2017-04-30_lab-03

Collaborators

Deniz Ilyasoglu
Tony Dorfmeister

spent time

8h

lessons learned

- read and write files to disk
- that you can iterate through an array using a char type
- how to retrieve bytes arrays from different file types

exercise 01

Dateien werden eingelesen, in dem man den Pfad einer Datei in einem Path Objekt speichert, welches dann wiederum mit Hilfe der Methode `new InputStream(path)` der Klasse InputStream in einem InputStream gespeichert wird. Mit der Methode `read()` wird dann der nächste Buchstabe einer Textdatei ausgegeben.

```
public void readFirstCharacterOfFile(String uri) throws IOException {
        Path path = Paths.get(uri);
        is = Files.newInputStream(path);
        char output = (char) is.read();
        System.out.println("First character of specified file is: " +
output);
    }
```

exercise 02

Mit Hilfe eines PrintWriter Objekts, welches den gewünschten Namen der Datei als Parameter erhält, wird mit der Methode `println(fileName)` der gewünschte String, Integer oder auch int in einer Textdatei ausgegeben. Mit der Methode `close()` muss dann der Stream geschlossen werden, da sonst keine Werte ausgegeben werden können.

```
public void writeToFile(String text) throws FileNotFoundException {
        String file = "stringToFile.txt";
        PrintWriter writer = new PrintWriter(file);
        writer.println(text);
        writer.close();
    }
```

exercise 03

Als erstes wird mit der Methode `readAllBytes(path)` der Files Klasse die Datei Buchstabe für Buchstabe ausgelesen, welche als Bytes in einem Byte Array gespeichert werden. Mit einem for-loop wird dann über dieses Array iteriert und in einem int Array(frequencyArray), in welchem jedes Element für einen Buchstaben im Alphabet steht, wird der Wert, der für die Frequenz eines Buchstaben steht erhöht, sobald sich dieser wiederholt. In einem weiteren for-loop wird über das frequencyArray iteriert und die Häufigkeiten werden als Integer in einem String gespeichert, welcher wieder mit Hilfe eines PrintWriter Objekts in einer Textdatei ausgegeben wird.

```
public void writeFrequenciesToFile(String str, boolean button, String
fileLocation) throws IOException {
        byte[] bytes;
        int[] frequencies = new int['z' + 1];

        // check if str is a path or just a string by checking the boolean
button
        if (button) {
            Path path = Paths.get(str);
            bytes = Files.readAllBytes(path);
        } else {
            StringBufferInputStream is = new StringBufferInputStream(str);
            bytes = new byte[is.available()];
            for (int i = 0; i < bytes.length; i++) {
                is.read(bytes);
            }
        }

        // add found characters to new frequencies
        for (int i = 0; i < bytes.length; i++) {
            if (bytes[i] >= 'A' && bytes[i] <= 'Z') {
                frequencies[bytes[i]]++;
            }
            if (bytes[i] >= 'a' && bytes[i] <= 'z') {
                frequencies[bytes[i]]++;
            }
        }

        this.frequencies = frequencies;

        // calculate all frequencies
        String result = "";
        for (int i = 0; i < frequencies.length; i++) {
            if (frequencies[i] != 0) {
                result += (char) i + " - Frequency = " + frequencies[i] +
"\n";
            }
        }

        // print frequencies to file
        if (fileLocation != null) {
            PrintWriter writer = new PrintWriter(fileLocation);
            writer.println(result);
            writer.close();
        }
    }
```

exercise 04

Für die Testcases mussten wir eine getter Methode, für die in der Methode `printFrequenciesToFile()` erstellten frequencyArray implementieren.

Die Implementierung des StringBufferInputStream wurde in der `printFrequenciesToFile()` vorgenommen.

```
public void testFrequencies() throws IOException{
        String output = "frequencyJUnit.txt";
        main.writeFrequenciesToFile("abbcccAAABBC |[“ “}{¶≠ 98135685",
false, output);
        int[] result = main.getFrequencyArray();

        assertTrue(result['A'] == 3);
        assertTrue(result['B'] == 2);
        assertTrue(result['C'] == 1);
        assertTrue(result['a'] == 1);
        assertTrue(result['b'] == 2);
        assertTrue(result['c'] == 3);
        assertTrue(result[9] == 0);
    }
```

exercise 05

Das *frequencyArray* aus Aufgabe 3 wird hier wieder verwendet. In einem for-loop wird über dieses iteriert und mit einer if-Klausel wird das aktuelle Element des Arrays mit einer weiteren Laufvariable *f* verglichen. Falls das Element einen höheren Wert hat, wird es in *f* gespeichert. Der Buchstabe mit der höchsten Frequenz wird dann ausgegeben.

```
public char getHighestFrequencyCharacter(String str, boolean button) throws
IOException {
        writeFrequenciesToFile(str, button, null);
        int[] freqArray = getFrequencyArray();
        int f = 0;
        char c = ' ';

        for (int i = 0; i < freqArray.length; i++) {
            if (freqArray[i] > f){
                f = freqArray[i];
                c = (char) i;
            }
        }
        return c;
    }
```

exercise 06

Auch hier wird das *frequencyArray* aus Aufgabe 3 verwendet. In einem for-loop wird über dieses iteriert und in einem weiteren verschachtelten for-loop wird entsprechend der Häufigkeit eine Anzahl an Sternchen(*) in einem String gespeichert. Die Häufigkeit eines jeden Character wird zur Normalisierung durch einen user-spezifischen Threshold angegeben. Für jeden Buchstaben wird dies dann als Histogramm in der Konsole ausgegeben.

```
public void printHistogram(String str, boolean button, int threshold) throws
IOException{
        writeFrequenciesToFile(str, button, null);
        int[] freq = getFrequencyArray();
        String result = "";

        for (int i = 0; i < freq.length; i++){
            if (freq[i] != 0){
                String starCount = "";
                for (int h = 0; h < (freq[i]/threshold); h++){
                    starCount += "*";
                }
                result += (char) i + " : " + starCount + "\n";
            }
        }
        System.out.println(result);
    }
```

exercise 07

Die Häufigkeit für "a" = 22403585

Die Häufigkeit für "z" = 279224

exercise 08

Komplexität für `writeFrequenciesToFile()` = O(2n)

Komplexität für `getHighestFrequencyCharacter` = O(3n)

Komplexität für `printHistogram()` = O(2n + n^2)