AARYAN BAIRAGI
BE-47004

ISR LAB 2:

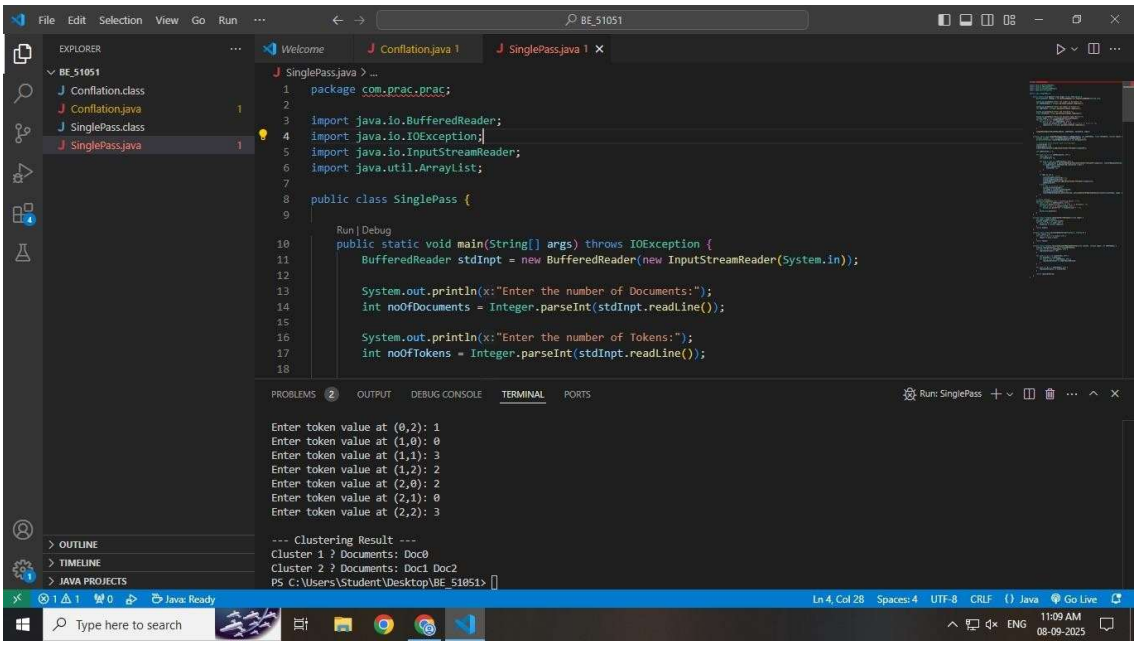
```
package com.prac.prac;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.ArrayList;

public class SinglePass {

    public static void main(String[] args) throws IOException {
        BufferedReader stdInpt = new BufferedReader(new InputStreamReader(System.in));

        System.out.println("Enter the number of Documents:");
        int noOfDocuments = Integer.parseInt(stdInpt.readLine());

        System.out.println("Enter the number of Tokens:");
        int noOfTokens = Integer.parseInt(stdInpt.readLine());

        System.out.println("Enter the Threshold:");
        float threshold = Float.parseFloat(stdInpt.readLine());

        System.out.println("Enter the Document Token Matrix:");
        int[][] input = new int[noOfDocuments][noOfTokens];
        for (int i = 0; i < noOfDocuments; i++) {
            for (int j = 0; j < noOfTokens; j++) {
                System.out.print("Enter token value at (" + i + "," + j + "): ");
                input[i][j] = Integer.parseInt(stdInpt.readLine());
            }
        }
```

```
        singlePassAlgorithm(noOfDocuments, noOfTokens, threshold, input);
    }

    private static void singlePassAlgorithm(int noOfDocuments, int noOfTokens, float threshold, int[][]
input) {
        int[][] cluster = new int[noOfDocuments][noOfDocuments + 1];
        ArrayList<Float[]> clusterRepresentative = new ArrayList<>();

        // Initialize first cluster with first document
        cluster[0][0] = 1;
        cluster[0][1] = 0;
        clusterRepresentative.add(convertIntArrToFloatArr(input[0]));

        int noOfClusters = 1;

        for (int i = 1; i < noOfDocuments; i++) {
            float max = -1;
            int clusterId = -1;

            for (int j = 0; j < noOfClusters; j++) {
                float similarity = calculateSimilarity(convertIntArrToFloatArr(input[i]),
clusterRepresentative.get(j));
                if (similarity > threshold && similarity > max) {
                    max = similarity;
                    clusterId = j;
                }
            }

            if (max == -1) {
                // Create new cluster
                cluster[noOfClusters][0] = 1;
                cluster[noOfClusters][1] = i;
```

```
            clusterRepresentative.add(convertIntArrToFloatArr(input[i]));
            noOfClusters++;
        } else {
            // Add to existing cluster
            cluster[clusterId][0]++;
            int index = cluster[clusterId][0];
            cluster[clusterId][index] = i;
            clusterRepresentative.set(clusterId, calculateClusterRepresentative(cluster[clusterId], input,
noOfTokens));
        }
    }

    // Print clusters
    System.out.println("\n--- Clustering Result ---");
    for (int i = 0; i < noOfClusters; i++) {
        System.out.print("Cluster " + (i + 1) + " → Documents: ");
        for (int j = 1; j <= cluster[i][0]; j++) {
            System.out.print("Doc" + cluster[i][j] + " ");
        }
        System.out.println();
    }
}

private static Float[] convertIntArrToFloatArr(int[] input) {
    int size = input.length;
    Float[] answer = new Float[size];
    for (int i = 0; i < size; i++) {
        answer[i] = (float) input[i];
    }
    return answer;
}
```

```
    private static float calculateSimilarity(Float[] a, Float[] b) {
        float result = 0;
        for (int i = 0; i < a.length; i++) {
            result += a[i] * b[i];
        }
        return result;
    }

    private static Float[] calculateClusterRepresentative(int[] cluster, int[][] input, int noOfTokens) {
        Float[] representative = new Float[noOfTokens];
        for (int i = 0; i < noOfTokens; i++) {
            representative[i] = 0f;
        }

        for (int i = 1; i <= cluster[0]; i++) {
            int docIndex = cluster[i];
            for (int j = 0; j < noOfTokens; j++) {
                representative[j] += input[docIndex][j];
            }
        }

        for (int i = 0; i < noOfTokens; i++) {
            representative[i] /= cluster[0];
        }

        return representative;
    }
}
```

