МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Тихоокеанский государственный университет»

Кафедра «Программное обеспечение вычислительной техники и автоматизированных систем»

Реализация простейшей иерархии классов

Лабораторная работа №2

по дисциплине «Объекно-ориентированное программирование»

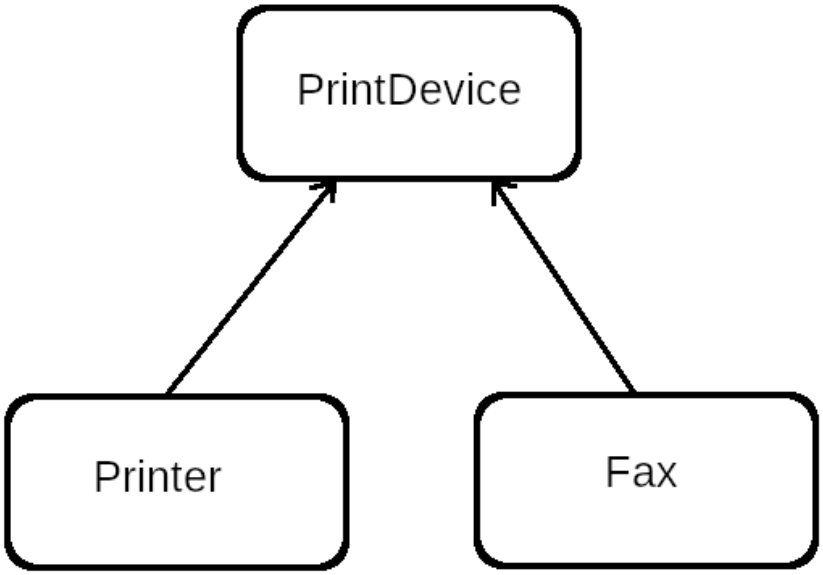
Выполнил студент Чекулаев В. Ю.

Факультет, группа ФКФН, ПО(аб)-81

Проверил Федосеев А. А.

Хабаровск – 2020г.

1. Описание иерархии классов



2. Описание классов

Имя:

PrintDevice;

Методы:

PrintDevice() - конструктор;

virtual ~PrintDevice() - деструктор;

virtual void print() const — чистый виртуальный метод;

virtual std::string get_info() const — чистый виртуальный метод;

Имя:

Printer (наследник PrintDevice)

Свойства:

std::string m_name;

Методы:

Printer(std::string name = "Printer") - конструктор;

~Printer() - деструктор;

virtual void print() const override - метод состояния печати;

virtual std::string get_info() const override - геттер метода m_name;

Имя:

Fax (наследник PrintDevice)

Свойства:

std::string m_name;

Методы:

Fax (std::string name = "Fax ") - конструктор;

~Fax () - деструктор;

virtual void print() const override - метод состояния печати;

virtual std::string get_info() const override - геттер метода m_name;

virtual std::string send_data() - метод состояния отправки информации

virtual void get_data() - метод состояния принятия информации;

3. Содержимое заголовочного файла devices.h

```
#ifndef DEVICES.H
#define DEVICES.H
#include <iostream>
#include <string>

class PrintDevise{
   public:
      PrintDevise(){}
      virtual ~PrintDevise(){}
      virtual void print() const = 0;
      virtual std::string get_info() const = 0;
};
```

```
class Printer : public PrintDevise{
    public:
        Printer(std::string name = "Printer") : PrintDevise(), m_name(name){}
        ~Printer(){}
        void print() const override { std::cout << m_name << " is printing\n"; }
        std::string get_info() const override { return m_name; }
    private:
        std::string m_name;
};
class Fax : public PrintDevise{
    public:
        Fax(std::string name = "Fax") : PrintDevise(), m_name(name){}
        ~Fax(){}
        void print() const override { std::cout << m_name << " is printing\n"; }
        std::string get_info() const override { return m_name; }
        virtual std::string send_data(){ std::cout << m_name << " is sending data\n"; }
        virtual void get_data(){ std::cout << m_name << " is getting data\n"; }
    private:
        std::string m_name;
};

#endif
```

4. Пример работы тестового приложения

Содержание файла main.cpp:

```
#include <iostream>
#include «devices.h»
int main(){
    PrintDevise* parent = new Printer("Printer 1");
    print_info(parent);
    parent→print();

    Fax fax(std::string("Fax 1"));
    delete parent;
    parent = &fax;
    std::cout << "\n";
    print_info(parent);
    parent->print();
    fax.get_data();
    fax.send_data();
    std::cout << "\n";
}
```

Вывод программы:

```
alway@alway-MS-7693:~/Документы/root/3 курс/ООР/lab2$ g++ 2.cpp -o 2
alway@alway-MS-7693:~/Документы/root/3 курс/ООР/lab2$ ./2
Printer 1
Printer 1 is printing

Fax 1
Fax 1 is printing
Fax 1 is getting data
Fax 1 is sending data

alway@alway-MS-7693:~/Документы/root/3 курс/ООР/lab2$
```