

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ
«БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

ФАКУЛЬТЕТ ЭЛЕКТРОННО-ИНФОРМАЦИОННЫХ СИСТЕМ

Кафедра интеллектуальных информационных технологий

Отчет по лабораторной работе №3

Специальность ПО

Выполнил
Войтюк Е.О.
студент группы ПО-8
Проверил
А. А. Крощенко,
ст. преп. кафедры ИИТ,
«__k_____» 2024 г.

Брест 2024

Цель работы: научиться создавать и использовать классы в программах на языке программирования Java

Задание 1. 6) Множество вещественных чисел ограниченной мощности – Предусмотреть возможность объединения двух множеств, вывода на печать элементов множества, а так же метод, определяющий, принадлежит ли указанное значение множеству. Класс должен содержать методы, позволяющие добавлять и удалять элемент в/из множества. Конструктор должен позволить создавать объекты с начальной инициализацией. Мощность множества задается при создании объекта. Реализацию множества осуществить на базе одномерного массива. Реализовать метод equals, выполняющий сравнение объектов данного типа.

Выполнение:

Код программы

```
import java.util.Scanner;

public class RealNumberSet {
    private double[] numbers;
    private int size;

    public RealNumberSet(int capacity) {
        this.numbers = new double[capacity];
        this.size = 0;
    }

    public boolean add(double number) {
        if (size < numbers.length) {
            if (!contains(number)) {
                numbers[size] = number;
                size++;
                return true;
            } else {
                System.out.println("Число " + number + " уже существует.");
                return false;
            }
        } else {
            return false;
        }
    }
}
```

```

public boolean remove(double number) {
    for (int i = 0; i < size; i++) {
        if (numbers[i] == number) {
            System.arraycopy(numbers, i + 1, numbers, i, size - i - 1);
            size--;
            return true;
        }
    }
    return false;
}

```

```

public boolean contains(double number) {
    for (int i = 0; i < size; i++) {
        if (numbers[i] == number) {
            return true;
        }
    }
    return false;
}

```

@Override

```

public String toString() {
    StringBuilder sb = new StringBuilder();
    for (int i = 0; i < size; i++) {
        sb.append(numbers[i]);
        if (i < size - 1) {
            sb.append(", ");
        }
    }
    return sb.toString();
}

```

@Override

```

public boolean equals(Object obj) {
    if (this == obj) {
        return true;
    }
    if (obj == null || getClass() != obj.getClass()) {
        return false;
    }
    RealNumberSet other = (RealNumberSet) obj;
    if (size != other.size) {
        return false;
    }
}

```

```
for (int i = 0; i < size; i++) {  
    if (numbers[i] != other.numbers[i]) {  
        return false;  
    }  
}  
return true;  
}
```

```
public static void main(String[] args) {  
    Scanner scanner = new Scanner(System.in);  
    System.out.println("Введите мощность множества:");  
    int capacity = scanner.nextInt();  
    RealNumberSet set = new RealNumberSet(capacity);  
  
    while (true) {  
        System.out.println("Выберите действие: 1 - добавить число, 2 - проверить число, 3 -  
удалить число, 4 - вывести множество, 5 - выйти");  
        int action = scanner.nextInt();  
  
        switch (action) {  
            case 1:  
                System.out.println("Введите число для добавления в множество:");  
                double numberToAdd = scanner.nextDouble();  
                set.add(numberToAdd);  
                break;  
            case 2:  
                System.out.println("Введите число для проверки его принадлежности множеству:");  
                double numberToCheck = scanner.nextDouble();  
                System.out.println("Принадлежит ли " + numberToCheck + " множеству? " +  
set.contains(numberToCheck));  
                break;  
            case 3:  
                System.out.println("Введите число для удаления из множества:");  
                double numberToRemove = scanner.nextDouble();  
                set.remove(numberToRemove);  
                break;  
            case 4:  
                System.out.println("Множество: " + set);  
                break;  
            case 5:  
                System.out.println("Выход из программы.");  
                return;  
            default:  
                System.out.println("Неверный ввод. Пожалуйста, введите число от 1 до 5.");  
        }  
    }  
}
```

```
        break;
    }
}
}
```

