keyboardAnalyser.java

```java
// Comparing the efficiency of various keyboard layouts
// Based on user input, the amount of distance that needs to be travelled to press each key is calculated, provided all fingers are used
// The layouts in which less distance is travelled is most efficient.
// Currently there are qwerty, dvorak and colemak layouts which are compared.

import java.util.Scanner;

public class keyboardAnalyser {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter sentence: ");
        String sentence = sc.nextLine();
        sc.close();
        sentence = sentence.trim().concat(" ");
        System.out.println("Distance moved by fingers to type above sentence: ");
        System.out.println("\nQwerty: ");
        wordFingerDistance(sentence, "QWERTY");
        System.out.println("\nDvorak: ");
        wordDistance(sentence, "DVORAK");
        wordFingerDistance(sentence, "DVORAK");
        System.out.println("\nColemak: ");
        wordDistance(sentence, "COLEMAK");
        wordFingerDistance(sentence, "COLEMAK");
    }

    // Finding the finger required to press a given letter for all layouts
    static int findFingerQWERTY(char a) {
        if ("qaz".contains(Character.toString(a))) return 0;
        if ("wsx".contains(Character.toString(a))) return 1;
        if ("edc".contains(Character.toString(a))) return 2;
        if ("rfvtgb".contains(Character.toString(a))) return 3;
        if ("yhnujm".contains(Character.toString(a))) return 4;
        if ("ik".contains(Character.toString(a))) return 5;
        if ("ol".contains(Character.toString(a))) return 6;
        if ("p".contains(Character.toString(a))) return 7;
        else return 8;
    }

    static int findFingerDVORAK(char a) {
        if ("a".contains(Character.toString(a))) return 0;
        if ("oq".contains(Character.toString(a))) return 1;
        if ("ej".contains(Character.toString(a))) return 2;
        if ("pukyix".contains(Character.toString(a))) return 3;
        if ("fdbghm".contains(Character.toString(a))) return 4;
        if ("ctw".contains(Character.toString(a))) return 5;
        if ("rnv".contains(Character.toString(a))) return 6;
        if ("lsz".contains(Character.toString(a))) return 7;
        else return 8;
    }

    static int findFingerCOLEMAK(char a) {
        if ("qaz".contains(Character.toString(a))) return 0;
        if ("wrx".contains(Character.toString(a))) return 1;
        if ("fsc".contains(Character.toString(a))) return 2;
        if ("ptvgdb".contains(Character.toString(a))) return 3;
        if ("jhklne".contains(Character.toString(a))) return 4;
        if ("ue".contains(Character.toString(a))) return 5;
        if ("yi".contains(Character.toString(a))) return 6;
        if ("o".contains(Character.toString(a))) return 7;
        else return 8;
    }

    // Finding if the character is on the home row (finger doesn't to move to press it) or not on the home row (the finger has to move to press it)
    static int findDistanceQWERTY(int finger, int ch) {
        int distance = 0;
        if (finger == 0) {
            if (ch != 'a')
                distance += 1;
        }
        if (finger == 1) {
            if (ch != 's')
                distance += 1;
        }
        if (finger == 2) {
            if (ch != 'd')
                distance += 1;
        }
        if (finger == 3) {
            if (ch != 'f')
                distance += 1;
        }
        if (finger == 4) {
            if (ch != 'j') {
                distance += 1;
            }
        }
        if (finger == 5) {
            if (ch != 'k') {
                distance += 1;
            }
        }
        if (finger == 6) {
            if (ch != 'l') {
                distance += 1;
            }
        }
        if (finger == 7) {
```

```
            distance += 1;
        }
        return distance;
    }

    static int findDistanceDVORAK(int finger, int ch) {
        int distance = 0;
        if (finger == 1) {
            if (ch != 'o')
                distance += 1;
        }
        if (finger == 2) {
            if (ch != 'e')
                distance += 1;
        }
        if (finger == 3) {
            if (ch != 'u')
                distance += 1;
        }
        if (finger == 4) {
            if (ch != 'h') {
                distance += 1;
            }
        }
        if (finger == 5) {
            if (ch != 't') {
                distance += 1;
            }
        }
        if (finger == 6) {
            if (ch != 'n') {
                distance += 1;
            }
        }
        if (finger == 7) {
            if (ch != 's') {
                distance += 1;
            }
        }
        return distance;
    }

    static int findDistanceCOLEMAK(int finger, int ch) {
        int distance = 0;
        if (finger == 0) {
            if (ch != 'a')
                distance += 1;
        }
        if (finger == 1) {
            if (ch != 'r')
                distance += 1;
        }
        if (finger == 2) {
            if (ch != 's')
                distance += 1;
        }
        if (finger == 3) {
            if (ch != 't')
                distance += 1;
        }
        if (finger == 4) {
            if (ch != 'n')
                distance += 1;
        }
        if (finger == 5) {
            if (ch != 'e') {
                distance += 1;
            }
        }
        if (finger == 6) {
            if (ch != 'i')
                distance += 1;
        }
        return distance;
    }

    // Finding the total distance required by the fingers to press each key of the input
    static void wordDistance(String word, String layout) {
        int distance = 0;
        for (int i = 0; i < word.length(); i++) {
            char ch = word.charAt(i);
            if (layout.equals("QWERTY"))
                distance += findDistanceQWERTY(findFingerQWERTY(ch), ch);
            if (layout.equals("DVORAK"))
                distance += findDistanceDVORAK(findFingerDVORAK(ch), ch);
            if (layout.equals("COLEMAK"))
                distance += findDistanceCOLEMAK(findFingerCOLEMAK(ch), ch);
        }
        System.out.println(distance);
    }

    // Finding the distance moved by each finger separately to press each key and then the average of each
    static void wordFingerDistance(String word, String layout) {
        int[] distance = {0, 0, 0, 0, 0, 0, 0, 0, 0};
        for (int i = 0; i < word.length(); i++) {
            char ch = word.charAt(i);
            if (layout.equals("QWERTY"))
                distance[findFingerQWERTY(ch)] += findDistanceQWERTY(findFingerQWERTY(ch), ch);
            if (layout.equals("DVORAK"))
                distance[findFingerDVORAK(ch)] += findDistanceDVORAK(findFingerDVORAK(ch), ch);
```

```
            if (layout.equals("COLEMAK"))
                distance[findFingerCOLEMAK(ch)] += findDistanceCOLEMAK(findFingerCOLEMAK(ch), ch);
        }
        System.out.println("Layout: " + layout);
        int avDistance = 0;
        for (int i = 0; i < distance.length; i++) {
            System.out.println("Finger " + (i + 1) + ": " + distance[i]);
            avDistance += distance[i];
        }
        // Sum and Average of all fingers
        System.out.println("Total distance: " + avDistance);
        System.out.println("Average of all fingers: " + (avDistance / distance.length));
    }
}
```