|  |  |
| --- | --- |
| 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**

МАГИСТЕРСКАЯ ПРОГРАММА **09.04.01/12 Интеллектуальный анализ больших**

**данных в системах поддержки принятия решений.**

**Отчет**

**По лабораторной работе №3**

**Название:** Классы, наследование, полиморфизм

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

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ6-23М |  |  | В.Д. Рожнов |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | П.В. Степанов |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2023

**Задание 1:** определить класс Матрица размерности (m x n). Класс должен содержать несколько конструкторов. Объявить массив объектов. Передать объекты в метод, меняющий местами строки с максимальным и минимальным элементами k-го столбца. Создать метод, который изменяет i-ю матрицу путем возведения ее в квадрат.

Решение задания представлено в листингах 1-2.

Листинг 1 – Программный код модуля Main.java для задания 1

import java.util.Scanner;

public class Main {

public static void square(int n, Matrix[] mas){

for (int i = 0; i < mas[n].rowNum; i++){

for (int j = 0; j < mas[n].colNum; j++){

mas[n].matrix[i][j] \*= mas[n].matrix[i][j];

}

}

}

public static void replace(int k, Matrix matrix){

int maxRow = matrix.matrix[0][k];

int minRow = matrix.matrix[0][k];

int maxI = 0;

int minI = 0;

for (int i = 0; i < matrix.rowNum; i++){

if (matrix.matrix[i][k] > maxRow){

maxRow = matrix.matrix[i][k];

maxI = i;

}

if (matrix.matrix[i][k] < minRow){

minRow = matrix.matrix[i][k];

minI = i;

}

}

for (int j = 0; j < matrix.colNum; j++) {

int temp = matrix.matrix[maxI][j];

matrix.matrix[maxI][j] = matrix.matrix[minI][j];

matrix.matrix[minI][j] = temp;

}

}

public static void main(String[] args) {

Scanner in = new Scanner(System.in);

System.out.print("Input a number of objects: ");

int n = in.nextInt();

Matrix[] matrixMas = new Matrix[n];

for(int i = 0; i < n; i++){

System.out.print("Input a number of rows: ");

int rows = in.nextInt();

System.out.print("Input a number of columns: ");

int columns = in.nextInt();

matrixMas[i] = new Matrix(columns, rows);

System.out.println(i + 1 + " matrix is: ");

matrixMas[i].output();

}

square(0, matrixMas);

System.out.println("Squared matrix: ");

matrixMas[0].output();

replace(1, matrixMas[1]);

System.out.println("Replaced matrix: ");

matrixMas[1].output();

}

}

Листинг 2 – Программный код модуля Matrix.java для задания 1

import java.util.concurrent.ThreadLocalRandom;

public class Matrix{

int colNum;

int rowNum;

int[][] matrix;

public void output(){

for (int i = 0; i < rowNum; i++){

for (int j = 0; j < colNum; j++){

System.out.print(matrix[i][j] + " ");

}

System.out.println();

}

}

private void init() {

for (int i = 0; i < rowNum; i++) {

for (int j = 0; j < colNum; j++) {

matrix[i][j] = ThreadLocalRandom.current().nextInt(50 \* (-1), 50 + 1);

}

}

}

public Matrix(int col, int row){

colNum = col;

rowNum = row;

matrix = new int[rowNum][colNum];

init();

}

public Matrix(int colRow) {

colNum = colRow;

rowNum = colRow;

init();

}

}

**Задание 2:** определить класс Цепная дробь.

Решение задания представлено в листингах 3-4.