Рисунки с результатами работы программы

```
Введите мощность множества:
2
Выберите действие: 1 - добавить число, 2 - проверить число, 3 - удалить число, 4 - вывести множество, 5 - выйти
1
Введите число для добавления в множество:
5
Выберите действие: 1 - добавить число, 2 - проверить число, 3 - удалить число, 4 - вывести множество, 5 - выйти
1
Введите число для добавления в множество:
4
Выберите действие: 1 - добавить число, 2 - проверить число, 3 - удалить число, 4 - вывести множество, 5 - выйти
1
Введите число для добавления в множество:
5
Выберите действие: 1 - добавить число, 2 - проверить число, 3 - удалить число, 4 - вывести множество, 5 - выйти
4
Множество: 5.0, 4.0
Выберите действие: 1 - добавить число, 2 - проверить число, 3 - удалить число, 4 - вывести множество, 5 - выйти
3
Введите число для удаления из множества:
4
Выберите действие: 1 - добавить число, 2 - проверить число, 3 - удалить число, 4 - вывести множество, 5 - выйти
4
Множество: 5.0
Выберите действие: 1 - добавить число, 2 - проверить число, 3 - удалить число, 4 - вывести множество, 5 - выйти
5
Выход из программы.

Process finished with exit code 0
```

Задание 2.

6) Автоматизированная система аренды квартир

Составить программу, которая содержит информацию о квартирах, содержащихся в базе данных бюро обмена квартир. Сведения о каждой квартире (Room) содержат:

- количество комнат;
- общую площадь;
- этаж;
- адрес;
- цену аренды.
- сдается ли квартира.

Программа должна обеспечить:

- Формирование списков свободных занятых квартир;
- Поиск подходящего варианта (при равенстве количества комнат и этажа и различии площадей в пределах 10 кв. м.);
- Удаление квартиры из списка свободных квартир и перемещение в список сдаваемых квартир;
- Вывод полного списка.
- Список квартир, имеющих заданное число комнат;
- Список квартир, имеющих заданное число комнат и расположенных на этаже, который находится в заданном промежутке;
- Список квартир, имеющих площадь, превосходящую заданную.

Выполнение:

Код программы

Main:

```
import java.util.List;
import java.util.Scanner;

public static void main(String[] args) {
    RentalSystem rentalSystem = new RentalSystem();

    rentalSystem.loadRoomsFromFile("C:/Users/egor-/IdeaProjects/lab3task2/src/rooms.txt");

    Scanner scanner = new Scanner(System.in);

    int action = 0;
    while (action != 8) {
        System.out.println("1. Печать всех комнат:");
        System.out.println("2. Печать всех свободных комнат:");
        System.out.println("3. Печать всех сданных комнат:");
        System.out.println("4. Поиск комнат по количеству комнат и этажу:");
        System.out.println("5. Получение комнат по максимальной цене аренды:");
        System.out.println("6. Получение комнат по адресу:");
        System.out.println("7. Получение комнат по площади:");
        System.out.println("8. Выход");

        action = scanner.nextInt();
        scanner.nextLine(); // consume newline left-over

        switch (action) {
            case 1:
                System.out.println("Все комнаты:");
                rentalSystem.printAllRooms();
                break;
            case 2:
                System.out.println("Свободные комнаты:");
                List<Room> freeRooms = rentalSystem.getFreeRooms();
                for (Room room : freeRooms) {
                    System.out.println("Свободная комната: " + room.getAddress());
                }
                break;
            case 3:
                System.out.println("Сданные комнаты:");
                List<Room> rentedRooms = rentalSystem.getRentedRooms();
                for (Room room : rentedRooms) {
                    System.out.println("Сданная комната: " + room.getAddress());
                }
                break;
        }
    }
}
```

```

    }
    break;
case 4:
    System.out.println("Введите количество комнат и этаж:");
    int rooms = scanner.nextInt();
    int floor = scanner.nextInt();
    int maxFloor = scanner.nextInt();

    scanner.nextLine(); // consume newline left-over
    System.out.println("Комнаты по количеству комнат и этажу:");
    List<Room> foundRooms = rentalSystem.getRoomsByRoomsNumberAndFloor(rooms,
floor, maxFloor);
    for (Room room : foundRooms) {
        System.out.println("Найденная комната: " + room.getAddress());
    }
    break;
case 5:
    System.out.println("Введите максимальную цену аренды:");
    double maxRentPrice = scanner.nextDouble();
    scanner.nextLine(); // consume newline left-over
    System.out.println("Комнаты по максимальной цене аренды:");
    List<Room> roomsByMaxRentPrice =
rentalSystem.getRoomsByMaxRentPrice(maxRentPrice);
    for (Room room : roomsByMaxRentPrice) {
        System.out.println("Комната по максимальной цене аренды: " + room.getAddress());
    }
    break;
case 6:
    System.out.println("Введите адрес:");
    String address = scanner.nextLine();
    System.out.println("Комнаты по адресу:");
    List<Room> roomsByAddress = rentalSystem.getRoomsByAddress(address);
    for (Room room : roomsByAddress) {
        System.out.println("Комната по адресу: " + room.getAddress());
    }
    break;
case 7:
    System.out.println("Введите площадь:");
    double area = scanner.nextDouble();
    scanner.nextLine(); // consume newline left-over
    System.out.println("Комнаты по площади:");
    List<Room> roomsByArea = rentalSystem.getRoomsByArea(area);
    for (Room room : roomsByArea) {
        System.out.println("Комната по площади: " + room.getAddress());
    }

