ФГБОУ ВО Хакасский государственный университет им. Н.Ф. Катанова

Инженерно-технологический институт

Кафедра программного обеспечения вычислительной техники и

автоматизированных систем

ОТЧЁТ

ПО ЛЕКЦИОННАЯ РАБОТЕ № 1

ТЕХНОЛОГИИ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

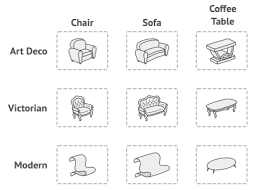
Проверил

доцент кафедры ПОВТиАС Санников Е.В.

Выполнил

студент группы 120-1 Хвостов Д.А.

Абакан 2022



#include <iostream>

#include <cmath>

using namespace std;

#pragma region Sofa

class Sofa{

public:

virtual void print() = 0;

};

class VictorianSofa: public Sofa{

public:

virtual void print() {

cout << "VictorianSofa"<< endl;

}};

class ModernSofa: public Sofa{

public:

virtual void print() {

cout << "ModernSofa"<< endl;

}};

#pragma endregion

#pragma region Armchair

class Armchair{

public:

virtual void print() = 0;

};

class VictorianArmchair: public Armchair{

public:

virtual void print() {

cout << "VictorianSofa"<< endl;

}};

class ModernArmchair: public Armchair{

public:

virtual void print() {

cout << "ModernSofa"<< endl;

}};

#pragma endregion

#pragma region Table

class Table{

public:

virtual void print() = 0;};

class VictorianTable: public Table{

public:

virtual void print() {

cout << "VictorianTable"<< endl;

}};

class ModernTable: public Table{

public:

virtual void print() {

cout << "ModernTable"<< endl;}};

#pragma endregion

class Factory{

public:

virtual Armchair\* CreateArmCheir() = 0;

virtual Sofa\* CreateSofa() = 0;

virtual Table\* CreateTable() = 0;

};

class VictorianFactory: public Factory{

public:

virtual Armchair\* CreateArmCheir(){

return new VictorianArmchair();

}

virtual Sofa\* CreateSofa(){

return new VictorianSofa();

}

virtual Table\* CreateTable(){

return new VictorianTable();

}};

class ModernFactory: public Factory{

public:

virtual Armchair\* CreateArmCheir(){

return new ModernArmchair();}

virtual Sofa\* CreateSofa(){

return new ModernSofa();}

virtual Table\* CreateTable(){

return new ModernTable();

}};

class Client{

private:

Factory\* factory = nullptr;

public:

void setFactory(Factory\* factory){

this->factory = factory; }

Factory\* getFactory(){

return factory; }

Client(Factory\* factory){

this->factory = factory;

}};

int main(){

VictorianFactory victorianFactory;

ModernFactory modernFactory;

Client client(&victorianFactory);

client.getFactory()->CreateSofa()->print();

client.getFactory()->CreateTable()->print();

}