

**Министерство образования Республики Беларусь**

**Учреждение образования**

**«Брестский государственный технический университет»**

Кафедра ИИТ

**ОТЧЕТ**

**ЛАБОРАТОРНАЯ РАБОТА №2**

Выполнил студент  
группы АС-50:  
Красильский П. А.  
Проверил:  
Крощенко А. А.

Брест 2020

Цель работы: научиться разрабатывать простейшие программы на языке программирования Java, получить практический опыт работы с компилятором javac.

**Задание 1. Напишите программу, выполняющую чтение текстовых данных из файла и их последующую обработку:**

7) Необходимо подсчитать число цифр в текстовом файле. Локализовать и вывести на экран строку, содержащую цифру с порядковым номером  $n/2$ , где  $n$ —общее количество подсчитанных цифр.

```
package com.company;

import java.nio.charset.StandardCharsets;
import java.nio.file.Files;
import java.nio.file.Paths;
import java.util.HashMap;
import java.util.Iterator;
import java.util.List;

public class Main {
    public Main() {
    }

    public static void main(String[] args) throws Exception {
        List<String> lines = Files.readAllLines(Paths.get("file1.txt"), StandardCharsets.UTF_8);
        HashMap<Integer, Integer> map = new HashMap();
        String regex = "[0-9]+";
        int count = 0;
        int numline = 0;
        Iterator var6 = lines.iterator();

        while(var6.hasNext()) {
            String line = (String)var6.next();
            ++numline;
            String[] var8 = line.split("");
            int var9 = var8.length;

            for(int var10 = 0; var10 < var9; ++var10) {
                String symbol = var8[var10];
                if (symbol.matches(regex)) {
                    ++count;
                    map.put(count, numline);
                }
            }
        }

        int pointer = count / 2;
        System.out.println((String)lines.get((Integer)map.get(pointer) - 1));
    }
}
```



**Задание 2 Написать консольную утилиту, обрабатывающую ввод пользователя и дополнительные ключи. Проект упаковать в jar-файл, написать bat-файл для запуска.**

7) Утилита uniq отфильтровывает повторяющиеся строки во входном файле. Если входной файл задан как – или не задан вовсе, то чтение производится из стандартного ввода. Если выходной файл не задан, запись производится в стандартный вывод. Если одна и та же строка встречается

второй и более разы, то она не записывается в вывод программы. Формат использования: `uniq [-c | -d | -u] [-i] [входной_файл [выходной_файл]]`, где ключи имеют следующее значение:

- -u Выводить только те строки, которые не повторяются на входе.
- -d Выводить только те строки, которые повторяются на входе.
- -c Перед каждой строкой выводить число повторений этой строки на входе и один пробел.
- -i Сравнивать строки без учёта регистра.

```
package com.company;

import java.io.IOException;
import java.nio.charset.StandardCharsets;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.*;

public class Main {

    public static void main(String[] args) throws IOException {
        System.out.println("Please, input params by pattern ([ -c | -d | -u] [-i] [Path_input_file [Path_output_file]])\nFor example: uniq -c -i file1.txt file2.txt");
        Scanner scanner = new Scanner(System.in);
        String input;
        List<String> defInput = new ArrayList<>();
        String defOutput = "File2.txt";
        List<String> params;
        int i = 0;
        input = scanner.nextLine();
        params = Arrays.asList(input.split(" "));
        if (params.size() >= 2 && params.size() <= 5) {
            if (params.get(0).equals("uniq") && (params.get(1).equals("-c") || params.get(1).equals("-d") || params.get(1).equals("-u"))) {
                switch (params.get(1)) {
                    case "-c":
                        if (params.size() == 5) {
                            if (params.get(2).equals("-i")) {
                                if (params.get(3).equals("-")) {
                                    System.out.println("Input number your strings: ");
                                    i = Integer.parseInt(scanner.next());
                                    System.out.println("Enter your strings: ");
                                    for (int j = 0; j < i; j++)
                                        defInput.add(scanner.next());
                                    uniqC(defInput, params.get(4), true); // -i - out
                                } else {
                                    uniqC(params.get(3), params.get(4), true); // -i in out
                                }
                            }
                        }
                    else if (params.size() == 4) {
                        if (params.get(2).equals("-i")) {
                            if (params.get(3).equals("-")) {
                                System.out.println("Input number your strings: ");
                                i = Integer.parseInt(scanner.next());
                                System.out.println("Enter your strings: ");
                                for (int j = 0; j < i; j++)
                                    defInput.add(scanner.next());
                                uniqC(defInput, defOutput, true); // -i -
                            } else {
                                System.out.println("Input number your strings: ");
                                i = Integer.parseInt(scanner.next());
                                System.out.println("Enter your strings: ");
                                for (int j = 0; j < i; j++)
                                    defInput.add(scanner.next());
                            }
                        }
                    }
                }
            }
        }
    }
}
```

```

        uniqC(defInput, params.get(3), true); //-i out
    }
} else if (params.get(2).equals("-")) {
    System.out.println("Input number your strings: ");
    i=Integer.parseInt(scanner.next());
    System.out.println("Enter your strings: ");
    for (int j=0; j<i;j++)
        defInput.add(scanner.next());
    uniqC(defInput, params.get(3), false); //- out
} else {
    uniqC(params.get(2), params.get(3), false); //in out
}
} else if (params.size() == 3) {
    if (params.get(2).equals("-i")) {
        System.out.println("Input number your strings: ");
        i=Integer.parseInt(scanner.next());
        System.out.println("Enter your strings: ");
        for (int j=0; j<i;j++)
            defInput.add(scanner.next());
        uniqC(defInput, defOutput, true); //-i
    } else if (params.get(2).equals("-")) {
        System.out.println("Input number your strings: ");
        i=Integer.parseInt(scanner.next());
        System.out.println("Enter your strings: ");
        for (int j=0; j<i;j++)
            defInput.add(scanner.next());
        uniqC(defInput, defOutput, false); //-
    } else {
        System.out.println("Input number your strings: ");
        i=Integer.parseInt(scanner.next());
        System.out.println("Enter your strings: ");
        for (int j=0; j<i;j++)
            defInput.add(scanner.next());
        uniqC(defInput, params.get(2), false); //out
    }
} else if (params.size() == 2) {
    System.out.println("Input number your strings: ");
    i=Integer.parseInt(scanner.next());
    System.out.println("Enter your strings: ");
    for (int j=0; j<i;j++)
        defInput.add(scanner.next());
    uniqC(defInput, defOutput, false); //
}
break;
case "-d":
    if (params.size() == 5) {
        if (params.get(2).equals("-i")) {
            if (params.get(3).equals("-")) {
                System.out.println("Input number your strings: ");
                i=Integer.parseInt(scanner.next());
                System.out.println("Enter your strings: ");
                for (int j=0; j<i;j++)
                    defInput.add(scanner.next());
                uniqD(defInput, params.get(4), true); // -i - out
            } else {
                uniqD(params.get(3), params.get(4), true); // -i in out
            }
        }
    }
} else if (params.size() == 4) {
    if (params.get(2).equals("-i")) {
        if (params.get(3).equals("-")) {
            System.out.println("Input number your strings: ");

```

```

        i=Integer.parseInt(scanner.next());
        System.out.println("Enter your strings: ");
        for (int j=0; j<i;j++)
            defInput.add(scanner.next());
        uniqD(defInput, defOutput, true); //-i -
    } else {
        System.out.println("Input number your strings: ");
        i=Integer.parseInt(scanner.next());
        System.out.println("Enter your strings: ");
        for (int j=0; j<i;j++)
            defInput.add(scanner.next());
        uniqD(defInput, params.get(3), true); //-i out
    }
} else if (params.get(2).equals("-")) {
    System.out.println("Input number your strings: ");
    i=Integer.parseInt(scanner.next());
    System.out.println("Enter your strings: ");
    for (int j=0; j<i;j++)
        defInput.add(scanner.next());
    uniqD(defInput, params.get(3), false); //- out
} else {
    uniqD(params.get(2), params.get(3), false); //in out
}
} else if (params.size() == 3) {
    if (params.get(2).equals("-i")) {
        System.out.println("Input number your strings: ");
        i=Integer.parseInt(scanner.next());
        System.out.println("Enter your strings: ");
        for (int j=0; j<i;j++)
            defInput.add(scanner.next());
        uniqD(defInput, defOutput, true); //-i
    } else if (params.get(2).equals("-")) {
        System.out.println("Input number your strings: ");
        i=Integer.parseInt(scanner.next());
        System.out.println("Enter your strings: ");
        for (int j=0; j<i;j++)
            defInput.add(scanner.next());
        uniqD(defInput, defOutput, false); //-
    } else {
        System.out.println("Input number your strings: ");
        i=Integer.parseInt(scanner.next());
        System.out.println("Enter your strings: ");
        for (int j=0; j<i;j++)
            defInput.add(scanner.next());
        uniqD(defInput, params.get(2), false); //out
    }
} else if (params.size() == 2) {
    System.out.println("Input number your strings: ");
    i=Integer.parseInt(scanner.next());
    System.out.println("Enter your strings: ");
    for (int j=0; j<i;j++)
        defInput.add(scanner.next());
    uniqD(defInput, defOutput, false); //
}
break;
case "-u":
    if (params.size() == 5) {
        if (params.get(2).equals("-i")) {
            if (params.get(3).equals("-")) {
                System.out.println("Input number your strings: ");
                i=Integer.parseInt(scanner.next());
                System.out.println("Enter your strings: ");

