Practical No - 2

Title : Implement Single-pass Algorithm for clustering of files.(Consider 4to 5 files).

```
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 stdInput = new BufferedReader(new InputStreamReader(System.in));


      System.out.println("Enter the number of Documents:");

int noOfDocuments = Integer.parseInt(stdInput.readLine());


      System.out.println("Enter the number of Tokens:");

int noOfTokens = Integer.parseInt(stdInput.readLine());


      System.out.println("Enter the threshold:");        float

threshold = Float.parseFloat(stdInput.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 (" + i + ", " + j + "): ");

input[i][j] = Integer.parseInt(stdInput.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<>();

        cluster[0][0] = 1; // Number of documents in the first
cluster        cluster[0][1] = 0; // Index of the first document
int noOfClusters = 1;

        Float[] temp = convertIntArrToFloatArr(input[0]);
clusterRepresentative.add(temp);

        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) {
if (similarity > max) {
max = similarity;
clusterId = j;
                    }
                }
            }

            if (max == -1) {
                cluster[noOfClusters][0] = 1; // New cluster
cluster[noOfClusters][1] = i; // Index of the first document in the new cluster
```

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

      // Output clusters        for (int i = 0; i <
noOfClusters; i++) {
System.out.print("Cluster " + i + ": ");
for (int j = 1; j <= cluster[i][0]; j++) {
            System.out.print(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;
   }
```

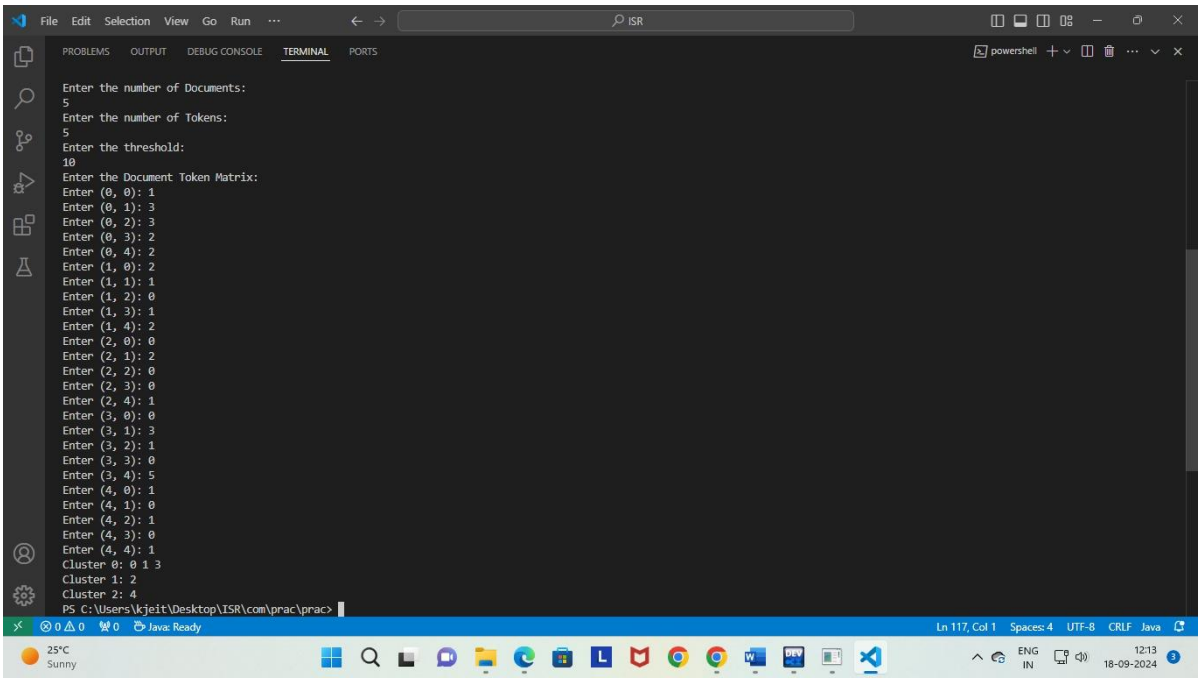
```
   private static float calculateSimilarity(Float[] a, Float[] b)
{       float answer = 0;         for (int i = 0; i < a.length; i++) {
answer += a[i] * b[i];
      }
      return answer;
   }

   private static Float[] calculateClusterRepresentative(int[] cluster, int[][] input, int noOfTokens) {
Float[] answer = new Float[noOfTokens];
      for (int i = 0; i < noOfTokens; i++) {
answer[i] = 0.0f; // Initialize to 0
      }

      for (int i = 1; i <= cluster[0]; i++) {           for (int j = 0; j <
noOfTokens; j++) {              answer[j] += input[cluster[i]][j]; //
Sum up token values
         }
      }

      for (int i = 0; i < noOfTokens; i++) {           answer[i]
/= cluster[0]; // Average the token values
      }
      return answer;
   }
}
```

OUTPUT :


File Edit Selection View Go Run
ISR
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
powershell
Enter the number of Documents:
5
Enter the number of Tokens:
5
Enter the threshold:
10
Enter the Document Token Matrix:
Enter (0, 0): 1
Enter (0, 1): 3
Enter (0, 2): 3
Enter (0, 3): 2
Enter (0, 4): 2
Enter (1, 0): 2
Enter (1, 1): 1
Enter (1, 2): 0
Enter (1, 3): 1
Enter (1, 4): 2
Enter (2, 0): 0
Enter (2, 1): 2
Enter (2, 2): 0
Enter (2, 3): 0
Enter (2, 4): 1
Enter (3, 0): 0
Enter (3, 1): 3
Enter (3, 2): 1
Enter (3, 3): 0
Enter (3, 4): 5
Enter (4, 0): 1
Enter (4, 1): 0
Enter (4, 2): 1
Enter (4, 3): 0
Enter (4, 4): 1
Cluster 0: 0 1 3
Cluster 1: 2
Cluster 2: 4
PS C:\Users\kjeit\Desktop\ISR\com\prac\prac>
Java: Ready
Ln 117, Col 1 Spaces: 4 UTF-8 CRLF Java
25°C
Sunny
ENG
IN
12:13
18-09-2024