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 ПО ЛАБОРАТОРНОЙ РАБОТЕ**

Дисциплина: «Основы алгоритмизации и программирования»

Тема : "Классы и объекты. Использование конструкторов."

Выполнила работу

Студентка группы РИС-22-1Б

Верхоланцева Е. С.

Проверила

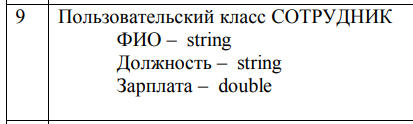
Доцент кафедры ИТАС

Полякова О.А.

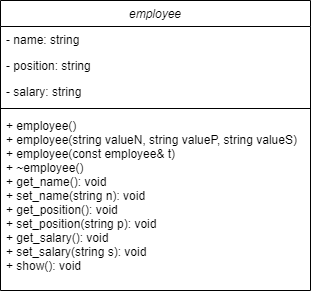
Г. Пермь-2

**Постановка задачи**

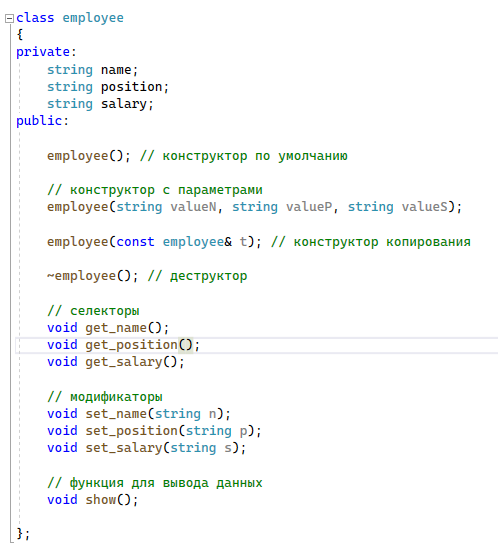
1. Определить пользовательский класс.  
2. Определить в классе следующие конструкторы: без параметров, с параметрами, копирования.  
3. Определить в классе деструктор.  
4. Определить в классе компоненты-функции для просмотра и установки полей данных (селекторы и модификаторы).  
5. Написать демонстрационную программу, в которой продемонстрировать все три случая вызова конструктора-копирования, вызов конструктора с параметрами и конструктора без параметров.



**Диаграмма класса**



**Описание класса**



Класс employee. Переменные:

name - ФИО, position - должность, salary - зарплата.

Методы:

3 вида конструкторов: по умолчанию, с параметрами, копирования.

Деструктор.

Селекторы для получения ФИО, должности и зарплаты.

Модификаторы для установки введенных значений на место ФИО, должности и зарплаты.

Метод show() для вывода всех данных класса.

**Определение компонентных функций**

employee::employee() // конструктор по умолчанию

{

cout << "Вызов конструктора по умолчанию." << endl;

}

// конструктор с параметрами

employee::employee(string valueN, string valueP, string valueS)

{

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

name = valueN;

position = valueP;

salary = valueS;

}

employee::employee(const employee& t) // конструктор копирования

{

name = t.name;

position = t.position;

salary = t.salary;

}

employee::~employee() // деструктор

{

cout << "Вызов деструктора." << endl;

}

void employee::get\_name()

{

cout << "Имя: " << name << endl;

}

void employee::set\_name(string n)

{

name = n;

}

void employee::get\_position()

{

cout << "Должность: " << position << endl;

}

void employee::set\_position(string p)

{

position = p;

}

void employee::get\_salary()

{

cout << "Зарплата: " << salary << endl;

}

void employee::set\_salary(string s)

{

salary = s;

}

void employee::show()

{

cout << "ФИО: " << name << endl;

cout << "Должность: " << position << endl;

cout << "Зарплата: " << salary << endl;

}

**Определение функции show()**

employee make\_employee()

{

string n, p, s;

cout << "Введите ФИО: "; getline(cin, n);

cout << "Введите должность: "; getline(cin, p);

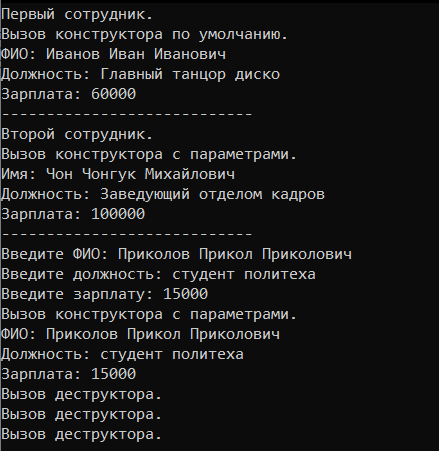
cout << "Введите зарплату: "; cin >> s;

employee emp(n,p,s);

return emp;

}

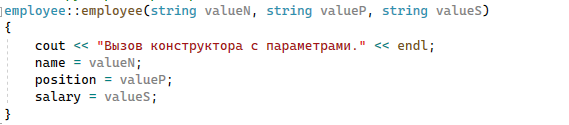
**Результаты работы программы**



**Ответы на контрольные вопросы**

1. Для чего нужен конструктор?