Листинг 3 – Программный код модуля Main.java для задания 2

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner in = new Scanner(System.in);

System.out.print("Input x: ");

int x1 = in.nextInt();

System.out.print("Input n: ");

int n1 = in.nextInt();

int[] mas1 = new int[n1];

for (int i = 0; i < n1; i++){

System.out.print("Input a: ");

int a = in.nextInt();

mas1[i] = a;

}

ChainShot shot1 = new ChainShot(x1, n1, mas1);

System.out.println("Value of shot1 is: " + shot1.calculatedShot);

System.out.print("Input x: ");

int x2 = in.nextInt();

System.out.print("Input n: ");

int n2 = in.nextInt();

int[] mas2 = new int[n2];

for (int i = 0; i < n2; i++){

System.out.print("Input a: ");

int a = in.nextInt();

mas2[i] = a;

}

ChainShot shot2 = new ChainShot(x2, n2, mas2);

System.out.println("Value of shot2 is: " + shot2.calculatedShot);

System.out.println(shot1.addition(shot2));

}

}

Листинг 4 – Программный код модуля ChainShot.java для задания 2

public class ChainShot {

public double calculatedShot;

public ChainShot(int x, int n, int[] mas){

calculatedShot = mas[n-1];

for (int i = 0; i < n; i++){

calculatedShot = x / calculatedShot;

calculatedShot = calculatedShot + mas[n - i - 1];

}

}

public double addition(ChainShot shot){

return calculatedShot + shot.calculatedShot;

}

public double subtraction(ChainShot shot){

return calculatedShot - shot.calculatedShot;

}

public double division(ChainShot shot){

return calculatedShot / shot.calculatedShot;

}

public double multiplication(ChainShot shot){

return calculatedShot \* shot.calculatedShot;

}

}

**Задание 3:** Создать класс. Определить конструкторы и методы setТип(), getТип(), toString(). Определить дополнительно методы в классе, создающем массив объектов. Задать критерий выбора данных и вывести эти данные на консоль.

Book: id, Название, Автор(ы), Издательство, Год издания, Количество страниц, Цена, Переплет. Создать массив объектов. Вывести: a) список книг заданного автора; b) список книг, выпущенных заданным издательством; c) список книг, выпущенных после заданного года.

Решение задания представлено в листингах 5-7.

Листинг 5 – Программный код модуля Main.java для задания 3

public class Main {

public static void main(String[] args) {

Book[] books = new Book[] {

new Book(1, "The Lord of the Rings", new String[]{"J.R.R. Tolkien"}, "HarperCollins", 1954, 1178, 25.99, "Hardcover"),

new Book(2, "The Hobbit", new String[]{"J.R.R. Tolkien"}, "Houghton Mifflin Harcourt", 1937, 310, 12.99, "Paperback"),

new Book(3, "Harry Potter and the Philosopher's Stone", new String[]{"J.K. Rowling"}, "Bloomsbury", 1997, 223, 9.99, "Paperback"),

new Book(4, "Harry Potter and the Chamber of Secrets", new String[]{"J.K. Rowling"}, "Bloomsbury", 1998, 251, 10.99, "Paperback"),

new Book(5, "Harry Potter and the Prisoner of Azkaban", new String[]{"J.K. Rowling"}, "Bloomsbury", 1999, 317, 11.99, "Paperback"),

new Book(6, "The Da Vinci Code", new String[]{"Dan Brown"}, "Doubleday", 2003, 454, 12.99, "Paperback"),

new Book(7, "Angels & Demons", new String[]{"Dan Brown"}, "Pocket Books", 2000, 736, 11.99, "Paperback"),

new Book(8, "Digital Fortress", new String[]{"Dan Brown"}, "St. Martin's Press", 1998, 384, 10.99, "Paperback"),

new Book(9, "The Lost Symbol", new String[]{"Dan Brown"}, "Doubleday", 2009, 672, 13.99, "Hardcover"),

new Book(10, "Origin", new String[]{"Dan Brown"}, "Doubleday", 2017, 480, 14.99, "Hardcover")

};

BookArray bookArray = new BookArray(books);

// список книг заданного автора

System.out.println("Books by J.K. Rowling:");

Book[] jkRowlingBooks = bookArray.getByAuthor("J.K. Rowling");

for (Book book : jkRowlingBooks) {

System.out.println(book);

}

// список книг, выпущенных заданным издательством

System.out.println("Books published by Bloomsbury:");

Book[] bloomsburyBooks = bookArray.getByPublisher("Bloomsbury");

for (Book book : bloomsburyBooks) {

System.out.println(book);

