|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ **ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ**

КАФЕДРА **КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.04.01 Информатика и вычислительная техника**

**Отчет по лабораторной работе № 4**

***по дисциплине «Языки программирования для работы с большими данными»***

Студент ИУ6-21М **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Е. А. Щербакова**

(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ П. В. Степанов**

(Подпись, дата) (И.О.Фамилия)

*2020 г.*

**Задание 1:**

**Вариант 4:** Создать класс Художественная Выставка с внутренним классом, с помощью объектов которого можно хранить информацию о картинах, авторах и времени проведения выставок.

**Код программы**

**Файл Exhibition.java**

package Exhibition;

import java.util.ArrayList;

import java.util.Date;

public class Exhibition {

class Author {

public String name;

public String surname;

public Author(String name, String surname) {

this.name = name;

this.surname = surname;

}

}

class Painting {

public int width;

public int height;

public Author author;

public Painting(int width, int height, Author author) {

this.width = width;

this.height = height;

this.author = author;

}

}

public Date date;

public ArrayList<Painting> paintings = new ArrayList<Painting>();

public Exhibition(Date date) {

this.date = date;

}

public void addPainting(Painting painting) {

this.paintings.add(painting);

}

}

**Файл Main.java**

package Exhibition;

import java.util.Date;

import Exhibition.Exhibition.Author;

import Exhibition.Exhibition.Painting;

public class Main {

public static void main(String[] args) {

Exhibition exhibition = new Exhibition(new Date());

Author author = exhibition.new Author("Иван", "Иванов");

Painting painting = exhibition.new Painting(100, 200, author);

exhibition.addPainting(painting);

}

}

**Задание 2**

**Вариант 3:** Реализовать абстрактные классы или интерфейсы, а также наследование и полиморфизм для следующих классов:

interface Сотрудник <- class Инженер <- class Руководитель

**Код программы:**

**Файл Main.java**

package Worker;

public class Main {

public static void main(String[] args) {

Leed leed = new Leed("Иван", "Иванов");

Engineer worker = new Engineer("Петр", "Петров");

leed.work();

worker.work();

leed.letGo(worker);

}

}

**Файл Leed.java**package Worker;

public class Leed extends Engineer {

public Leed(String name, String company) {

super(name, company);

}

public void letGo(Engineer worker) {

worker.goToWeekend();

}

}

**Файл Engineer.java**

}

package Worker;

public class Engineer implements Worker {

public String name;

public String company;

public boolean isWorkNow = true;

public Engineer(String name, String company) {

this.name = name;

this.company = company;

}

public void work() {

this.isWorkNow = true;

}

public void goToWeekend() {

this.isWorkNow = false;

}

}

**Файл Worker.java**

package Worker;

public interface Worker {

void work();

void goToWeekend();

}

**Вывод программы**

Work!

Work!

Go!

Weekend!

**Вывод:** Программы работают корректно.