

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ  
УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ  
“БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ”  
КАФЕДРА ИНТЕЛЛЕКТУАЛЬНЫХ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Отчёт  
по лабораторной работе №3

Выполнил:  
студент группы ПО-9  
Качаловский Данил Сергеевич

Проверил:  
Крощенко А. А.

Брест 2024

**Цель работы:** приобрести базовые навыки работы с файловой системой в Java

## **Вариант 7**

### **Задание 1**

Для класса

- Создать поля классов
- Создать методы классов
- Добавьте необходимые get и set методы (по необходимости)
- Укажите соответствующие модификаторы видимости
- Добавьте конструкторы
- Переопределить методы toString() и equals()

Множество символов 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.

**Код программы (файл CharSet.java)**

```
package Lab3_1;

/**
 * Множество символов 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.
 */

public class CharSet {
    private char[] set;
    private int size;

    public CharSet(int capacity) {
        set = new char[capacity];
        size = 0;
    }

    public void add(char element) {
        if (!contains(element)) {
            if (size < set.length) {
                set[size++] = element;
            }
        }
    }

    public void remove(char element) {
        for (int i = 0; i < size; i++) {
```

```

        if (set[i] == element) {
            set[i] = set[size - 1];
            set[size - 1] = '\u0000';
            size--;
            return;
        }
    }
}

public boolean contains(char element) {
    for (int i = 0; i < size; i++) {
        if (set[i] == element) {
            return true;
        }
    }
    return false;
}

public void union(CharSet otherSet) {
    for (int i = 0; i < otherSet.size; i++) {
        if (!contains(otherSet.set[i])) {
            add(otherSet.set[i]);
        }
    }
}

public void print() {
    for (int i = 0; i < size; i++) {
        System.out.print(set[i] + " ");
    }
    System.out.println();
}

public boolean equals(CharSet otherSet) {
    if (this.size != otherSet.size) {
        return false;
    }
    for (int i = 0; i < size; i++) {
        if (this.set[i] != otherSet.set[i]) {
            return false;
        }
    }
    return true;
}
}

```

Код программы (файл Lab3\_1.java)

```
package Lab3_1;

public class Lab3_1 {
    public static void main(String[] args) {
        CharSet set1 = new CharSet(5);
        set1.add('a');
        set1.add('b');
        set1.add('c');
        set1.add('d');
        CharSet set2 = new CharSet(5);
        set2.add('c');
        set2.add('d');
        set2.add('e');
        set1.print();
        set2.print();
        set1.union(set2);
        set1.print();
        System.out.println("Множество set1 содержит 'c': " +
set1.contains('c'));
        System.out.println("Множество set1 содержит 'z': " +
set1.contains('z'));
        set1.remove('c');
        set1.print();
        System.out.println("Множества set1 и set2 равны: " +
set1.equals(set2));
    }
}
```

**Вывод**

a b c d

c d e

a b c d e

Множество set1 содержит 'c': true

Множество set1 содержит 'z': false

a b e d

Множества set1 и set2 равны: false

## Задание 2

Система оповещений на дорожном вокзале

Автоматизированная информационная система на железнодорожном вокзале содержит сведения об отправлении поездов дальнего следования. Составить программу, которая должна хранить расписание поездов в структурированном, отсортированном по времени отправления виде (используя бинарное дерево).

- Обеспечивает первоначальный ввод данных в информационную систему о текущем расписании из файла и формирование дерева;
- Печатает все расписание на экран по команде;
- Выводит информацию о поезде по номеру поезда;
- По названию станции назначения выводит данные обо всех поездах, которые следуют до этой станции;
- Список поездов, следующих до заданного пункта назначения и отправляющихся после заданного часа;
- Список поездов, отправляющихся до заданного пункта назначения и имеющих общие места;
- За 10, 5, 3 минуты до отправления поезда показывает информационное сообщение об отправлении поезда.

**Код программы (Train.java)**

```
package Lab3_2;

class Train {
    int trainNumber;
    String destination;
    String departureTime;
    int availableSeats;

    public Train(int trainNumber, String destination, String departureTime,
int availableSeats) {
        this.trainNumber = trainNumber;
        this.destination = destination;
        this.departureTime = departureTime;
        this.availableSeats = availableSeats;
    }

    @Override
    public String toString() {
        return "Train{" +
            "trainNumber=" + trainNumber +
            ", destination='" + destination + '\'' +
            ", departureTime='" + departureTime + '\'' +
            ", availableSeats=" + availableSeats +
            '}';
    }
}
```