}

// список книг, выпущенных после заданного года

System.out.println("Books published after 2000:");

Book[] booksAfter2000 = bookArray.getAfterYear(2000);

for (Book book : booksAfter2000) {

System.out.println(book);

}

}

}

Листинг 6 – Программный код модуля Book.java для задания 3

import java.util.Arrays;

public class Book {

private int id;

private String title;

private String[] authors;

private String publisher;

private int year;

private int pageCount;

private double price;

private String binding;

public Book(int id, String title, String[] authors, String publisher, int year, int pageCount, double price, String binding) {

this.id = id;

this.title = title;

this.authors = authors;

this.publisher = publisher;

this.year = year;

this.pageCount = pageCount;

this.price = price;

this.binding = binding;

}

public void setId(int id) {

this.id = id;

}

public int getId() {

return id;

}

public void setTitle(String title) {

this.title = title;

}

public String getTitle() {

return title;

}

public void setAuthors(String[] authors) {

this.authors = authors;

}

public String[] getAuthors() {

return authors;

}

public void setPublisher(String publisher) {

this.publisher = publisher;

}

public String getPublisher() {

return publisher;

}

public void setYear(int year) {

this.year = year;

}

public int getYear() {

return year;

}

public void setPageCount(int pageCount) {

this.pageCount = pageCount;

}

public int getPageCount() {

return pageCount;

}

public void setPrice(double price) {

this.price = price;

}

public double getPrice() {

return price;

}

public void setBinding(String binding) {

this.binding = binding;

}

public String getBinding() {

return binding;

}

public String toString() {

return "Book{" +

"id=" + id +

", title='" + title + '\'' +

", authors=" + Arrays.toString(authors) +

", publisher='" + publisher + '\'' +

", year=" + year +

", pageCount=" + pageCount +

", price=" + price +

", binding='" + binding + '\'' +

'}';

}

}

Листинг 7 – Программный код модуля BookArray.java для задания 3

import java.util.ArrayList;

public class BookArray {

private Book[] books;

public BookArray(Book[] books) {

this.books = books;

}

public Book[] getBooks() {

return books;

}

public Book[] getByAuthor(String author) {

ArrayList<Book> result = new ArrayList<Book>();

for (Book book : books) {

for (String bookAuthor : book.getAuthors()) {

if (bookAuthor.equals(author)) {

result.add(book);

break;

}

}

}

return result.toArray(new Book[result.size()]);

}

public Book[] getByPublisher(String publisher) {

ArrayList<Book> result = new ArrayList<Book>();

for (Book book : books) {

if (book.getPublisher().equals(publisher)) {

result.add(book);

}

}

return result.toArray(new Book[result.size()]);

}

public Book[] getAfterYear(int year) {

ArrayList<Book> result = new ArrayList<Book>();

for (Book book : books) {

if (book.getYear() > year) {

result.add(book);

}

}

return result.toArray(new Book[result.size()]);

}

}

**Задание 4:** создать классы, спецификации которых приведены ниже. Определить конструкторы и методы setТип(), getТип(), toString(). Определить дополнительно методы в классе, создающем массив объектов. Задать критерий выбора данных и вывести эти данные на консоль.

House: id, Номер квартиры, Площадь, Этаж, Количество комнат, Улица, Тип здания, Срок эксплуатации. Создать массив объектов. Вывести: a) список квартир, имеющих заданное число комнат; b) список квартир, имеющих заданное число комнат и расположенных на этаже, который находится в заданном промежутке; c) список квартир, имеющих площадь, превосходящую заданную.

Решение задания представлено в листингах 8-10.