OUTPUT:


File Edit Selection View Go Run
BE_51051
EXPLORER
BE_51051
Conflation.class
Conflation.java 1
SinglePass.class
SinglePass.java 1
Welcome
Conflation.java 1
SinglePass.java 1
SinglePass.java > ...
1 package com.prac.prac;
2
3 import java.io.BufferedReader;
4 import java.io.IOException;
5 import java.io.InputStreamReader;
6 import java.util.ArrayList;
7
8 public class SinglePass {
9
Run | Debug
10 public static void main(String[] args) throws IOException {
11 BufferedReader stdInpt = new BufferedReader(new InputStreamReader(System.in));
12
13 System.out.println(x:"Enter the number of Documents:");
14 int noOfDocuments = Integer.parseInt(stdInpt.readLine());
15
16 System.out.println(x:"Enter the number of Tokens:");
17 int noOfTokens = Integer.parseInt(stdInpt.readLine());
18
PROBLEMS 2
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
Run: SinglePass
Enter token value at (0,2): 1
Enter token value at (1,0): 0
Enter token value at (1,1): 3
Enter token value at (1,2): 2
Enter token value at (2,0): 2
Enter token value at (2,1): 0
Enter token value at (2,2): 3
--- Clustering Result ---
Cluster 1 ? Documents: Doc0
Cluster 2 ? Documents: Doc1 Doc2
PS C:\Users\Student\Desktop\BE_51051>
OUTLINE
TIMELINE
JAVA PROJECTS
Java: Ready
Ln 4, Col 28
Spaces: 4
UTF-8
CRLF
Java
Go Live
Type here to search
ENG
11:09 AM
08-09-2025