```

```

        for (int j=0; j<i;j++)
            defInput.add(scanner.next());
        uniqU(defInput, params.get(4), true); // -i - out
    } else {
        uniqU(params.get(3), params.get(4), true); // -i in out
    }
}
} else if (params.size() == 4) {
    if (params.get(2).equals("-i")) {
        if (params.get(3).equals("-")) {
            System.out.println("Input number your strings: ");
            i=Integer.parseInt(scanner.next());
            System.out.println("Enter your strings: ");
            for (int j=0; j<i;j++)
                defInput.add(scanner.next());
            uniqU(defInput, defOutput, true); // -i -
        } else {
            System.out.println("Input number your strings: ");
            i=Integer.parseInt(scanner.next());
            System.out.println("Enter your strings: ");
            for (int j=0; j<i;j++)
                defInput.add(scanner.next());
            uniqU(defInput, params.get(3), true); // -i out
        }
    } else if (params.get(2).equals("-")) {
        System.out.println("Input number your strings: ");
        i=Integer.parseInt(scanner.next());
        System.out.println("Enter your strings: ");
        for (int j=0; j<i;j++)
            defInput.add(scanner.next());
        uniqU(defInput, params.get(3), false); // - out
    } else {
        uniqU(params.get(2), params.get(3), false); // in out
    }
}
} else if (params.size() == 3) {
    if (params.get(2).equals("-i")) {
        System.out.println("Input number your strings: ");
        i=Integer.parseInt(scanner.next());
        System.out.println("Enter your strings: ");
        for (int j=0; j<i;j++)
            defInput.add(scanner.next());
        uniqU(defInput, defOutput, true); // -i
    } else if (params.get(2).equals("-")) {
        System.out.println("Input number your strings: ");
        i=Integer.parseInt(scanner.next());
        System.out.println("Enter your strings: ");
        for (int j=0; j<i;j++)
            defInput.add(scanner.next());
        uniqU(defInput, defOutput, false); // -
    } else {
        System.out.println("Input number your strings: ");
        i=Integer.parseInt(scanner.next());
        System.out.println("Enter your strings: ");
        for (int j=0; j<i;j++)
            defInput.add(scanner.next());
        uniqU(defInput, params.get(2), false); // out
    }
}
} else if (params.size() == 2) {
    System.out.println("Input number your strings: ");
    i=Integer.parseInt(scanner.next());
    System.out.println("Enter your strings: ");
    for (int j=0; j<i;j++)

```

```

        defInput.add(scanner.next());
        uniqU(defInput, defOutput, false); //
    }
    break;
default:
    System.out.println("Not correct param for read");
    break;
}
} else {
    System.out.println("Not correct param/method");
}
} else if (params.size() == 1 && params.get(0).equals("uniq")) {
    System.out.println("Please, input at least 1 param");
} else {
    System.out.println("Not correct method, try again");
}
}
scanner.close();
}

static void uniqU(String inputPath, String outputPath, boolean ignoreCase) throws IOException {
    List<String> lines = Files.readAllLines(Paths.get(inputPath), StandardCharsets.UTF_8);
    int i = 0;
    for (String line: lines) {
        if (ignoreCase) {
            lines.set(i, line.toLowerCase());
        }
        i++;
    }
    Set<String> s = new LinkedHashSet<>(lines);
    Files.write(Path.of(outputPath), s);
    if (ignoreCase)
        System.out.println("Util uniq with keys -u, -i is successfully completed");
    else
        System.out.println("Util uniq with key -u is successfully completed");
}

static void uniqU(List<String> lines, String outputPath, boolean ignoreCase) throws IOException {
    int i = 0;
    for (String line: lines) {
        if (ignoreCase) {
            lines.set(i, line.toLowerCase());
        }
        i++;
    }
    Set<String> s = new LinkedHashSet<>(lines);
    Files.write(Path.of(outputPath), s);
    if (ignoreCase)
        System.out.println("Util uniq with keys -u, -i is successfully completed");
    else
        System.out.println("Util uniq with key -u is successfully completed");
}

static void uniqD(String inputPath, String outputPath, boolean ignoreCase) throws IOException {
    List<String> lines = Files.readAllLines(Paths.get(inputPath), StandardCharsets.UTF_8);
    List<String> outLines = new ArrayList<>();
    int i = 0;
    for (String line : lines) {
        for (int j = 0; j < lines.size(); j++) {
            if (i != j) {
                if (ignoreCase) {
                    if (line.equalsIgnoreCase(lines.get(j))) {

