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, ИСКУССТВЕННЫЙ ИНТЕЛЛЕКТ И СИСТЕМЫ УПРАВЛЕНИЯ**

КАФЕДРА **КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.04.01 Информатика и вычислительная техника**

МАГИСТЕРСКАЯ ПРОГРАММА **09.04.01/07 Интеллектуальные системы анализа, обработки и интерпретации больших данных**

О Т Ч Е Т

по лабораторной работе №6

Название: Коллекции

Дисциплина: Языки программирования для работы с большими данными

Студент ИУ6-22М (Группа) _____ (Подпись, дата) М.А. Зотов (И.О. Фамилия)

Преподаватель _____ (Подпись, дата) П.В. Степанов (И.О. Фамилия)

Москва, 2023 г.

Цель: получить опыт работы с коллекциями.

Вариант 1 – задание 1

Условие: определить множество на основе множества целых чисел. Создать методы для определения пересечения и объединения множеств.

Код:

```
import java.util.HashSet;
import java.util.Set;

class IntSet {
    private final Set<Integer> set;

    public IntSet() {
        set = new HashSet<>();
    }

    public IntSet(Set<Integer> set) {
        this.set = set;
    }

    public void add(int n) {
        set.add(n);
    }

    public IntSet intersection(IntSet otherSet) {
        Set<Integer> intersection = new HashSet<>(set);
        intersection.retainAll(otherSet.set);
        return new IntSet(intersection);
    }

    public IntSet union(IntSet otherSet) {
        Set<Integer> union = new HashSet<>(set);
        union.addAll(otherSet.set);
        return new IntSet(union);
    }

    public String toString() {
        StringBuilder sb = new StringBuilder();
        for (Integer elem : set)
            sb.append(elem).append(" ");
        return sb.toString();
    }
}

public class var1_ex1 {
    public static void main(String[] args) {
        IntSet set1 = new IntSet();
        set1.add(1);
        set1.add(2);
        set1.add(3);
```

```
        IntSet set2 = new IntSet();
        set2.add(2);
        set2.add(3);
        set2.add(4);

        IntSet intersection = set1.intersection(set2);
        IntSet union = set1.union(set2);

        System.out.println("Set 1: " + set1);
        System.out.println("Set 2: " + set2);
        System.out.println("Intersection: " + intersection);
        System.out.println("Union: " + union);
    }

}
```

Результат выполнения:

```
Set 1: 1 2 3
Set 2: 2 3 4
Intersection: 2 3
Union: 1 2 3 4
```

Рисунок 1 – Результат выполнения 1.9

Вариант 1 – задание 9

Условие: задан файл с текстом на английском языке. Выделить все различные слова. Слова, отличающиеся только регистром букв, считать одинаковыми. Использовать класс HashSet.

Код:

```
import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.io.IOException;
import java.util.HashSet;
import java.util.Set;

public class var1_ex9 {
    public static void main(String[] args) {
        File inputFile = new File("var1_ex9.txt");
        Set<String> uniqueWords = new HashSet<>();
        try {
            FileReader fileReader = new FileReader(inputFile);
            BufferedReader bufferedReader = new
BufferedReader(fileReader);
            String line;
            while ((line = bufferedReader.readLine()) != null) {
                String[] words = line.split("\\s+");
                for (String word : words) {
                    uniqueWords.add(word.toLowerCase());
                }
```

```
            }
            bufferedReader.close();

            System.out.println("Unique words:");
            for (String word : uniqueWords) {
                System.out.println(word);
            }
        } catch (IOException e) {
            System.out.println("Error reading file: " +
e.getMessage());
        }
    }
}
```

Результат выполнения:

```
Unique words:
orange
apple
cucumber
limon
```

Рисунок 2 – Результат выполнения 1.10

Вариант 2 – задание 9

Условие: дана матрица из целых чисел. Найти в ней прямоугольную подматрицу, состоящую из максимального количества одинаковых элементов. Использовать класс Stack.