Листинг 8 – Программный код модуля Main.java для задания 4

public class Main {

public static void main(String[] args) {

House[] houses = new House[]{

new House(1, 10, 70, 5, 2, "Pushkinskaya", "brick", 50),

new House(2, 12, 80, 2, 3, "Lenina", "panel", 40),

new House(3, 15, 90, 7, 3, "Kirova", "brick", 60),

new House(4, 20, 60, 1, 2, "Sovetskaya", "panel", 35),

new House(5, 25, 100, 8, 4, "Gagarina", "brick", 70)

};

HouseArray HouseArray = new HouseArray(houses);

// список квартир, имеющих заданное число комнат

System.out.println("Houses by number of rooms = 2:");

House[] roomHouses = HouseArray.getByNumberOfRooms(2);

for (House house : roomHouses) {

System.out.println(house);

}

// список квартир, имеющих заданное число комнат и расположенных на этаже, который находится в заданном промежутке

System.out.println("Houses by rooms number and floor range:");

House[] roomAndFloorRangeHouses = HouseArray.getByNumberOfRoomsAndFloorRange(2, 5, 7);

for (House house : roomAndFloorRangeHouses) {

System.out.println(house);

}

// список квартир, имеющих площадь, превосходящую заданную

System.out.println("Houses with area grater than 80:");

House[] housesAreaGraterThan = HouseArray.getByAreaGreaterThan(80);

for (House house : housesAreaGraterThan) {

System.out.println(house);

}

}

}

Листинг 9 – Программный код модуля House.java для задания 4

public class House {

private int id;

private int apartmentNumber;

private double area;

private int floor;

private int numberOfRooms;

private String street;

private String buildingType;

private int yearsOfExploitation;

public House(int id, int apartmentNumber, double area, int floor, int numberOfRooms, String street, String buildingType, int yearsOfExploitation) {

this.id = id;

this.apartmentNumber = apartmentNumber;

this.area = area;

this.floor = floor;

this.numberOfRooms = numberOfRooms;

this.street = street;

this.buildingType = buildingType;

this.yearsOfExploitation = yearsOfExploitation;

}

public void setId(int id) {

this.id = id;

}

public int getId() {

return id;

}

public void setApartmentNumber(int apartmentNumber) {

this.apartmentNumber = apartmentNumber;

}

public int getApartmentNumber() {

return apartmentNumber;

}

public void setArea(double area) {

this.area = area;

}

public double getArea() {

return area;

}

public void setFloor(int floor) {

this.floor = floor;

}

public int getFloor() {

return floor;

}

public void setNumberOfRooms(int numberOfRooms) {

this.numberOfRooms = numberOfRooms;

}

public int getNumberOfRooms() {

return numberOfRooms;

}

public void setStreet(String street) {

this.street = street;

}

public String getStreet() {

return street;

}

public void setBuildingType(String buildingType) {

this.buildingType = buildingType;

}

public String getBuildingType() {

return buildingType;

}

public void setYearsOfExploitation(int yearsOfExploitation) {

this.yearsOfExploitation = yearsOfExploitation;

}

public int getYearsOfExploitation() {

return yearsOfExploitation;

}

public String toString() {

return "House{" +

"id=" + id +

", apartmentNumber=" + apartmentNumber +

", area=" + area +

", floor=" + floor +

", numberOfRooms=" + numberOfRooms +

", street='" + street + '\'' +

", buildingType='" + buildingType + '\'' +

", yearsOfExploitation=" + yearsOfExploitation +

'}';

}

}

Листинг 10 – Программный код модуля HouseArray.java для задания 4

import java.util.ArrayList;

import java.util.Arrays;

public class HouseArray {

private House[] houses;

public HouseArray(House[] houses) {

this.houses = houses;

}

public House[] getByNumberOfRooms(int numberOfRooms) {

ArrayList<House> filteredHouses = new ArrayList<>();

for (House house : houses) {

if (house.getNumberOfRooms() == numberOfRooms) {

filteredHouses.add(house);

}

}

return filteredHouses.toArray(new House[0]);

}

public House[] getByNumberOfRoomsAndFloorRange(int numberOfRooms, int startFloor, int endFloor) {

ArrayList<House> filteredHouses = new ArrayList<>();

for (House house : houses) {

if (house.getNumberOfRooms() == numberOfRooms && house.getFloor() >= startFloor && house.getFloor() <= endFloor) {

filteredHouses.add(house);

}

}

return filteredHouses.toArray(new House[0]);

}

public House[] getByAreaGreaterThan(double area) {

ArrayList<House> filteredHouses = new ArrayList<>();

for (House house : houses) {

if (house.getArea() > area) {

filteredHouses.add(house);

}

}

return filteredHouses.toArray(new House[0]);

}

}

**Задание 5:** создать приложение, удовлетворяющее требованиям, приведенным в задании. Аргументировать принадлежность классу каждого создаваемого метода и корректно переопределить для каждого класса методы equals(), hashCode(), toString().

