|  |  |
| --- | --- |
| **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\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.04.01 Информатика и вычислительная техника**

МАГИСТЕРСКАЯ ПРОГРАММА **09.04.01/07 Интеллектуальные системы анализа,**

**обработки и интерпретации больших данных.**

**Отчет**

|  |  |
| --- | --- |
| **по лабораторной работе №** | 4 |

**Название:**

Внутренние классы. Интерфейсы.

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

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ6-22М |  |  | Е.Е. Шморгун |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | П.В. Степанов |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2023

**Вариант 1 Задача 9.** Создать класс Park (парк) с внутренним классом, с помощью объектов которого можно хранить информацию об аттракционах, времени их работы и стоимости.

Main.java

import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;

import java.util.Date;

public class Main {

public static void main(String[] args) {

List<HashMap<String, Object>> attractions = new ArrayList<>();

HashMap<String, Object> attraction = new HashMap<>();

attraction.put("name", "name1");

attraction.put("price", 10.0);

attraction.put("beginDate", new Date(2023, 12, 12, 12, 0, 0));

attraction.put("endDate", new Date(2023, 12, 12, 18, 0, 0));

attractions.add(attraction);

attraction = new HashMap<>();

attraction.put("name", "name2");

attraction.put("price", 15.0);

attraction.put("beginDate", new Date(2023, 12, 12, 13, 0, 0));

attraction.put("endDate", new Date(2023, 12, 12, 17, 0, 0));

attractions.add(attraction);

Park park = new Park(attractions);

}

}

Park.java

import java.util.Date;

import java.util.HashMap;

import java.util.List;

import java.util.stream.Collectors;

public class Park {

private List<Attraction> attractions;

private class Attraction {

private String name;

private double price;

private Date beginTime;

private Date endTime;

public Attraction(String name, double price, Date beginTime, Date endTime) {

this.name = name;

this.price = price;

}

}

public Park(List<HashMap<String, Object>> attractions) {

this.attractions = attractions

.stream()

.map(attraction ->

new Attraction(

attraction.get("name").toString(),

(double) attraction.get("price"),

(Date) attraction.get("beginTime"),

(Date) attraction.get("endTime")

)

)

.collect(Collectors.toList());

}

}

**Вариант 1 Задача 10.** Создать класс Cinema (кино) с внутренним классом, с помощью объектов которого можно хранить информацию об адресах кинотеатров, фильмах и времени сеансов.

Main.java

import java.util.\*;

public class Main {

public static void main(String[] args) {

List<HashMap<String, Object>> movieTheaters = new ArrayList<>();

HashMap<String, Object> theater = new HashMap<>();

HashMap<String, List<Date>> sessions = new HashMap<>();

List<Date> avatarDates = new ArrayList<>();

avatarDates.add(new Date(2023, 3, 12, 12, 0, 0));

avatarDates.add(new Date(2023, 4, 12, 12, 0, 0));

sessions.put("Avatar", avatarDates);

theater.put("address", "some\_address");

theater.put("movieSessions", sessions);

movieTheaters.add(theater);

Cinema cinema = new Cinema(movieTheaters);

}

}

Cinema.java

import java.util.Date;

import java.util.HashMap;

import java.util.List;

import java.util.stream.Collectors;

public class Cinema {

private List<MovieTheater> movieTheaters;

private class MovieTheater {

private String address;

private HashMap<String, List<Date>> movieSessions;

public MovieTheater(String address, HashMap<String, List<Date>> movieSessions) {

this.address = address;

this.movieSessions = movieSessions;

}

}

public Cinema(List<HashMap<String, Object>> movieTheaters) {

this.movieTheaters = movieTheaters

.stream()

.map(theater -> new MovieTheater(

theater.get("address").toString(),

(HashMap<String, List<Date>>) theater.get("movieSessions")

))

.collect(Collectors.toList());

}

}

**Вариант 2 Задача 9.** Реализовать абстрактные классы или интерфейсы, а также наследование и полиморфизм для следующих классов interface Мебель <- abstract class Шкаф <- class Книжный Шкаф.

Main.java

public class Main {

public static void main(String[] args) {

BookCabinet cabinet = new BookCabinet(10, 10, 10);

cabinet.setUp();

}

}

Furniture.java

public interface Furniture {

public void setUp();

}

Cabinet.java

public abstract class Cabinet {

protected int height;

protected int width;

protected int length;

}

BookCabinet.java

public class BookCabinet extends Cabinet implements Furniture{

public BookCabinet(int height, int width, int length) {

this.height = height;

this.width = width;

this.length = length;

}

@Override

public void setUp() {

System.out.printf("Set up book cabinet with size: %s x %s x %s\n", height, width, length);

}

}

**Вариант 2 Задача 10.** Реализовать абстрактные классы или интерфейсы, а также наследование и полиморфизм для следующих классов interface Фильм <- class Отечественный Фильм <- class Комедия.

Main.java

public class Main {

public static void main(String[] args) {

DomesticMovie movie = new DomesticMovie("Movie", 120);

Comedy comedy = new Comedy("Comedy", 110);

movie.show();

comedy.show();

}

}

Movie.java

public interface Movie {

public void show();

}

Comedy.java

public class Comedy extends DomesticMovie {

public Comedy(String name, int duration) {

super(name, duration);

}

public void show() {

System.out.printf("Showing domestic comedy %s for %s minutes\n", name, duration);

}

}

DomesticMovie.java

public class DomesticMovie implements Movie {

protected String name;

protected int duration;

public DomesticMovie() {

this.name = "";

this.duration = 0;

}

public DomesticMovie(String name, int duration) {

this.name = name;

this.duration = duration;

}

public void show() {

System.out.printf("Showing domestic movie %s for %s minutes\n", name, duration);

}

}

**Вывод:** В данной работе я познакомился с интерфейсами и как их реализовывать внутри классов.