Институт информационных технологий и управления в технических системах

Кафедра информационных технологий и компьютерных систем

ОТЧЁТ

по лабораторной работе №4

«ИССЛЕДОВАНИЕ МЕХАНИЗМА МНОЖЕСТВЕННОГО НАСЛЕДОВАНИЯ»

по дисциплине «Объектно-ориентированное программирование»

Выполнил студент группы ПИН/б-19-1-о

Мельник А.С.

Проверил ассистент

Тимофеев И.С.

Севастополь

2020

**1. ЦЕЛЬ РАБОТЫ**

Исследование основных средств описания класса, наследующего свойства нескольких базовых классов.

**2. ОПИСАНИЕ ПРОЕКТА**

Разработать программу на языке С++, в которой необходимо создать класс-наследник двух других классов. В базовых классах должны быть уникальные поля и методы.

Создать объекты базовых классов. Вывести на печать информацию о базовых классах. Создать класс-наследник, поля заполнить соответствующими полями базовых классов

Разработать тестовые примеры.

Выполнить отладку программы.

Сформулировать выводы.

Оформить отчет по проделанной работе.

**3. АНАЛИЗ ЗАДАЧИ**

Описать интерфейс ввода-вывода. Описать иерархию классов, заданную по варианту. Для каждого класса описать конструкторы и деструктор (по необходимости), функции ввода и вывода значений полей. В каждом классе должны присутствовать минимум одно уникальное поле один уникальный метод. Проиллюстрировать корректную работу механизма множественного наследования — для этого создать объекты базовых классов и заполнить их поля данными, вывести на печать. Создать объект класса-наследника, его поля заполнить значениями соответствующих полей базовых классов. Вывести на печать данные полученного объекта.

Вариант 3.  
Базовые классы: Устройство (название, мощность, производитель), Звук (частота, громкость). Класс-наследник: Сирена

**4. ТЕКСТ С++ ПРОГРАММЫ, ЗАДАННОЙ ВАРИАНТОМ ЗАДАНИЯ**

#include <iostream>

using std::cout;

using std::cin;

using std::endl;

using std::string;

class Device

{

private:

string name, manufacturer;

int power;

public:

void InputPower(int \_power) { power = \_power; };

void OutputInfo()

{

cout << "name:" << name << endl << "power:" << power << endl << "manufacturer:" << manufacturer << endl;

}

Device(string \_name, int \_power, string \_manufacturer) { name = \_name, power = \_power, manufacturer = \_manufacturer; }

~Device() {};

};

class Sound

{

private:

int frequency, volume;

public:

void InputVolume(int \_volume) { volume = \_volume; };

void OutputInfo(){ cout << "frequency:" << frequency << endl << "volume:" << volume << endl; }

Sound(int \_frequency, int \_volume) { frequency = \_frequency, volume = \_volume; }

~Sound() {};

};

class Siren : Device, Sound

{

public:

Siren(string \_name, int \_power, string \_manufacturer, int \_frequency, int \_volume) :

Device(\_name, \_power, \_manufacturer), Sound(\_frequency, \_volume) {};

};

int main()

{

setlocale(LC\_ALL, "rus");

cout << "Device:" << endl;

Device\* dev = new Device("Rock", 10, "Germany");

dev->OutputInfo();

cout << "Sound:" << endl;

Sound\* sound = new Sound(144, 8);

sound->OutputInfo();

cout << "Siren:" << endl;

Siren\* siren = new Siren("Rock", 10, "Germany", 144, 8);

((Device\*)siren)->Device::OutputInfo();

((Sound\*)siren)->Sound::OutputInfo();

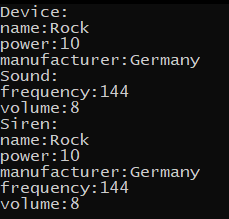
delete dev;

delete sound;

delete siren;

}

**5. СВЕДЕНИЯ ОБ ОТЛАДКЕ ПРОГРАММЫ**

****

**6. ВЫВОД**

В ходе лабораторной работы исследованы основные механизмы множественного наследования