1) How many lexemes does the following Java code contain?

```
1. public class CountDigits { 4
2.     public static void main(String[] args) { 11
3.           SimpleIO.prompt("Enter an integer: "); 7
4.           String userInput = SimpleIO.readLine(); 9
5.           int number = Integer.parseInt(userInput); 10
6.           int numDigits = 0; 5
7.           while (number > 0) { 7
8.                number /= 10; 5
9.                numDigits++; 4
10.          } 1
11.          System.out.println("The number " + userInput + " has " +
12.               numDigits + " digits"); 17
13.   } 1
14. } 1                                                          Total: 82
```

2)The following class contains several errors that violate the rules of Java:

```
1. class Thermometer {
2.      private int temperature      Missing Semicolon: Syntactic
3.      public Thermometer(int degrees) {
4.         temperature = degrees;
5.      }
6.      public Thermometer() {
7.         temperature = 0.0;        Different Type: Semantic
8.      }
9.      public void makeWarmer(int degrees) {
10.         temperature =+ degrees;
11.      }
12.      public void makeCooler(int degrees) {
13.         temperature -= degrees;
14.      }
15.    public getTemperature() {     No Return Type: Syntactic
16.         return temperature;
17.     }
18.     public string tostring() {      Should be String: Syntactic
19.         return temperature + " degrees';      Should be in Double Quotes: Lexical
20.     }
21. }
```

Describe each error and specify whether it is (a) lexical, (b) syntactic, or (c)semantic.

Use the numbers shown to identify the line on which each error occurs. The class may also contain programming errors that do not violate the rules of Java and will not be detected by a Java compiler. You should ignore these errors.

3) Write code in a language of your choice that checks a source file (input file in plain text format) that separates lexemes by white space and special characters. This lexical analyzer will only have tokens for special characters and alphanumeric strings.

Ie: 2345 6tgbsauhd9sa67*I{OPKDSl;jaklhl

Would be

2345

6tgbsauhd9sa67

*

I

{

OPKDSl

;

jaklhl

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

public class LexicalAnalyzer {

   public static void main(String[] args) throws IOException {

       BufferedReader br = new BufferedReader(new FileReader("le.txt"));
       String str = br.readLine();
       String specialCharacters = " !#$%&'()*+,-./:;<=>?@[]^_`{|}";

       while (str != null) {

           String[] split = str.split(" ");

           for (int i = 0 ; i < split.length ; i++) {

               String lex = split[i];
               int count = 0;

               while (count < lex.length()) {

                   if (specialCharacters.contains(Character.toString(lex.charAt(count)))) {

                       if (!specialCharacters.contains(Character.toString(lex.charAt(count-1)))){
                           System.out.println();
                       }
                       System.out.println(lex.charAt(count));
                   } else {
                       System.out.print(lex.charAt(count));
                   }

                   count++;
               }
               System.out.println();

           }

           str = br.readLine();

       }

   }

}
```