컴프2 5주차 : 단어 수 세기

과제 목표 : Map을 활용하여 단어의 개수를 세고 최댓값, 최솟값을 구한다. 또한 파일 입출력을 활용하여 파일을 읽고 쓸 수 있다.

학번 : 201902722

이름 : 유형곤

큰 과제 : 단어 수 세기

해결과정의 흐름 :

과제에서 요구하는 것은 가장 출현 빈도가 높은, 또는 낮은 단어를 출력하는 것입니다.

우선 과제를 하기 위해선 단어의 개수를 세서 저장해야 합니다. 따라서 Map을 활용하여 단어마다 개수를 세었습니다. 따라서 Key : 단어, Value : 단어의 개수 에 해당하는 자료구조가 생겼습니다. 그런데 최댓값, 최솟값에 해당하는 단어를 구하기 위해선 먼저 최댓값과 최솟값이 무엇인지를 구해야 합니다. 따라서 최댓값과 최솟값을 먼저 구했고, 구한 최댓값과 최솟값에 해당하는 key를 리스트에 넣어서 출력했습니다.

작은 과제 : 단어 수 세기

```
public static void main(String[] args) {
    File file = new File("input.txt");
    BufferedReader br = null;
    try {
        br = new BufferedReader(new FileReader(file));
        String line = br.readLine();
        HashMap<String, Integer> map = new HashMap<String, Integer>();

        //단어의 수를 세어서 map에 저장한다. (key : 단어, value : 개수)
        while(line != null) {
            StringTokenizer st = new StringTokenizer(line);
            while(st.hasMoreTokens()) {
                String key = st.nextToken();
                if(!map.containsKey(key)){
                    map.put(key, 1);
                } else {
                    map.put(key, map.get(key) + 1);
                }
            }
            line = br.readLine();
        }
```

우선, 단어를 세기 전 BufferedReader와 map을 초기화합니다. (BufferedReader 객체 생성시 처리하는 예외와, br.readLine()에서 처리하는 예외는 이후 catch문에서 처리합니다.) 그 후 단어를 세는데, line에 br.readLine() 함수를 이용하여 한 줄 한 줄씩 읽어온 후, StringTokenizer로 띄어쓰기를 기준으로 한 단어씩 읽어옵니다. 한 단어를 읽어오면 먼저 map에 해당 단어가 존재하는지 확인하고, 없으면 1을 넣고, 이미 존재하면 기존 값에 1을 더합니다.

작은 과제 : 최대/최솟값 구하기

```
//최댓값과 최솟값을 찾는다.
int max = 0;
int min = Integer.MAX_VALUE;
List<Integer> valueList = new ArrayList<Integer>(map.values());
Iterator<Integer> iterInt = valueList.iterator();
while(iterInt.hasNext()) {
    int val = iterInt.next();
    if(max < val) {
        max = val;
    }
    if(min > val) {
        min = val;
    }
}
```

최대, 최솟값을 구하는 것은 value만 알아도 되므로 map.values()를 list에 담은 후, Iterator로 순회하면서 최댓값과 최솟값을 구했습니다. (여기서 valueList가 아니라 valueSet을 쓸까도 했지만 만 더 느려질 것 같아서 관뒀습니다.)

작은 과제 : 단어 출현 빈도가 최대/최소인 단어를 리스트에 저장하기

```
//value가 max, min에 해당하는 key들을 리스트에 넣는다.
LinkedList<String> maxList = new LinkedList<String>();
LinkedList<String> minList = new LinkedList<String>();
Set<String> keySet = map.keySet();
for(String key : keySet) {
    int val = map.get(key);
    if(val == min) {
        minList.add(key);
    }
    if(val == max) {
        maxList.add(key);
    }
}
```

이제 각각의 maxList, minList에, 단어의 출현 빈도수가 최댓값, 최솟값인 단어를 넣었습니다.

이때, map을 순회할 때에는 keySet() 함수를 호출하였습니다.