Создать объект класса Роза, используя классы Лепесток, Бутон. Методы: расцвести, завять, вывести на консоль цвет бутона.

Решение задания представлено в листингах 11-13.

Листинг 11 – Программный код модуля Main.java для задания 5

public class Main {

public static void main(String[] args) {

Bud bud = new Bud("Red");

Petal petal = new Petal();

Rose rose = new Rose();

rose.setBud(bud);

rose.setPetal(petal);

rose.bloom();

rose.printBudColor();

rose.wither();

System.out.println(rose);

}

}

Листинг 12 – Программный код модуля Bud.java для задания 5

import java.util.Objects;

public class Bud {

private String color;

public Bud(String color) {

this.color = color;

}

public String getColor() {

return color;

}

// Getters and setters for color

@Override

public boolean equals(Object obj) {

if (this == obj) {

return true;

}

if (obj == null || getClass() != obj.getClass()) {

return false;

}

Bud bud = (Bud) obj;

return Objects.equals(color, bud.color);

}

@Override

public int hashCode() {

return Objects.hash(color);

}

@Override

public String toString() {

return "Bud{" +

"color='" + color + '\'' +

'}';

}

}

Листинг 13 – Программный код модуля Rose.java для задания 5

import java.util.Objects;

public class Rose {

private Petal petal;

private Bud bud;

public void bloom() {

System.out.println("The rose is blooming.");

}

public void wither() {

System.out.println("The rose is withering.");

}

public void printBudColor() {

System.out.println("Bud color: " + bud.getColor());

}

public void setPetal(Petal petal) {

this.petal = petal;

}

public void setBud(Bud bud) {

this.bud = bud;

}

// Getters and setters for petal and bud

@Override

public boolean equals(Object obj) {

if (this == obj) {

return true;

}

if (obj == null || getClass() != obj.getClass()) {

return false;

}

Rose rose = (Rose) obj;

return Objects.equals(petal, rose.petal) && Objects.equals(bud, rose.bud);

}

@Override

public int hashCode() {

return Objects.hash(petal, bud);

}

@Override

public String toString() {

return "Rose{" +

"petal=" + petal +

", bud=" + bud +

'}';

}

}

**Задание 6:** создать приложение, удовлетворяющее требованиям, приведенным в задании. Аргументировать принадлежность классу каждого создаваемого метода и корректно переопределить для каждого класса методы equals(), hashCode(), toString().

Создать объект класса Дерево, используя классы Лист. Методы: зацвести, опасть листьям, покрыться инеем, пожелтеть листьям.

Решение задания представлено в листингах 14-16.

Листинг 11 – Программный код модуля Main.java для задания 6

public class Main {

public static void main(String[] args) {

Leaf leaf = new Leaf("green");

Tree tree = new Tree(leaf);

tree.blossom();

tree.fall();

tree.coverWithHoarfrost();

tree.turnYellow();

System.out.println(tree);

}

}

Листинг 12 – Программный код модуля Leaf.java для задания 6

import java.util.Objects;

class Leaf {

private String color;

public Leaf(String color) {

this.color = color;

}

public void blossom() {

// Логика для зацветания листа

}

public void fall() {

// Логика для опадания листа

}

public void coverWithHoarfrost() {

// Логика для покрытия листа инеем

}

public void turnYellow() {

// Логика для пожелтения листа

}

// Переопределение методов equals(), hashCode() и toString()

@Override

public boolean equals(Object o) {

if (this == o) return true;

if (o == null || getClass() != o.getClass()) return false;

Leaf leaf = (Leaf) o;

return Objects.equals(color, leaf.color);

