonus(int bonus)

{

this->bonus = bonus;

}

string Salary::getName()

{

return this->name;

}

double Salary::getSalary()

{

return this->casing;

}

int Salary::getBonus()

{

return this->bonus;

}

void Salary::print()

{

cout << "ФИО: " << this->name << "\nОклад: " << this->casing << "\nПремия: " << this->bonus << "%\n\n";

}

Файл salary.h

#pragma once

class Salary

{

private:

string name;

double casing;

int bonus;

public:

Salary();

Salary(string, double, int);

Salary(const Salary&);

~Salary();

void setName(string);

void setSalary(double);

void setBonus(int);

string getName();

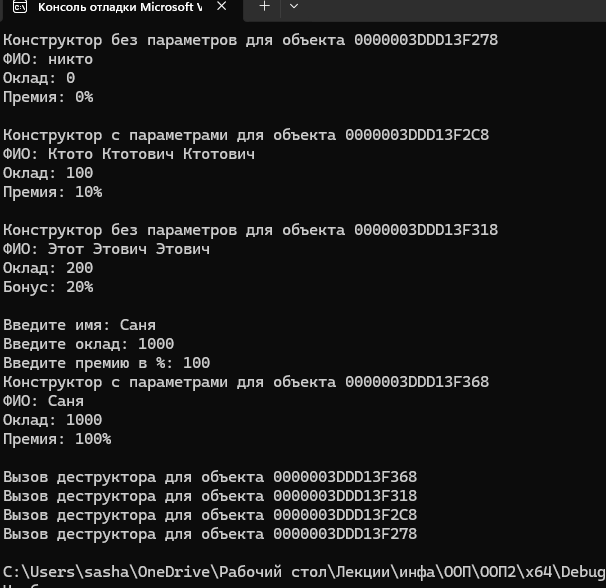
double getSalary();

int getBonus();

void print();

};

**Результат работы**



**GitHub**

[https://github.com/Korovay4ik/Laboratory-works](https://github.com/Korovay4ik/Laboratory-works%20%0c)

**Контрольные вопросы**

1. **Для чего нужен конструктор?**

Конструкторы используются для создания экземпляров класса и инициализации их данных.

1. **Сколько типов конструкторов существует в С++?**

* С++ существует несколько типов конструкторов:
  + Конструктор без параметров (или конструктор по умолчанию)
  + Конструктор с параметрами, который позволяет передавать значения при создании объекта
  + Конструктор копирования, который используется при копировании

объектов

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

Деструктор используется для освобождения ресурсов, занятых объектом, таких как память, файлы или соединения с базой данных. Если конструктор описан явно, то необходимо и деструктор описать явно.



1. **Для чего используется конструктор без параметров?**

**Конструктор с параметрами? Конструктор копирования?**

Конструктор без параметров используется для инициализации объекта с начальными значениями по умолчанию. Конструктор с параметрами позволяет задать конкретные значения при создании объекта. Конструктор копирования используется для создания копии существующего объекта.

1. **В каких случаях вызывается конструктор копирования?**

Конструктор копирования вызывается, когда необходимо создать копию объекта, например, при присваивании одного объекта другому.

1. **Перечислить свойства конструкторов.**

Свойства конструкторов:

* + Может иметь параметры для инициализации объекта
  + Может быть перегружен для поддержки различных форматов инициализации
  + Вызывается автоматически при создании объекта

1. **Перечислить свойства деструкторов.**

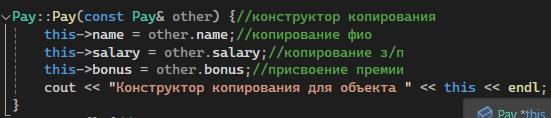
* Вызывается автоматически при уничтожении объекта
* Не может быть перегружен
* Не принимает параметров
* Не возвращает значения
* Не может быть объявлен как const , volatile или static

1. **К каким атрибутам имеют доступ методы класса?**

Private, Public, Protected

1. **Что представляет собой указатель this?**

Указатель this указывает на текущий объект, к которому обращается метод класса. Он используется внутри методов для доступа к данным объекта и вызова других методов того же класса.



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

Разница между методами, определенными внутри класса и вне класса, заключается в области видимости и доступе. Методы, определенные внутри класса, являются частью определения класса и могут быть вызваны только через объекты этого класса; можно не передать параметры. Методы, определенные вне класса, могут быть статическими и не требуют создания экземпляра класса для их вызова; передача параметров обязательна.

1. **Какое значение возвращает конструктор?**

Конструктор не возвращает значение, так как его цель - создание объекта, а не возврат результата.

1. **Какие методы создаются по умолчанию?**

Конструктор и деструктор.

1. **Какое значение возвращает деструктор?**

Деструктор не возвращает значение, так как его цель - освободить ресурсы, связанные с объектом, а не возвращать результат.

1. **Дано описание класса class Student**

**{**

**string name; int group; public:**

**student(string, int); student(const student&)**

**~student();**

**};**

**Какой метод отсутствует в описании класса?**

Конструктор по умолчанию.

1. **Какой метод будет вызван при выполнении следующих операторов: student\*s;**

**s=new student;**

Конструктор по умолчанию.

1. **Какой метод будет вызван при выполнении следующих операторов: student s(“Ivanov”,20);**

Конструктор с параметрами.

1. **Какие методы будут вызваны при выполнении следующих операторов: student s1(“Ivanov”,20);**

**student s2=s1;**

Конструктор с параметрами для первого объекта и конструктор копирования для второго.

1. **Какие методы будут вызваны при выполнении следующих операторов: student s1(“Ivanov”,20);**

**student s2; s2=s1;**

Конструктор с параметрами для первого объекта, конструктор без

параметров для второго объекта, конструктор копирования для второго

объекта.

1. **Какой конструктор будет использоваться при передаче параметра в функцию print(): void print(student a)**

**{a.show();}**

Конструктор по умолчанию.

1. **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?**

p.set\_name(“name”);