```



```

        break;
    case 8:
        System.out.println("Выход из программы");
        break;
    default:
        System.out.println("Неверное действие");
    }
}
}

```

RentalSystem:

```

import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;
import java.util.stream.Collectors;

public class RentalSystem {
    private List<Room> rooms;

    public RentalSystem() {
        this.rooms = new ArrayList<>();
    }

    public void addRoom(Room room) {
        rooms.add(room);
    }

    public void removeRoom(Room room) {
        rooms.remove(room);
    }

    public List<Room> getFreeRooms() {
        return rooms.stream().filter(room -> !room.isRented()).collect(Collectors.toList());
    }

    public List<Room> getRentedRooms() {
        return rooms.stream().filter(Room::isRented).collect(Collectors.toList());
    }

    public List<Room> findRooms(int rooms, int floor, double area) {
        return this.rooms.stream()
            .filter(room -> room.getRooms() == rooms && room.getFloor() == floor &&

```

```

Math.abs(room.getArea() - area) <= 10)
    .collect(Collectors.toList());
}

public void rentRoom(Room room) {
    if (rooms.contains(room) && !room.isRented()) {
        room.setRented(true);
    }
}

public List<Room> getRoomsByRoomsNumber(int rooms) {
    return this.rooms.stream().filter(room -> room.getRooms() == rooms).collect(Collectors.toList());
}

public List<Room> getRoomsByRoomsNumberAndFloor(int rooms, int minFloor, int maxFloor) {
    return this.rooms.stream().filter(room -> room.getRooms() == rooms && room.getFloor() >=
minFloor && room.getFloor() <= maxFloor).collect(Collectors.toList());
}

public List<Room> getRoomsByMaxRentPrice(double maxRentPrice) {
    return this.rooms.stream().filter(room -> room.getRentPrice() <=
maxRentPrice).collect(Collectors.toList());
}

public List<Room> getRoomsByAddress(String address) {
    return this.rooms.stream().filter(room ->
room.getAddress().equals(address)).collect(Collectors.toList());
}

public List<Room> getRoomsByArea(double area) {
    return rooms.stream().filter(room -> room.getArea() > area).collect(Collectors.toList());
}

public void printAllRooms() {
    for (Room room : rooms) {
        System.out.println("Количество комнат: " + room.getRooms());
        System.out.println("Площадь: " + room.getArea());
        System.out.println("Этаж: " + room.getFloor());
        System.out.println("Адрес: " + room.getAddress());
        System.out.println("Цена аренды: " + room.getRentPrice());
        System.out.println("Сдана: " + (room.isRented() ? "Да" : "Нет"));
        System.out.println("-----");
    }
}

public void loadRoomsFromFile(String filename) {

