Next Gen Assignment 4 18343763

Problem 1

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

public class Stegano1Problem1 {
        /**
        * Constructor for objects of class Stegano1
        */
        public Stegano1Problem1()
        {
        }

        public static void main(String[] args) {
        String arg1, arg2, arg3, arg4;
        Boolean err = false;

        if (args != null && args.length > 1) { // Check for minimum number of arguments
        arg1 = args[0];
        arg2 = args[1];

        if (arg2 == "") {
                err = true;
        }
        else if ((arg1.charAt(0) == 65) && (args.length > 3)) {
                // Get other arguments
                arg3 = args[2];
                arg4 = args[3];
                if (arg3 == "" || arg4 == "") {
                err = true;
                }
                else {
                // Hide bitstring
                hide(arg2, arg3, arg4);
                }
        }
        else if (arg1.charAt(0) == 69){
                // Extract bitstring from text
                retrieve(arg2);
```

```
    }
    else {
        err = true;
    }
    }
    else {
    err = true;
    }

    if (err == true) {
    System.out.println();
    System.out.println("Use: Stegano1 <A:E><Input File><OutputFile><Bitstring>");
    System.out.println("Example: Stegano1 A inp.txt out.txt 0010101");
    System.out.println("Example: Stegano1 E inp.txt");

    }
    }

    static void hide(String inpFile, String outFile, String bitString) {
    BufferedReader reader;
    BufferedWriter writer;

    try {
    reader = new BufferedReader(new FileReader(inpFile));
    writer = new BufferedWriter(new FileWriter(outFile));
    String line = reader.readLine();
    int i = 0;
    while (line != null) {
        // Checking if it can check character otherwise function will return out of range
index errors if in any case the bitString wasnt even
        if (i < bitString.length()) {
        // Hiding the bits as spaces at the end of the line
        // Example, if the i = 0 in the bitString, two spaces will be added to the file.
        if (bitString.charAt(i) == 48) {
        line += "  ";
        } else if (bitString.charAt(i) == 49) {
        line += "   ";
        }
        }

        // Moving position forward to see the next character
        i++;

        // Attempts Made
    /*int i = 0;
    //System.out.printf("%d\n", bitString);
```

```
//System.out.printf("%d\n", );

while (line != null) {
    // Your code starts here
    //for (i = 0; i < bitString.length(); i++) {
    if (i < bitString.length()) {
    if (bitString.charAt(i) == 48) {
    line +=" ";
    } else if (bitString.charAt(i) == 49){
    line += "  ";
    }
    i++;*/
    // Store amended line in output file
    writer.write(line);
    writer.newLine();
    // read next line
    line = reader.readLine();
}
reader.close();
writer.close();
} catch (IOException e) {
e.printStackTrace();
}

}

static void retrieve(String inpFile) {
BufferedReader reader;
String bitString = "";
try {
reader = new BufferedReader(new FileReader(inpFile));
String line = reader.readLine();
while (line != null) {
    // Your code starts here

    // Checking if the line contains any of the following sequences & then adding that to bitString to print out later
    if (line.contains("   ")) bitString += "1";
    else if (line.contains("  ")) bitString += "0";

    // If none of the above sequences are found, the white loop exits so we do not keep checking the rest of the lines pointlessly
    else break;

    // Attempts made
    /*int i = line.length() - 1;
```

```
                boolean isSpace = false;
                if (line.charAt(i) == 32) {
                isSpace = true;
                if (line.charAt(i - 1) == 32) {
                bitString += "1";
                } else {
                bitString += "0";
                }
                }

                if (!isSpace) break;
                // System.out.println(line);*/
                // read next line
                line = reader.readLine();
        }
        System.out.print(bitString);
        reader.close();
        } catch (IOException e) {
        e.printStackTrace();
        }