}

@Override

public int hashCode() {

return Objects.hash(color);

}

@Override

public String toString() {

return "Leaf{" +

"color='" + color + '\'' +

'}';

}

}

Листинг 13 – Программный код модуля Tree.java для задания 6

import java.util.Objects;

class Tree {

private Leaf leaf;

public Tree(Leaf leaf) {

this.leaf = leaf;

}

public void blossom() {

leaf.blossom();

}

public void fall() {

leaf.fall();

}

public void coverWithHoarfrost() {

leaf.coverWithHoarfrost();

}

public void turnYellow() {

leaf.turnYellow();

}

// Переопределение методов equals(), hashCode() и toString()

@Override

public boolean equals(Object o) {

if (this == o) return true;

if (o == null || getClass() != o.getClass()) return false;

Tree tree = (Tree) o;

return Objects.equals(leaf, tree.leaf);

}

@Override

public int hashCode() {

return Objects.hash(leaf);

}

@Override

public String toString() {

return "Tree{" +

"leaf=" + leaf +

'}';

}

}

**Задание 7:** построить модель программной системы.

Система Библиотека. Читатель оформляет Заказ на Книгу. Система осуществляет поиск в Каталоге. Библиотекарь выдает Читателю Книгу на абонемент или в читальный зал. При невозвращении Книги Читателем он может быть занесен Администратором в «черный список».

Решение задания представлено в листингах 14-19.

Листинг 14 – Программный код модуля LibrarySystem.java для задания 7

public class LibrarySystem {

public static void main(String[] args) {

Catalog catalog = new Catalog();

catalog.addBook(new Book(1, "Book 1", "Author 1"));

catalog.addBook(new Book(2, "Book 2", "Author 2"));

catalog.addBook(new Book(3, "Book 3", "Author 3"));

Reader reader = new Reader(1, "John Doe");

reader.placeOrder(catalog.searchBooks("Book 1").get(0));

Librarian librarian = new Librarian();

librarian.issueBookOnLoan(reader.getOrders().get(0), reader);

Administrator administrator = new Administrator();

administrator.addToBlacklist(reader);

}

}

Листинг 15 – Программный код модуля Book.java для задания 7

class Book {

private int id;

private String title;

private String author;

public Book(int id, String title, String author) {

this.id = id;

this.title = title;

this.author = author;

}

// Геттеры и сеттеры для полей класса

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getTitle() {

return title;

}

public void setTitle(String title) {

this.title = title;

}

public String getAuthor() {

return author;

}

public void setAuthor(String author) {

this.author = author;

}

}

Листинг 16 – Программный код модуля Administrator.java для задания 7

class Administrator {

public void addToBlacklist(Reader reader) {

// Логика добавления читателя в черный список

}

public void removeFromBlacklist(Reader reader) {

// Логика удаления читателя из черного списка

}

}

Листинг 17 – Программный код модуля Catalog.java для задания 7

import java.util.ArrayList;

import java.util.List;

class Catalog {

private List<Book> books;

public Catalog() {

books = new ArrayList<>();

}

public void addBook(Book book) {

books.add(book);

}

public List<Book> searchBooks(String keyword) {

List<Book> matchingBooks = new ArrayList<>();

for (Book book : books) {

if (book.getTitle().contains(keyword) || book.getAuthor().contains(keyword)) {

matchingBooks.add(book);

}

}

return matchingBooks;

}

}

Листинг 18 – Программный код модуля Librarian.java для задания 7

class Librarian {

public void issueBookOnLoan(Book book, Reader reader) {

// Логика выдачи книги на абонемент

}

public void issueBookForReadingRoom(Book book, Reader reader) {

// Логика выдачи книги в читальный зал

}

}

Листинг 19 – Программный код модуля Reader.java для задания 7

import java.util.ArrayList;

import java.util.List;

class Reader {

private int id;

private String name;

private List<Book> orders;

public Reader(int id, String name) {

this.id = id;

this.name = name;

orders = new ArrayList<>();

}

public void placeOrder(Book book) {

orders.add(book);

}

public List<Book> getOrders() {

return orders;

}

// Методы для получения книг на абонемент и в читальный зал

// Методы для обработки черного списка

// Геттеры и сеттеры для полей класса

}

**Задание 8:** построить модель программной системы.

Система Конструкторское бюро. Заказчик представляет Техническое Задание (ТЗ) на проектирование многоэтажного Дома. Конструктор регистрирует ТЗ, определяет стоимость проектирования и строительства, выставляет Заказчику Счет за проектирование и создает Бригаду Конструкторов для выполнения Проекта.