Конструктор необходим для инициализации полей класса.

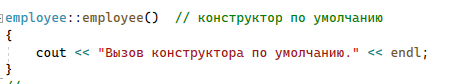


  
2. Сколько типов конструкторов существует в C++?

Существует 3 типа конструкторов: Без параметров, с параметрами, конструктор копирования.

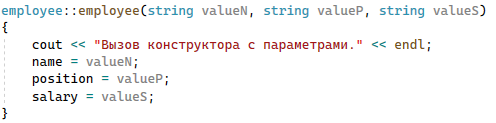
- конструктор без параметров:





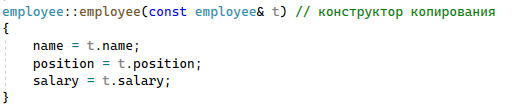
- конструктор с параметрами:



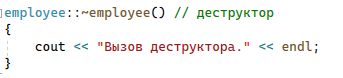


- конструктор копирования:



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

Деструктор используется для завершения работы с конструктором класса после выхода из области видимости функции. Если конструктор описан явно, то необходимо и деструктор описать явно.



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

Без параметров – для инициализации полей по умолчанию(заранее заданными значениями). С параметрами – для инициализации полей переданными в конструктор значениями. Копирования – для инициализации полей с помощью копирования их из другого обьекта.

5. В каких случаях вызывается конструктор копирования?

Конструктор копирования вызывается в случае, когда нужно заполнить n обьектов одинаковыми значениями полей.

6. Перечислить свойства конструкторов.

Инициализация полей переданными параметрами, инициализация полей по умолчанию, инициализация полей с помощью копирования, перегрузка конструкторов, конструктор списка по умолчанию, конструктор списка инициализаторов.

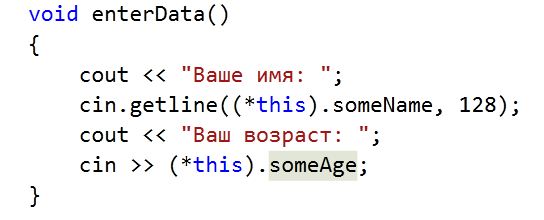
7. Перечислить свойства деструкторов.

Не могут иметь аргументов; Не возвращают значение (или void ); Не может быть объявлен как const , volatile или static . Однако они могут вызываться для уничтожения объектов, объявленных как const , volatile или static ; Может быть объявлен как virtual . Используя виртуальные деструкторы, можно уничтожать объекты, не зная их тип — правильный деструктор для объекта вызывается с помощью механизма виртуальных функций. Обратите внимание, что для абстрактных классов деструкторы также могут объявляться как чисто виртуальные функции; не описаны явно – автоматически вызовутся.

8. К каким атрибутам имеют доступ методы класса?

Private, Public, Protected;  
9. Что представляет собой указатель this?

Адрес текущего обьекта класса.



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

Внутри класса можно не передать параметры, вне – обязательно. Использование шаблонов ф-ий порой обязательно.

11. Какое значение возвращает конструктор?

Конструкторы не возвращают значений.

12. Какие методы создаются по умолчанию?

Конструктор и деструктор.

13. Какое значение возвращает деструктор?

Деструктор не возвращает значений.

14. Дано описание класса  
class Student  
{  
 string name;  
 int group;  
 public:  
 student(string, int);  
 student(const student&)  
 ~student();  
};  
Какой метод отсутствует в описании класса?  
15. Какой метод будет вызван при выполнении следующих операторов:  
student\*s;  
s=new student;

16. Какой метод будет вызван при выполнении следующих операторов: student s("Ivanov",20);  
17. Какие методы будут вызваны при выполнении следующих операторов: student s1("Ivanov",20);  
student s2=s1;  
18. Какие методы будут вызваны при выполнении следующих операторов: student s1("Ivanov",20);  
student s2;  
s2=s1;  
19. Какой конструктор будет использоваться при передаче параметра в функцию print():  
void print(student a)  
{a.show();}  
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 р?

* Аналогично, x
* Сеттер.
* Никакой.
* Конструктор с параметрами.
* Конструктор с параметрами для первого обьекта и конструктор копирования для второго.
* Аналогично предыдущему, только сначала второй обьект вызовет конструктор без параметров.
* С параметрами.
* p.set\_name(“name”);