Министерство науки и высшего образования Российской Федерации Федеральное государственное автономное учреждение высшего образования «Пермский национальный исследовательский политехнический университет»

ПНИПУ

**Лабораторная работа  
“Классы”. №1**

Выполнил:   
студент группы ИВТ-23-2Б  
Чудинов Данил Николаевич

Проверила:   
доцент кафедры ИТАС   
О.А. Полякова

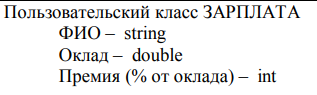
Пермь, 2024 г.

**«Классы и объекты. Использование конструкторов»**

**Анализ задачи:**

Создать класс с конструкторами (по умолчанию, с параметрами и копирования) и деструктором, создать геттеры и сеттеры.

***Вариант 15:***

****

**Код на языке C++:**

#include <iostream>

#include <string>

using namespace std;

class zarplata {

private:

string fio;

double oklad;

int bonus;

public:

zarplata() {

fio = "noname";

oklad = 0;

bonus = 0;

cout << "Вызов конструктора с параметрами: " << this->fio << " " << this->oklad << " " << this->bonus << endl;

}

zarplata(string fio, double oklad, int bonus) {

this->fio = fio;

this->oklad = oklad;

this->bonus = bonus;

cout << "Вызов конструктора с параметрами: " << this->fio << " " << this->oklad << " " << this->bonus << endl;

}

zarplata(const zarplata& d) {

fio = d.fio;

oklad = d.oklad;

bonus = d.bonus;

cout << "Вызов конструктора с параметрами: " << this->fio << " " << this->oklad << " " << this->bonus << endl;

}

void SetFIO(string name) {

fio = name;

}

void SetOKLAD(double babki) {

oklad = babki;

}

void SetBONUS(double premia) {

bonus = premia;

}

string GetFIO() {

return fio;

}

double GetOKLAD() {

return oklad;

}

int GetBONUS() {

return bonus;

}

void Print() {

cout << "ФИО: " << fio << endl << "Оклад: " << oklad << endl << "Премия: " << bonus << endl;

}

~zarplata() {

cout << "Удаление коснтруктора" << endl;

}

};

int main() {

system("chcp 1251>NULL");

zarplata h1;

zarplata h2("Глебко Георгий Лучш", 777777, 77);

zarplata h3(h2);

zarplata h4;

h4.SetFIO("Чудинов Данил Николаевич");

h4.SetOKLAD(1488);

h4.SetBONUS(48);

cout << endl;

cout << "ФИО: " << h4.GetFIO() << endl;

cout << "Оклад: " << h4.GetOKLAD() << endl;

cout << "Премия: " << h4.GetBONUS() << endl;

cout << endl;

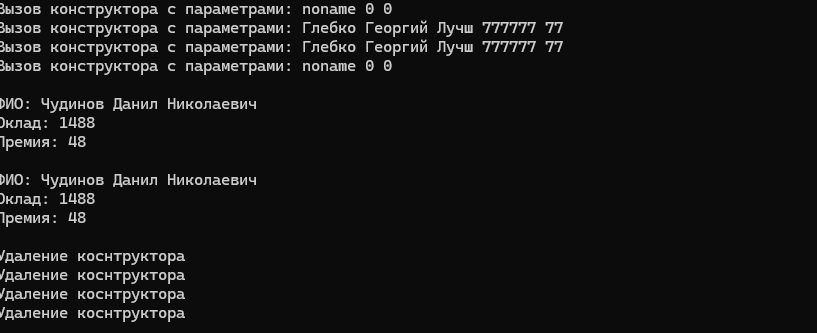
h4.Print();

cout << endl;

return 0;

}

**Работа программы:**



**Ответ на вопросы:**

1. Конструктор в языке C++ используется для инициализации объектов класса. Он вызывается при создании нового объекта и выполняет необходимые действия по инициализации полей объекта.

2. В C++ существует несколько типов конструкторов:

- Конструктор по умолчанию (без параметров)

- Конструктор с параметрами

- Конструктор копирования

- Конструктор перемещения

- Другие пользовательские конструкторы, определенные программистом.

3. Деструктор в C++ используется для освобождения ресурсов, выделенных объекту во время его жизни. Он вызывается при удалении объекта и занимается очисткой ресурсов, включая память, открытые файлы и т.д. Деструктор обычно описывается явно, если класс содержит динамически выделенные ресурсы или необходима какая-то специфическая логика при уничтожении объекта.

4. Конструктор без параметров используется для инициализации объекта, когда нет необходимости передавать какие-либо начальные значения. Конструктор с параметрами позволяет передать и установить начальные значения полей объекта непосредственно при его создании.

5. Конструктор копирования вызывается при создании нового объекта как копии существующего объекта. Это происходит, например, при передаче объекта в функцию по значению или при возврате объекта из функции.

6. Свойства конструкторов:

- Имеют имя, совпадающее с именем класса

- Не возвращают значения, в том числе и void

- Могут быть перегружены

- Имеют доступ к приватным членам класса через указатель this

7. Свойства деструкторов:

- Имеют тот же тип и имя, что и класс, но с символом ~ перед ними

- Не принимают аргументов

- Вызываются автоматически при уничтожении объекта

- Используются для освобождения ресурсов

8. Методы класса имеют доступ ко всем членам класса, включая приватные, за исключением статических методов, которые не имеют доступа к нестатическим членам класса.

9. Указатель `this` представляет собой указатель на текущий объект класса. Он используется для доступа к членам класса внутри методов.

10. Основная разница между методами, определенными внутри класса и вне класса, заключается в доступе к полям и методам класса. Методы, определенные внутри класса, имеют доступ ко всем членам класса без необходимости явного указания объекта (`this`). Методы, определенные вне класса, должны использовать объект класса для доступа к его членам.

11. Конструктор не возвращает никакого значения, он просто инициализирует объект.

12. По умолчанию, компилятор создает:

- Конструктор по умолчанию (если не определен другой конструктор)

- Конструктор копирования

- Оператор присваивания

- Деструктор

13. Деструктор не возвращает никакое значение. Его главная задача - освободить ресурсы, которые были захвачены объектом во время его жизни.

В описании класса Student отсутствует конструктор по умолчанию.

15. При выполнении оператора `student\* s; s = new student;` будет вызван конструктор по умолчанию.

16. При выполнении оператора `student s("Ivanov", 20);` будет вызван конструктор с параметрами.

17. При выполнении операторов `student s1("Ivanov", 20); student s2 = s1;` будет вызван конструктор копирования для создания копии объекта s1 в объекте s2.

18. При выполнении операторов `student s1("Ivanov", 20); student s2; s2 = s1;` будут вызваны конструктор с параметрами для создания s1 и оператор присваивания для копирования данных из s1 в s2.

19. При передаче параметра в функцию `print` будет использован конструктор копирования, так как параметр `a` передается по значению.

20. Чтобы присвоить новое значение атрибуту `name` объекта `p`, необходимо вызвать метод `set\_name(string)` для объекта `p`, передав ему новое значение атрибута `name`.

**Вывод:** Задача была выполнена. Всё получилось.