Код:

```
import java.util.Stack;

public class var2_ex9 {
    public static int[] findMaxRectangularSubmatrix(int[][]
matrix) {
        int numRows = matrix.length;
        int numCols = matrix[0].length;
        int[] result = new int[4]; // [rowStart, colStart,
rowEnd, colEnd]
        int maxArea = 0;

        for (int i = 0; i < numRows; i++) {
            int[] heights = new int[numCols];
            for (int j = 0; j < numCols; j++) {
                heights[j] = matrix[i][j];
            }
            int[] subResult =
findMaxRectangularSubarray(heights);
            int area = (subResult[2] - subResult[0] + 1) *
(subResult[3] - subResult[1] + 1);
            if (area > maxArea) {
                maxArea = area;
                result[0] = i - (subResult[2] - subResult[0]);
                result[1] = subResult[1];
```

```
                result[2] = i;
                result[3] = subResult[3];
            }
        }

        return result;
    }

    public static int[] findMaxRectangularSubarray(int[]
heights) {
        int numCols = heights.length;
        Stack<Integer> stack = new Stack<>();
        int[] left = new int[numCols];
        int[] right = new int[numCols];

        for (int i = 0; i < numCols; i++) {
            while (!stack.empty() && heights[stack.peek()] >=
heights[i]) {
                stack.pop();
            }
            left[i] = (stack.empty() ? 0 : stack.peek() + 1);
            stack.push(i);
        }

        stack.clear();

        for (int i = numCols - 1; i >= 0; i--) {
            while (!stack.empty() && heights[stack.peek()] >=
heights[i]) {
                stack.pop();
            }
            right[i] = (stack.empty() ? numCols - 1 :
stack.peek() - 1);
            stack.push(i);
        }

        int maxArea = 0;
        int[] result = new int[4]; // [rowStart, colStart,
rowEnd, colEnd]

        for (int i = 0; i < numCols; i++) {
            int area = heights[i] * (right[i] - left[i] + 1);
            if (area > maxArea) {
                maxArea = area;
                result[0] = 0;
                result[1] = left[i];
                result[2] = heights[i] - 1;
                result[3] = right[i];
            }
        }

        return result;
    }
```

```
    public static void main(String[] args) {
        int[][] matrix = {
                {1, 2, 3, 4, 5},
                {1, 1, 1, 2, 2},
                {2, 2, 2, 2, 3},
                {3, 3, 3, 3, 3},
                {4, 4, 4, 4, 4}};

        int[] res = findMaxRectangularSubmatrix(matrix);

        for (int i = res[0]; i <= res[2]; i++) {
            for (int j = res[1]; j <= res[3]; j++)
                System.out.print(matrix[i][j] + "\t");
            System.out.println();
        }
    }
}
```

Результат выполнения:

```
1   1   1   2   2
2   2   2   2   3
3   3   3   3   3
4   4   4   4   4
```

Рисунок 3 – Результат выполнения 2.9

Вариант 2 – задание 10

Условие: на прямой гоночной трассе стоит N автомобилей, для каждого из которых известны начальное положение и скорость. Определить, сколько произойдет обгонов.

Код:

```
public class var2_ex10 {
    public static int countOvertakings(int[] positions, int[]
speeds) {
        int numCars = positions.length;
        int numOvertakings = 0;

        for (int i = 0; i < numCars; i++) {
            for (int j = i + 1; j < numCars; j++) {
                if (positions[i] < positions[j] && speeds[i] >
speeds[j]) {
                    numOvertakings++;
                }
            }
        }

        return numOvertakings;
    }

    public static void main(String[] args) {
        int[] positions = {10, 20, 30, 40};
```

```
        int[] speeds = {3, 5, 2, 4};

        int numOvertakings = countOvertakings(positions,
speeds);

        System.out.println("Number of overtakings: " +
numOvertakings);
    }
}
```

Результат выполнения:

```
Number of overtakings: 3
```

Рисунок 4 – Результат выполнения 2.10

Вывод: в ходе выполнения лабораторной работы был получен опыт работы с коллекциями.