Name: Koiki Damilare Solomon

Matric No: 185887

Assignment: Write a program in any language that reads and detects mispelt keywords in another program of any language

Course: CSC 431

```
import java.io.FileInputStream;

import java.io.FileNotFoundException;

import java.io.IOException;

import java.util.logging.Level;

import java.util.logging.Logger;

import java.util.*;


public class KeywordChecker {


    private static ArrayList<String> keywords = new ArrayList<>();

    // keywords

    private static String[] keys={"try", "catch", "finally", "throw", "throws","import","for", "if", 
"else","switch", "case", "break"};


    public static void main(String[] args) {

        // TODO code application logic here

        try{

            // read file

            FileInputStream fi=new FileInputStream("C:\\Users\\Koiki 
Damilare\\Documents\\NetBeansProjects\\KeywordChecker\\src\\test_prog.java");

             int n=0;
```

```
try {

  // Array list that stores all read characters
  ArrayList<Character> word= new ArrayList();
  // tokens are read character by character
  while((n=fi.read())!=-1){
    char w=(char)n;
    word.add(w); // add character to arraylist
  }

  String newWord="";
  // loop through arraylist
  for(int j=0;j<word.size();j++){
    // If no empty char is encountered yet, append char to String newWord
    if(!" ".equals(String.valueOf(word.get(j)))){
      newWord+=String.valueOf(word.get(j));
    }else if(" ".equals(String.valueOf(word.get(j)))){ // else
      // if token is a keyword print
      if(tokenIsAKeyword(newWord)){
        printToken(newWord);
      }
      // if token is mispelt
      if(isAMispeltToken(newWord)){
        printToken(newWord);
      }
      // empty String newWord
```

```
                newWord="";
                // remove used tokens from arraylist
                for(int k=j;k<=0;k--){
                    word.remove(k);
                }
            }
        }
    } catch (IOException ex) {
        Logger.getLogger(KeywordChecker.class.getName()).log(Level.SEVERE, null, ex);
    }
    try {
        // close file
        fi.close();
    } catch (IOException ex) {
        Logger.getLogger(KeywordChecker.class.getName()).log(Level.SEVERE, null, ex);
    }
} catch (FileNotFoundException ex) {
    Logger.getLogger(KeywordChecker.class.getName()).log(Level.SEVERE, null, ex);
}

}
public static boolean tokenIsAKeyword(String token){
   keywords.addAll(Arrays.asList(keys));
   return keywords.contains(token);
}
public static void printToken(String token){
   String[] tokenArray1=token.split(";");
```

```
        for (String tokenArray11 : tokenArray1) {
            System.out.println(token);
        }
    }
    public static boolean isAMispeltToken(String token){
        // if token is not in kewords array
        for(String tk : keywords){
            // But its spelling matches that of some keywords
            if(!keywords.contains((token))){
                return (tk.regionMatches(0, token, 0, token.length()));
            }


        }
        return false;
    }


}
```