### Код программы (TrainSchedule.java)

```
package Lab3_2;

import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;

/** TODO
 * TODO Обеспечивает первоначальный ввод данных в информационную систему о
текущем расписании из файла и формирование дерева;
 * TODO • Печатает все расписание на экран по команде; OK
 * TODO • Выводит информацию о поезде по номеру поезда;
 * TODO • По названию станции назначения выводит данные обо всех поездах,
которые следуют до этой станции;
 * TODO • Список поездов, следующих до заданного пункта назначения и
отправляющихся после заданного часа;
 * TODO • Список поездов, отправляющихся до заданного пункта назначения и
имеющих общие места;
 * TODO • За 10, 5, 3 минуты до отправления поезда показывает информационное
сообщение об отправлении поезда
 */

public class Node{
    Train train;
    Node left, right;

    public Node(Train train) {
        this.train = train;
        left = right = null;
    }
}

public class TrainSchedule {
    Node root;
    public TrainSchedule(){
        root = null;
    }

    public void insert(Train train) {
        root = insertRec(root, train);
    }

    private Node insertRec(Node root, Train train) {
        if (root == null) {
            root = new Node(train);
            return root;
        }

        if (train.departureTime.compareTo(root.train.departureTime) < 0)
            root.left = insertRec(root.left, train);
        else if (train.departureTime.compareTo(root.train.departureTime) > 0)
            root.right = insertRec(root.right, train);

        return root;
    }
}
```

```

    }

    //Печатает все расписание
    public void printSchedule() {
        printInOrder(root);
    }

    private void printInOrder(Node root) {
        if (root != null) {
            printInOrder(root.left);
            System.out.println(root.train);
            printInOrder(root.right);
        }
    }

    public void findTrainByNumber(int trainNumber) {
        Node result = findTrain(root, trainNumber);
        if (result == null)
            System.out.println("Поезд с номером " + trainNumber + " не найден.");
        else
            System.out.println("Найден поезд: " + result.train);
    }

    private Node findTrain(Node root, int trainNumber) {
        if (root == null || root.train.trainNumber == trainNumber)
            return root;

        if (trainNumber < root.train.trainNumber)
            return findTrain(root.left, trainNumber);

        return findTrain(root.right, trainNumber);
    }

    public void printTrainsToDestination(String destination) {
        printTrainsToDestination(root, destination);
    }

    private void printTrainsToDestination(Node root, String destination) {
        if (root != null) {
            printTrainsToDestination(root.left, destination);
            if (root.train.destination.equals(destination)) {
                System.out.println(root.train);
            }
            printTrainsToDestination(root.right, destination);
        }
    }

    public void printTrainsToDestinationAfterTime(String destination, String time) {
        printTrainsToDestinationAfterTime(root, destination, time);
    }

    private void printTrainsToDestinationAfterTime(Node root, String destination, String time) {

```

```

        if (root != null) {
            printTrainsToDestinationAfterTime(root.left, destination, time);
            if (root.train.destination.equals(destination) &&
root.train.departureTime.compareTo(time) > 0) {
                System.out.println(root.train);
            }
            printTrainsToDestinationAfterTime(root.right, destination, time);
        }
    }

    public void printTrainsToDestinationWithAvailableSeats(String
destination) {
        printTrainsToDestinationWithAvailableSeats(root, destination);
    }

    private void printTrainsToDestinationWithAvailableSeats(Node root, String
destination) {
        if (root != null) {
            printTrainsToDestinationWithAvailableSeats(root.left,
destination);
            if (root.train.destination.equals(destination) &&
root.train.availableSeats > 0) {
                System.out.println(root.train);
            }
            printTrainsToDestinationWithAvailableSeats(root.right,
destination);
        }
    }

    public void notifyTrainDeparture(String time) {
        notifyTrainDeparture(root, time);
    }

    private void notifyTrainDeparture(Node root, String time) {
        if (root != null) {
            notifyTrainDeparture(root.left, time);
            int minutesBeforeDeparture =
Integer.parseInt(root.train.departureTime.split(":")[0])*60 +
Integer.parseInt(root.train.departureTime.split(":")[1]) - (
Integer.parseInt(time.split(":")[0])*60 +
Integer.parseInt(time.split(":")[1])
        );

            if (minutesBeforeDeparture == 3 || minutesBeforeDeparture == 5 ||
minutesBeforeDeparture == 10 ) {
                System.out.println("Оповещение: Поезд " + root.train.trainNumber
+ " отправляется через " +
                    minutesBeforeDeparture + " минут");
            }
            notifyTrainDeparture(root.right, time);
        }
    }
}

