|  |  |
| --- | --- |
| 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 |

**Название:** Внутренние классы. Интерфейсы

**Дисциплина:** Языки программирования для работы с большими данными

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ6-21М |  |  | Ю.А. Вишневская |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | П. В. Степанов |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2020

**Вариант 1**

9. Создать класс Park (парк) с внутренним классом, с помощью объектов которого можно хранить информацию об аттракционах, времени их работы и стоимости.

Park

|  |
| --- |
| public class Park {  int square;  String name;  public Park(int square, String name) {  this.square = square;  this.name = name;  }  public int getSquare() {  return square;  }  public String getName() {  return name;  }  public void setSquare(int square) {  this.square = square;  }  public void setName(String name) {  this.name = name;  }  public void displayName() {  System.out.println("Название парка: " + this.name);  }  public void displaySquare() {  System.out.println("Площадь парка: " + this.square);  }  public class Attraction {  String name\_attraction;  int ticket;  String working\_time;  public Attraction(String name\_attraction, int ticket, String working\_time) {  this.name\_attraction = name\_attraction;  this.ticket = ticket;  this.working\_time = working\_time;  }  public void displayNameAttraction() {  System.out.println("Название аттракциона: " + this.name\_attraction);  }  public void displayWorkTime() {  System.out.println("Время работы аттракциона: " + this.working\_time);  }  public void displayTicket() {  System.out.println("Стоимость билета на аттракцион: " + this.ticket);  }  }  } |

|  |
| --- |
| public class LR4\_V1\_9 {  public static void main(String[] args) {  Park disneyLand = new Park(5000, "Disney Land");  Park.Attraction rollerCoaster = disneyLand.new Attraction("Американские горки", 250, "10:00-20:00");  Park.Attraction carousel = disneyLand.new Attraction("Карусель", 100, "09:00-21:00");  disneyLand.displayName();  disneyLand.displaySquare();  rollerCoaster.displayNameAttraction();  rollerCoaster.displayTicket();  rollerCoaster.displayWorkTime();  carousel.displayNameAttraction();  carousel.displayTicket();  carousel.displayWorkTime();  }  } |

**Вариант 2**

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

9. interface Мебель <- abstract class Шкаф <- class Книжный Шкаф.

Furniture

|  |
| --- |
| public interface Furniture {  void toMove();  void toPaint();  } |

Cupboard

|  |
| --- |
| import java.util.Scanner;  public abstract class Cupboard implements Furniture {  String modelCupboard;  int hightCupboard;  public Cupboard(String modelCupboard, int hightCupboard) {  this.modelCupboard = modelCupboard;  this.hightCupboard = hightCupboard;  }  public String getModelCupboard() {  return modelCupboard;  }  public int getHightCupboard() {  return hightCupboard;  }  public void setModelCupboard(String modelCupboard) {  this.modelCupboard = modelCupboard;  }  public void setHightCupboard(int hightCupboard) {  this.hightCupboard = hightCupboard;  }  abstract void openDoor();  public void toMove(){  Scanner in = new Scanner(System.in);  System.out.print("Введите расстояние в см: ");  int distance = in.nextInt();  System.out.print("Введите направление: ");  Scanner in2 = new Scanner(System.in);  String direction = in2.nextLine();  System.out.printf("Шкаф передвинут на %d см в %s \n", distance, direction);  };  public abstract void toPaint();  } |

BookCase

|  |
| --- |
| import java.util.Scanner;  public class BookCase extends Cupboard{  public BookCase(String modelCupboard, int hightCupboard) {  super(modelCupboard, hightCupboard);  }  @Override  void openDoor() {  System.out.println("Дверца открыта! Можете взять книгу!");  }  @Override  public void toPaint() {  Scanner in = new Scanner(System.in);  System.out.print("Введите цвет: ");  String color = in.nextLine();  System.out.println("Шкаф перекрашен в " + color);  }  } |

|  |
| --- |
| public class LR4\_V2\_9 {  public static void main(String[] args) {  BookCase bookCase1 = new BookCase("Мечта", 210);  System.out.println("Модель: " + bookCase1.modelCupboard);  System.out.println("Высота: " + bookCase1.hightCupboard);  bookCase1.toMove();  bookCase1.toPaint();  bookCase1.openDoor();  }  } |