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
Акбашева Софья Руслановна

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 Pay определены три метода: Print(), set_Name() и set_Salary(). Метод Print() выводит информацию об объекте Pay, а методы set_Name() и set_Salary() устанавливают значения соответствующих полей объекта.

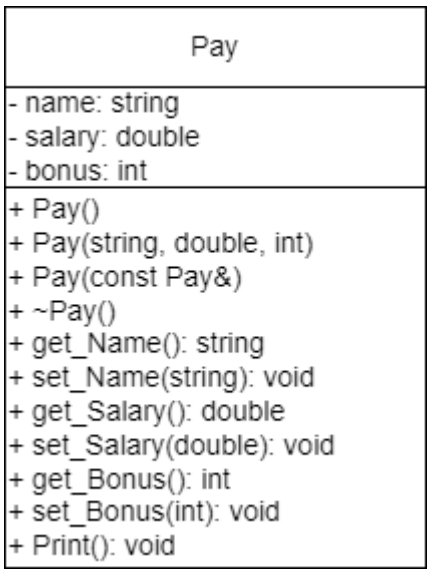
4) Класс Pay используется для хранения информации о сотруднике, включая его имя, оклад и премию. В классе Pay определены три конструктора: конструктор по умолчанию, конструктор с параметрами и конструктор копирования. Конструктор по умолчанию устанавливает значения полей объекта на определенные значения, конструктор с параметрами позволяет задать значения полей при создании объекта, а конструктор копирования создает копию существующего объекта.

5) Также в классе определены методы для извлечения и установки значений полей, а также метод Print(), который выводит информацию о сотруднике.

6) Заголовочный файл pay.h, который содержит объявление класса Pay. Класс Pay предназначен для хранения информации о сотруднике, включая его имя, оклад и премию. В классе Pay 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.

Блок схема



Код

Файл Class_2.cpp

```
#include <iostream>
#include <string>
#include "pay.h"

using namespace std;

Pay make_pay() {
	string name;
	double salary;
	int bonus;

	cout << endl << "Введите имя: ";
	getline(cin, name);
	cout << "Введите оклад: ";
	cin >> salary;
	cout << "Введите премию в % от оклада: ";
	cin >> bonus;
	Pay pay(name, salary, bonus);
	return pay;
}

int main() {
	system("chcp 1251 > Null");

	Pay p1;
	p1.Print();

	Pay p2("Дурбажева Мария Михайловна", 100000, 2);
	p2.Print();

	Pay p3;
	p3.set_Name("Петров Петр Петрович");
	p3.set_Salary(30000);
	p3.set_Bonus(200);
	p3.Print();

	Pay p4 = make_pay();
	p4.Print();
```

```
	cout << endl;
	return 0;
}
```

Файл pay.cpp

```
#include "Pay.h"
#include <iostream>
#include <string>
using namespace std;

Pay::Pay() {//конструктор по умолчанию
	name = "John Doe";
	salary = 10;
	bonus = 0;
	cout << "Конструктор по умолчанию для объекта " << this << endl;
}

Pay::Pay(string name = "Jhon", double salary = 0, int bonus = 0) { //конструктор с параметрами
	this->name = name;//присвоение фио
	this->salary = salary;//присвоение з/п
	this->bonus = bonus;//присвоение премии
	cout << "Конструктор с параметрами для объекта " << this << endl;
}

Pay::Pay(const Pay& other) {//конструктор копирования
	this->name = other.name;//копирование фио
	this->salary = other.salary;//копирование з/п
	this->bonus = other.bonus;//присвоение премии
	cout << "Конструктор копирования для объекта " << this << endl;
}
Pay::~Pay() {//деструктор
	cout << "Деструктор для объекта " << this << endl;
}

string Pay::get_Name() {//извлечение фио
	return this->name;
}
void Pay::set_Name(string name) {//присвоение фио
	this->name = name;
}
double Pay::get_Salary() {//извлечение з/п
	return this->salary;
}
void Pay::set_Salary(double salary) {//присвоение з/п
	this->salary = salary;
}
int Pay::get_Bonus() {//извлечение премии
	return this->bonus;
}
void Pay::set_Bonus(int bonus) {//присвоение премии
	this->bonus = bonus;
}
void Pay::Print() {//вывод параметров класса Pay
	cout << "ФИО: " << this->name << endl << "ОКЛАД: " << this->salary << endl << "ПРЕМИЯ: " << this->bonus << "
%" << endl << endl;
}
```

Файл pay.h

```
#pragma once //предотвращает повторную загрузку заголовочного файла, если он уже был включен
#include <iostream> //стандартные потоки ввода и вывода
using namespace std;
```

```
class Pay { //объявление класса Pay
	string name;//фио
	double salary;//з/п
	int bonus;//премия
public:
	Pay(); //конструктор класса Pay по умолчанию
	Pay(string, double, int); //конструктор класса Pay с параметрами
	Pay(const Pay&);//конструктор копирования класса Pay
	~Pay();//деструктор класса Pay

	string get_Name();//извлечение фио
	void set_Name(string);//присвоение фио
	double get_Salary();//извлечение з/п
	void set_Salary(double);//присвоение з/п
	int get_Bonus();//извлечение премии
	void set_Bonus(int);//присвоение премии
	void Print(); //вывод параметров класса
};
```