        }

}
```

Problem 2

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

public class Stegano1Problem2 {
        /**
        * Constructor for objects of class Stegano1
        */
        public Stegano1Problem2()
        {
        }

        public static void main(String[] args) {
        String arg1, arg2, arg3, arg4;
        Boolean err = false;

        if (args != null && args.length > 1) { // Check for minimum number of arguments
        arg1 = args[0];
        arg2 = args[1];

        if (arg2 == "") {
                err = true;
        }
        else if ((arg1.charAt(0) == 65) && (args.length > 3)) {
                // Get other arguments
                arg3 = args[2];
                arg4 = args[3];
                if (arg3 == "" || arg4 == "") {
                err = true;
                }
                else {
                // Hide bitstring
                hide(arg2, arg3, arg4);
                }
        }
        else if (arg1.charAt(0) == 69){
                // Extract bitstring from text
                retrieve(arg2);
        }
        else {
                err = true;
```

```
    }
    }
    else {
    err = true;
    }

    if (err == true) {
    System.out.println();
    System.out.println("Use: Stegano1 <A:E><Input File><OutputFile><Bitstring>");
    System.out.println("Example: Stegano1 A inp.txt out.txt 0010101");
    System.out.println("Example: Stegano1 E inp.txt");

    }
    }

    static void hide(String inpFile, String outFile, String bitString) {
    //
    BufferedReader reader;
    BufferedWriter writer;

    try {
    reader = new BufferedReader(new FileReader(inpFile));
    writer = new BufferedWriter(new FileWriter(outFile));
    String line = reader.readLine();
    int i = 0;
    if ((bitString.length() % 2) != 0) bitString += "0";

    while (line != null) {
        // Your code starts here

        // Checking if it can check character ahead of itself otherwise function will
return out of range index errors if in any case the bitString wasnt even
        if (i + 1 < bitString.length()) {
        // Hiding the bits as spaces at the end of the line
        // Example, if the i = 0 & i + 1 = 1 in the bitString, three spaces will be added
to the file.
        if (bitString.charAt(i) == 48 && bitString.charAt(i + 1) == 48) {
        line += "  ";
        } else if (bitString.charAt(i) == 48 && bitString.charAt(i + 1) == 49) {
        line += "   ";
        } else if (bitString.charAt(i) == 49 && bitString.charAt(i + 1) == 48) {
        line += "            ";
        } else if (bitString.charAt(i) == 49 && bitString.charAt(i + 1) == 49) {
        line += "            ";

        }
        }

        // Moving position forward to see the next two characters
        i += 2;

        // Attempts made at solving the problem.
        /*if (i < bitString.length()) {
        if (bitString.charAt(i) == 48) {
        line += " ";
        } else if (bitString.charAt(i) == 49) {
        line += " ";
        }
        if (i + 1 < bitString.length()) {
        if (bitString.charAt(i + 1) == 48) {
            line += "\t";
        } else if (bitString.charAt(i + 1) == 49) {
            line += "\t\t";
        } else {
            System.out.printf("invalid char in bitString at pos %d", i);
        }
        }
        i += 2;
        }*/

        // Store amended line in output file
        writer.write(line);
        writer.newLine();
        // read next line
        line = reader.readLine();
    }
    reader.close();
    writer.close();
    } catch (IOException e) {
    e.printStackTrace();
    }

    }

    static void retrieve(String inpFile) {
    BufferedReader reader;
    String bitString = "";
    try {
    reader = new BufferedReader(new FileReader(inpFile));
    String line = reader.readLine();
    while (line != null) {
        // Your code starts here
```

```
        // Checking if the line contains any of the following sequences & then adding
that to bitString to print out later
        if (line.contains("          ")) bitString += "11";
        else if (line.contains(" ")) bitString += "10";
        else if (line.contains("   ")) bitString += "01";
        else if (line.contains("  ")) bitString += "00";

        // If none of the above sequences are found, the white loop exits so we do not
keep checking the rest of the lines pointlessly
        else break;

        // Attempts made at solving the problem.
        /*int i = line.length() - 1;
        if (i + 1 < bitString.length()) {
        if (bitString.charAt(i) == 48 && bitString.charAt(i) == 48) {
        line += " ";
        } else if (bitString.charAt(i) == 48 && bitString.charAt(i) == 49) {
        line += "  ";
        } else if (bitString.charAt(i) == 49 && bitString.charAt(i) == 48) {
        line += "   ";
        } else if (bitString.charAt(i) == 49 && bitString.charAt(i) == 49) {
        line += "          ";
        }
        }
        if (line.charAt(i) == 9) {
        if (line.charAt(i - 1) == 9) {
        bitString += "1";
        } else {
        bitString += "0";
        }
        }
        if (line.charAt(i - 1) == 32 || line.charAt(i - 3) == 32) {
        bitString += "0";
        } else if (line.charAt(i) == 32 || line.charAt(i - 2) == 32) {
        bitString += "1";
        }
        }*/

        line = reader.readLine();
    }

    System.out.println(bitString);

    reader.close();
    } catch (IOException e) {
        e.printStackTrace();
        }
        }
}
```