```



```

    public void loadScheduleFromFile(String filename) {
        try {
            BufferedReader reader = new BufferedReader(new
FileReader(filename));
            String line;
            while ((line = reader.readLine()) != null) {
                String[] parts = line.split(",");
                if (parts.length == 4) {
                    int trainNumber = Integer.parseInt(parts[0]);
                    String destination = parts[1];
                    String departureTime = parts[2];
                    int availableSeats = Integer.parseInt(parts[3]);
                    insert(new Train(trainNumber, destination, departureTime,
availableSeats));
                }
            }
            reader.close();
        } catch (IOException e) {
            System.out.println("Ошибка при загрузке расписания из файла: " +
e.getMessage());
        }
    }
}

```

#### Код программы (Lab3\_2.java)

```

package Lab3_2;

public class Lab3_2 {
    public static void main(String[] args) {
        TrainSchedule schedule = new TrainSchedule();
        schedule.loadScheduleFromFile("src/Lab3_2/schedule.txt");
        System.out.println("Вывод всех поездов");
        schedule.printSchedule();
        System.out.println("Поиск по номеру");
        schedule.findTrainByNumber(1);
        System.out.println("Вывод по месту прибытия");
        schedule.printTrainsToDestination("Kobrin");
        System.out.println("Вывод по месту прибытия после указанного
времени");
        schedule.printTrainsToDestinationAfterTime("Kobrin", "15:00");
        System.out.println("Вывод по месту прибытия со свободными местами");
        schedule.printTrainsToDestinationWithAvailableSeats("Kobrin");
        schedule.notifyTrainDeparture("11:20");
    }
}

```

Содержимое файла *schedule.txt*

```
1,Brest,10:40,3
5,Kobrin,15:40,0
2,Kobrin,11:30,5
4,Minsk,11:25,10
```

## Вывод

[D:\Libs\openjdk-21.0.2\bin\java.exe](#) "-javaagent:D:\JetBrains\IntelliJ IDEA Ultimate\lib\idea\_rt.j

Вывод всех поездов

```
Train{trainNumber=1, destination='Brest', departureTime='10:40', availableSeats=3}
Train{trainNumber=4, destination='Minsk', departureTime='11:25', availableSeats=10}
Train{trainNumber=2, destination='Kobrin', departureTime='11:30', availableSeats=5}
Train{trainNumber=5, destination='Kobrin', departureTime='15:40', availableSeats=0}
```

Поиск по номеру

Найден поезд: Train{trainNumber=1, destination='Brest', departureTime='10:40', availableSeats=3}

Вывод по месту прибытия

```
Train{trainNumber=2, destination='Kobrin', departureTime='11:30', availableSeats=5}
Train{trainNumber=5, destination='Kobrin', departureTime='15:40', availableSeats=0}
```

Вывод по месту прибытия после указанного времени

```
Train{trainNumber=5, destination='Kobrin', departureTime='15:40', availableSeats=0}
```

Вывод по месту прибытия со свободными местами

```
Train{trainNumber=2, destination='Kobrin', departureTime='11:30', availableSeats=5}
```

Оповещение: Поезд 4 отправляется через 5 минут

Оповещение: Поезд 2 отправляется через 10 минут

### Задание 3

Напишите метод `String repeat(char ch, int repeat)` который строит строку из указанного символа, повторённого заданное количество раз.

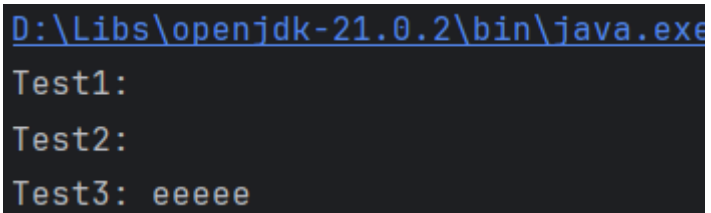
#### Код программы

```
import java.util.Arrays;

public class Main3 {
    public static String repeat(char ch, int repeat){
        return String.valueOf(ch).repeat(Math.max(0, repeat));
    }

    public static void main(String[] args) {
        System.out.println("Test1: " + repeat('e', -4));
        System.out.println("Test2: " + repeat('e', 0));
        System.out.println("Test3: " + repeat('e', 5));
    }
}
```

#### Вывод:



The screenshot shows a terminal window with the command `D:\Libs\openjdk-21.0.2\bin\java.exe` at the top. Below the command, the output of the program is displayed: `Test1:`, `Test2:`, and `Test3: eeeee`.