Результат работы

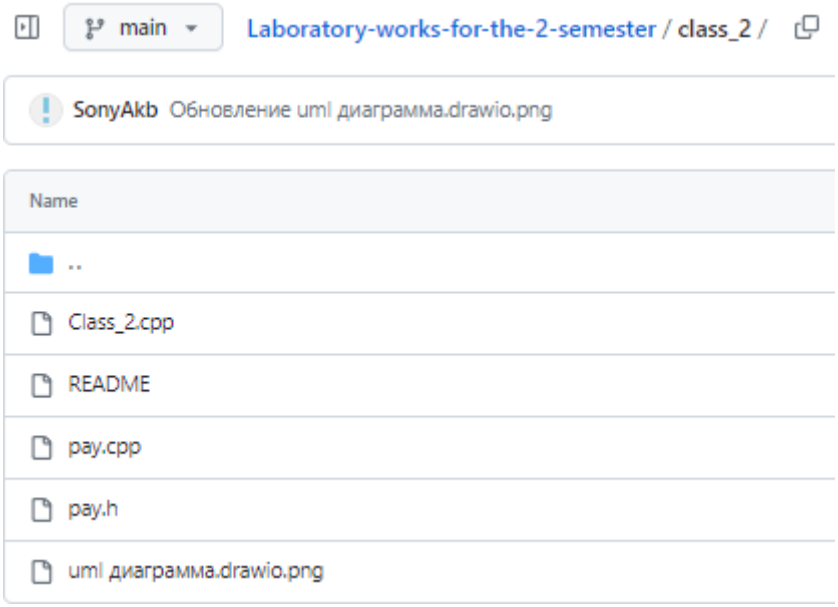
```
Конструктор по умолчанию для объекта 000000C670AFF458
ФИО: John Doe
ОКЛАД: 10
ПРЕМИЯ: 0 %

Конструктор с параметрами для объекта 000000C670AFF4A8
ФИО: Дурбажева Мария Михайловна
ОКЛАД: 100000
ПРЕМИЯ: 2 %

Конструктор по умолчанию для объекта 000000C670AFF4F8
ФИО: Петров Петр Петрович
ОКЛАД: 30000
ПРЕМИЯ: 200 %


Введите имя: Ольга Федорова
Введите оклад: 100000
Введите премию в % от оклада: 41
Конструктор с параметрами для объекта 000000C670AFF548
ФИО: Ольга Федорова
ОКЛАД: 100000
ПРЕМИЯ: 41 %


Деструктор для объекта 000000C670AFF548
Деструктор для объекта 000000C670AFF4F8
Деструктор для объекта 000000C670AFF4A8
Деструктор для объекта 000000C670AFF458
```

Вывод

В ходе работы я применила знания о работе с классами. В частности, об использовании перегрузки функций при работе с классами. По итогу работы была реализованная поставленная задача.

GitHub

Ссылка: https://github.com/SonyAkb/Laboratory-works-for-the-2-semester/tree/main/class_2



Контрольные вопросы

1. Для чего нужен конструктор?

Конструкторы используются для создания экземпляров класса и инициализации их данных.

```
Pay::Pay() {//конструктор по умолчанию
    name = "John Doe";
    salary = 10;
    bonus = 0;
    cout << "Конструктор по умолчанию для объекта " << this << endl;
}
```

2. Сколько типов конструкторов существует в C++?

В C++ 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

3. 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.

```
Pay::~Pay() {//деструктор
    cout << "Деструктор для объекта " << this << endl;
}
```

4. 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.

5. В каких случаях вызывается конструктор копирования?

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

6. Перечислить свойства конструкторов.

Свойства конструкторов:

- Может иметь параметры для инициализации объекта

- Может быть перегружен для поддержки различных форматов инициализации

- Вызывается автоматически при создании объекта

7. Перечислить свойства деструкторов.

- Вызывается автоматически при уничтожении объекта

- Не может быть перегружен

- Не принимает параметров

- Не возвращает значения

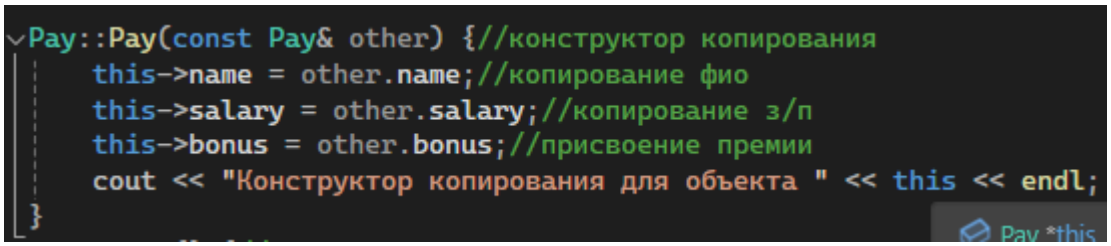
- Не может быть объявлен как const , volatile или static

8. К каким атрибутам имеют доступ методы класса?

Private, Public, Protected

9. 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.



10. 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.

11. Какое значение возвращает конструктор?

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

12. Какие методы создаются по умолчанию?

Конструктор и деструктор.

13. Какое значение возвращает деструктор?

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

14. 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.

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

s=new student;

Конструктор по умолчанию.

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

Конструктор с параметрами.

17. 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.

18. 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.

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

{a.show();}

Конструктор по умолчанию.

20. 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 p?

p.set_name(“name”);