```

```

        outLines.add(line);
    }
    } else {
        if (line.equals(lines.get(j))) {
            outLines.add(line);
        }
    }
}
}
i++;
}
Set<String> s = new LinkedHashSet<>(outLines);
Files.write(Path.of(outputPath),s);
if (ignoreCase)
    System.out.println("Util uniq with keys -d, -i is successfully completed");
else
    System.out.println("Util uniq with key -d is successfully completed");
}

static void uniqD(List<String> lines, String outputPath, boolean ignoreCase) throws IOException {
    List<String> outLines = new ArrayList<>();
    int i = 0;
    for (String line : lines) {
        for (int j = 0; j < lines.size(); j++) {
            if (i != j) {
                if (ignoreCase) {
                    if (line.equalsIgnoreCase(lines.get(j))) {
                        outLines.add(line);
                    }
                } else {
                    if (line.equals(lines.get(j))) {
                        outLines.add(line);
                    }
                }
            }
        }
    }
    i++;
}
Set<String> s = new LinkedHashSet<>(outLines);
Files.write(Path.of(outputPath),s);
if (ignoreCase)
    System.out.println("Util uniq with keys -d, -i is successfully completed");
else
    System.out.println("Util uniq with key -d is successfully completed");
}

static void uniqC(String inputPath, String outputPath, boolean ignoreCase) throws IOException {
    List<String> lines = Files.readAllLines(Path.of(inputPath), StandardCharsets.UTF_8);
    HashMap<String, Integer> hashMap = new HashMap<>();
    StringBuilder stringBuilder = new StringBuilder();
    for (String line : lines) {
        int repeats = 0;
        for (int j = 0; j < lines.size(); j++) {
            if (ignoreCase) {
                if (line.equalsIgnoreCase(lines.get(j))) {
                    repeats++;
                }
            } else {
                if (line.equals(lines.get(j))) {
                    repeats++;
                }
            }
        }
    }
}

```

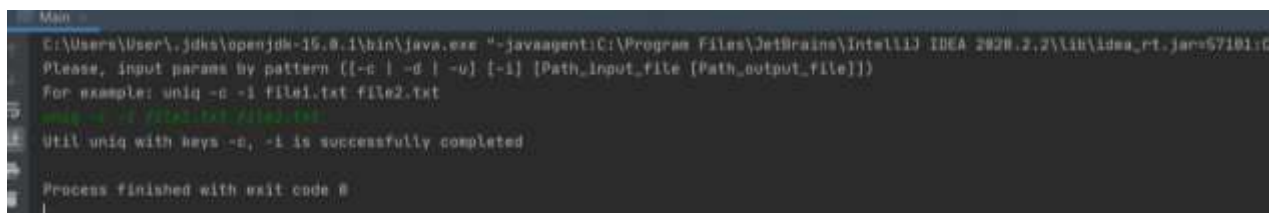


```

        hashMap.put(line, repeats);
    }
}
for (String line : lines) {
    stringBuilder.append(hashMap.get(line)).append(" ").append(line).append("\n");
}
Files.write(Path.of(outputPath), Collections.singleton(stringBuilder));
if (ignoreCase)
    System.out.println("Util uniq with keys -c, -i is successfully completed");
else
    System.out.println("Util uniq with key -c is successfully completed");
}

static void uniqC(List<String> lines, String outputPath, boolean ignoreCase) throws IOException {
    HashMap<String, Integer> hashMap = new HashMap<>();
    StringBuilder stringBuilder = new StringBuilder();
    for (String line : lines) {
        int repeats = 0;
        for (int j = 0; j < lines.size(); j++) {
            if (ignoreCase) {
                if (line.equalsIgnoreCase(lines.get(j))) {
                    repeats++;
                }
            } else {
                if (line.equals(lines.get(j))) {
                    repeats++;
                }
            }
        }
        hashMap.put(line, repeats);
    }
    for (String line : lines) {
        stringBuilder.append(hashMap.get(line)).append(" ").append(line).append("\n");
    }
    Files.write(Path.of(outputPath), Collections.singleton(stringBuilder));
    if (ignoreCase)
        System.out.println("Util uniq with keys -c, -i is successfully completed");
    else
        System.out.println("Util uniq with key -c is successfully completed");
}
}

```



```

Main
C:\Users\User\jdk8\openjdk-15.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2020.2.2\lib\idea_rt.jar=57181:11"
Please, input params by pattern [{-c | -d | -u} [-i] [Path_input_file [Path_output_file]]]
For example: uniq -c -i file1.txt file2.txt
uniq -c -i file1.txt file2.txt
Util uniq with keys -c, -i is successfully completed
Process finished with wait code 0

```