```

```

try (BufferedReader br = new BufferedReader(new FileReader(filename))) {
    String line;
    while ((line = br.readLine()) != null) {
        String[] values = line.split("(?=(?:[^\"]*" + "[^\"]*" + "[^\"]*" + "$)", -1);
        try {
            int rooms = Integer.parseInt(values[0]);
            double area = Double.parseDouble(values[1]);
            int floor = Integer.parseInt(values[2]);
            String address = values[3].replace("\"", ""); // убираем кавычки
            double rentPrice = Double.parseDouble(values[4]);
            boolean isRented = Boolean.parseBoolean(values[5]);
            this.addRoom(new Room(rooms, area, floor, address, rentPrice, isRented));
        } catch (NumberFormatException e) {
            System.err.println("Could not parse line: " + line);
            e.printStackTrace();
        }
    }
} catch (IOException e) {
    e.printStackTrace();
}
}

```

Room:

```

public class Room {
    private int rooms;
    private double area;
    private int floor;
    private String address;
    private double rentPrice;
    private boolean isRented;

    public Room(int rooms, double area, int floor, String address, double rentPrice, boolean isRented) {
        this.rooms = rooms;
        this.area = area;
        this.floor = floor;
        this.address = address;
        this.rentPrice = rentPrice;
        this.isRented = isRented;
    }

    // getters

```

```
public int getRooms() {  
    return rooms;  
}
```

```
public double getArea() {  
    return area;  
}
```

```
public int getFloor() {  
    return floor;  
}
```

```
public String getAddress() {  
    return address;  
}
```

```
public double getRentPrice() {  
    return rentPrice;  
}
```

```
public boolean isRented() {  
    return isRented;  
}
```

```
// setters
```

```
public void setRooms(int rooms) {  
    this.rooms = rooms;  
}
```

```
public void setArea(double area) {  
    this.area = area;  
}
```

```
public void setFloor(int floor) {  
    this.floor = floor;  
}
```

```
public void setAddress(String address) {  
    this.address = address;  
}
```

```
public void setRentPrice(double rentPrice) {  
    this.rentPrice = rentPrice;  
}
```

```

    public void setRented(boolean rented) {
        isRented = rented;
    }
}

```

Рисунки с результатами работы программы

```

C:\Users\egor-.jdk\openjdk-21.0.2\bin\java.exe --enable-preview "-javaagent:D:\IntelliJ IDEA Community Edition 2023.3.4\lib\idea_rt.jar=56054;D:\IntelliJ IDEA Community Edit
1. Печать всех комнат:
2. Печать всех свободных комнат:
3. Печать всех сданных комнат:
4. Поиск комнат по количеству комнат и этажу:
5. Получение комнат по максимальной цене аренды:
6. Получение комнат по адресу:
7. Получение комнат по площади:
8. Выход
1
Все комнаты:
Количество комнат: 2
Площадь: 30.5
Этаж: 1
Адрес: ул. Пушкина, д. Колотушкина
Цена аренды: 500.0
Сдана: Нет
-----
Количество комнат: 3
Площадь: 45.0
Этаж: 2
Адрес: ул. Лермонтова, д. 10
Цена аренды: 700.0
Сдана: Да
-----
Количество комнат: 1
Площадь: 18.0
Этаж: 5
Адрес: пр. Мира, д. 14
Цена аренды: 300.0
Сдана: Нет
-----
1. Печать всех комнат:

```

```

Количество комнат: 1
Площадь: 18.0
Этаж: 5
Адрес: пр. Мира, д. 14
Цена аренды: 300.0
Сдана: Нет
-----
1. Печать всех комнат:
2. Печать всех свободных комнат:
3. Печать всех сданных комнат:
4. Поиск комнат по количеству комнат и этажу:
5. Получение комнат по максимальной цене аренды:
6. Получение комнат по адресу:
7. Получение комнат по площади:
8. Выход

данные комнаты:
данная комната: ул. Лермонтова, д. 10
1. Печать всех комнат:
2. Печать всех свободных комнат:
3. Печать всех сданных комнат:
4. Поиск комнат по количеству комнат и этажу:
5. Получение комнат по максимальной цене аренды:
6. Получение комнат по адресу:
7. Получение комнат по площади:
8. Выход

Выход из программы

Process finished with exit code 0

```

Вывод: научился создавать и использовать классы в программах на языке программирования Java