Решение задания представлено в листингах 20-25.

Листинг 20 – Программный код модуля Main.java для задания 8

public class Main {

public static void main(String[] args) {

DesignOffice designOffice = new DesignOffice();

TechnicalTask task1 = new TechnicalTask(1, "Проектирование многоэтажного дома");

TechnicalTask task2 = new TechnicalTask(2, "Проектирование здания офисного комплекса");

Constructor constructor1 = new Constructor(1, "Иванов");

Constructor constructor2 = new Constructor(2, "Петров");

designOffice.registerTechnicalTask(task1);

designOffice.registerTechnicalTask(task2);

designOffice.assignConstructor(constructor1);

designOffice.assignConstructor(constructor2);

designOffice.createProject(task1, constructor1);

designOffice.createProject(task2, constructor2);

}

}

Листинг 21 – Программный код модуля ConstructionTeam.java для задания 8

import java.util.ArrayList;

import java.util.List;

class ConstructionTeam {

private List<Constructor> constructors;

public ConstructionTeam() {

constructors = new ArrayList<>();

}

public void addConstructor(Constructor constructor) {

constructors.add(constructor);

}

public List<Constructor> getConstructors() {

return constructors;

}

}

Листинг 22 – Программный код модуля Constructor.java для задания 8

class Constructor {

private int constructorId;

private String name;

public Constructor(int constructorId, String name) {

this.constructorId = constructorId;

this.name = name;

}

public int getConstructorId() {

return constructorId;

}

public String getName() {

return name;

}

public void createProject(TechnicalTask task) {

Project project = new Project(task, this);

project.calculateCost();

}

}

Листинг 23 – Программный код модуля DesignOffice.java для задания 8

import java.util.ArrayList;

import java.util.List;

class DesignOffice {

private List<TechnicalTask> technicalTasks;

private List<Project> projects;

private ConstructionTeam constructionTeam;

public DesignOffice() {

technicalTasks = new ArrayList<>();

projects = new ArrayList<>();

constructionTeam = new ConstructionTeam();

}

public void registerTechnicalTask(TechnicalTask task) {

technicalTasks.add(task);

}

public void assignConstructor(Constructor constructor) {

constructionTeam.addConstructor(constructor);

}

public void createProject(TechnicalTask task, Constructor constructor) {

if (technicalTasks.contains(task) && constructionTeam.getConstructors().contains(constructor)) {

Project project = new Project(task, constructor);

projects.add(project);

} else {

System.out.println("Невозможно создать проект. ТЗ или конструктор не найдены.");

}

}

}

Листинг 24 – Программный код модуля Project.java для задания 8

class Project {

private TechnicalTask task;

private Constructor assignedConstructor;

public Project(TechnicalTask task, Constructor assignedConstructor) {

this.task = task;

this.assignedConstructor = assignedConstructor;

}

public void calculateCost() {

double cost = 10000.0; // Пример расчета стоимости проектирования и строительства

System.out.println("Стоимость проектирования и строительства: $" + cost);

}

}

Листинг 25 – Программный код модуля TechnicalTask.java для задания 8

class TechnicalTask {

private int taskId;

private String description;

public TechnicalTask(int taskId, String description) {

this.taskId = taskId;

this.description = description;

}

public int getTaskId() {

return taskId;

}

public String getDescription() {

return description;

}

}

**Вывод:** в ходе выполнения данной лабораторной работы были изучены методы работы с классами на языке Java, а также использованы приемы наследования и полиморфизма.