Частное учреждение образования

«Колледж бизнеса и права»

Лабораторная работа № 17

Учащийся А.С. Парфянович

группа: Т-992

2021

**Задание 1.**

Реализовать абстрактную фабрику (тема по вариантам). Вывод интересующей пользователя информации в окне.

**Вариант 18:**

Имеется фабрика по производству кистей для художников, маляров и визажистов из конского, лосиного и собачьего волоса. Пользователь выбирает нужную ему категорию.

//Package Interfaces

//Interface PainterBrush

package Interfaces;

public interface PainterBrush {

String getPainterBrushType ();

}

//Interface DecoratorBrush

package Interfaces;

public interface DecoratorBrush {

String getDecoratorBrushType ();

}

//Interface VisagisteBrush

package Interfaces;

public interface VisagisteBrush {

String getVisagisteBrushType ();

}

//Interface IAbstractFactory

package Interfaces;

public interface IAbstractFactory {

PainterBrush getPainterBrush();

DecoratorBrush getDecoratorBrush();

VisagisteBrush getVisagisteBrush();

}

//Package HorsehairBrush

//Class HorsehairBrushAbstractFactory

package HorsehairBrush;

import Interfaces.DecoratorBrush;

import Interfaces.IAbstractFactory;

import Interfaces.PainterBrush;

import Interfaces.VisagisteBrush;

public class HorsehairBrushAbstractFactory implements IAbstractFactory {

@Override

public PainterBrush getPainterBrush() {

return new HorsehairPainterBrush();

}

@Override

public DecoratorBrush getDecoratorBrush() {

return new HorsehairDecoratorBrush();

}

@Override

public VisagisteBrush getVisagisteBrush() {

return new HorsehairVisagisteBrush();

}

}

//Class HorsehairPainterBrush

package HorsehairBrush;

import Interfaces.PainterBrush;

public class HorsehairPainterBrush implements PainterBrush {

@Override

public String getPainterBrushType() {

return "Кисточка для художника из конского волоса.";

}

}

//Class HorsehairDecoratorBrush

package HorsehairBrush;

import Interfaces.DecoratorBrush;

public class HorsehairDecoratorBrush implements DecoratorBrush {

@Override

public String getDecoratorBrushType() { return "Кисточка для маляра из конского волоса."; }

}

//Class HorsehairVisagisteBrush

package HorsehairBrush;

import Interfaces.VisagisteBrush;

public class HorsehairVisagisteBrush implements VisagisteBrush {

@Override

public String getVisagisteBrushType() { return "Кисточка для визажиста из конского волоса."; }

}

//Package MoosehairBrush

//Class MoosehairBrushAbstractFactory

package MoosehairBrush;

import Interfaces.DecoratorBrush;

import Interfaces.IAbstractFactory;

import Interfaces.PainterBrush;

import Interfaces.VisagisteBrush;

public class MoosehairBrushAbstractFactory implements IAbstractFactory {

@Override

public PainterBrush getPainterBrush() { return new MoosehairPainterBrush(); }

@Override

public DecoratorBrush getDecoratorBrush() { return new MoosehairDecoratorBrush(); }

@Override

public VisagisteBrush getVisagisteBrush() { return new MoosehairVisagisteBrush(); }

}

//Class MoosehairPainterBrush

package MoosehairBrush;

import Interfaces.PainterBrush;

public class MoosehairPainterBrush implements PainterBrush {

@Override

public String getPainterBrushType() { return "Кисточка для художника из лосиного волоса."; }

}

//Class MoosehairDecoratorBrush

package MoosehairBrush;

import Interfaces.DecoratorBrush;

public class MoosehairDecoratorBrush implements DecoratorBrush {

@Override

public String getDecoratorBrushType() { return "Кисточка для маляра из лосиного волоса."; }

}

//Class MoosehairVisagisteBrush

package MoosehairBrush;

import Interfaces.VisagisteBrush;

public class MoosehairVisagisteBrush implements VisagisteBrush {

@Override

public String getVisagisteBrushType() { return "Кисточка для визажиста из лосиного волоса."; }

}

//Package DoghairBrush

//Class DoghairBrushAbstarctFactory

package DoghairBrush;

import Interfaces.DecoratorBrush;

import Interfaces.IAbstractFactory;

import Interfaces.PainterBrush;

import Interfaces.VisagisteBrush;

public class DoghairBrushAbstractFactory implements IAbstractFactory {

@Override

public PainterBrush getPainterBrush() { return new DoghairPainterBrush(); }

@Override

public DecoratorBrush getDecoratorBrush() { return new DoghairDecoratorBrush(); }

@Override

public VisagisteBrush getVisagisteBrush() { return new DoghairVisagisteBrush(); }

}

//Class DoghairPainterBrush

package DoghairBrush;

import Interfaces.PainterBrush;

public class DoghairPainterBrush implements PainterBrush {

@Override

public String getPainterBrushType() { return "Кисточка для художника из собачего волоса."; }

}

//Class DoghairDecoratorBrush

package DoghairBrush;

import Interfaces.DecoratorBrush;

public class DoghairDecoratorBrush implements DecoratorBrush {

@Override

public String getDecoratorBrushType() { return "Кисточка для маляра из собачего волоса."; }

}

//Class DoghairVisagisteBrush

package DoghairBrush;

import Interfaces.VisagisteBrush;

public class DoghairVisagisteBrush implements VisagisteBrush {

@Override

public String getVisagisteBrushType() { return "Кисточка для визажиста из собачего волоса."; }

}

//BrashesAbstractFactory

import DoghairBrush.DoghairBrushAbstractFactory;

import HorsehairBrush.HorsehairBrushAbstractFactory;

import Interfaces.\*;

import MoosehairBrush.MoosehairBrushAbstractFactory;

public class BrashesAbstractFactory {

public static void main(String[] args) {

String brushType = "horse";

IAbstractFactory factory = null;

if (brushType.equals("horse")) {

factory = new HorsehairBrushAbstractFactory();

}

else if (brushType.equals("moose")) {

factory = new MoosehairBrushAbstractFactory();

}

else if (brushType.equals("dog")) {

factory = new DoghairBrushAbstractFactory();

}

else {

System.out.println("Тип фабрики не найден!");

}

PainterBrush painterBrush = factory.getPainterBrush();

System.out.println(painterBrush.getPainterBrushType());//поменять кодировку на windows-1251

}

}

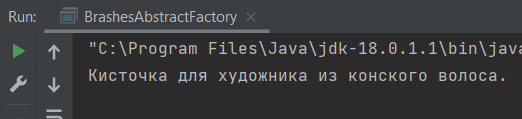


Рисунок 1