작은 과제 : 저장한 단어(최대, 최소인 단어)를 출력하기

```
//결과를 출력하기 위해 문자스트림을 연다.
file = new File("output.txt");
PrintWriter pw = new PrintWriter(new FileWriter(file));
StringBuilder sb = new StringBuilder();

//max 리스트를 출력한다.
Iterator<String> iterStr = maxList.iterator();
sb.append("max key : [");
while(iterStr.hasNext()) {
    sb.append(iterStr.next()).append(", ");
}
sb.delete(sb.length()-2, sb.length()); //마지막 콤마 (", ")는 지워준다.
sb.append("] / ").append(Integer.toString(max));
System.out.println(sb.toString());
```

maxList를 출력하기 위해서 Iterator를 선언하여 maxList의 원소를 하나씩 순회하였는데, 이때 출력할 문자열은, 문자열의 덧셈 연산이 느린 것을 감안하여 StringBuilder를 사용하여 합쳤습니다. 또한 마지막에 sb.delete(sb.length()-2, sb.length())는 마지막에 콤마가 붙는걸 방지하기 위해서 마지막 두 글자를 지운 것입니다.

```
//min 리스트를 출력한다.
iterStr = minList.iterator();
sb = new StringBuilder();
sb.append("min key : [");
while(iterStr.hasNext()) {
    sb.append(iterStr.next()).append(", ");
}
sb.delete(sb.length()-2, sb.length()); //마지막 콤마 (", ")는 지워준다.
sb.append("] / ").append(Integer.toString(min));
System.out.println(sb.toString());
```

maxList와 같은 방법으로 minList를 출력하였습니다

작은 과제 : 모든 단어의 단어 빈도수를 output.txt로 출력하기.

```
//모든 key, value를 파일에 출력한다.
keySet = map.keySet();
sb = new StringBuilder();
for(String key : keySet) {
    int val = map.get(key);
    sb.append(String.format("%s=%s", key, val));
    sb.append(", ");
}
sb.delete(sb.length()-2, sb.length()); //마지막 콤마 (", ")는 지워준다.
//자원 반환
pw.write(sb.toString());
pw.flush();
pw.close();
```

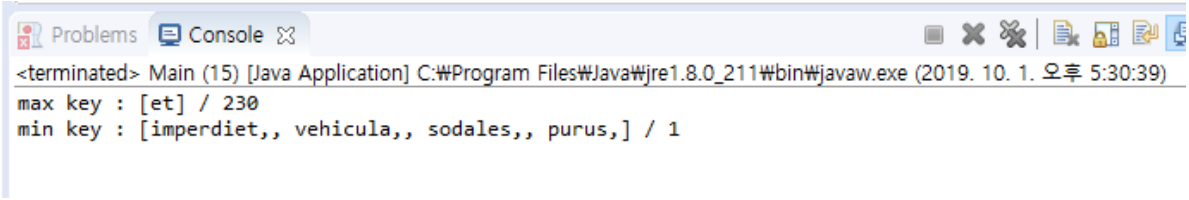
과제에서 모든 단어의 빈도수를 PrintWriter를 이용하여 output.txt에 기록할 것을 요구하므로, 앞에서와 같이 keySet() 함수를 활용해 map의 모든 원소를 순회한 뒤, StringBuilder에 key=value의 형태로 문자열을 합쳤습니다.

작은 과제 : 예외처리하기

```
public class Main {

    static final String FILE_NOT_FOUND = "파일을 찾을 수 없습니다!";
    static final String IO_EXCEPTION = "입출력에 문제가 있습니다!";

    public static void log(String msg) {
        System.out.println(msg);
    }


        } catch (FileNotFoundException e) {
            log(FILE_NOT_FOUND);
        } catch (IOException e) {
            log(IO_EXCEPTION);
        } finally {
            if(br != null) {
                try {
                    br.close();
                } catch (IOException e) {
                    log(IO_EXCEPTION);
                }
            }
        }
    }
}
```

코드 수행 도중 FileNotFoundException이 일어난 경우, FILE_NOT_FOUND에 해당하는 오류메시지를 출력하고, IOException가 일어난 경우, IO_EXCEPTION에 해당하는 오류메시지를 출력합니다.

#실행결과



<최대, 최솟값>

<모든 단어의 빈도 수>