Assignment No. 02

Code:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

typedef struct cluster
{
	char cnm[10];
	float terms[10];
	int element;
}

cluster;
void display(int doc_term[10][10], int no1, int no2)		//display Document Term Matrix
{
	int i, j;
	printf("Document Term Matrix is : \n");
	for (j = 0; j < no2; j++)
		printf("\tT%d", (j + 1));
	printf("\n");
	for (i = 0; i < no1; i++)	//Number of documents
	{
		printf("Doc%d\t", (i + 1));
		for (j = 0; j < no2; j++)	//Number of terms
		{
			printf("%d\t", doc_term[i][j]);
		}

		printf("\n");
	}
}

float similarity(float *arr1, int *arr2, int size)
{
	//Calculate similarity
	int i = 0;
	float sum = 0;
	for (i = 0; i < size; i++)
		sum = sum + (arr1[i] *arr2[i]);
	return sum;
}

void display_cluster(cluster c[10][10], int clust_count, int no2)	//display clusters
{
	int i, j;
	for (i = 0; i < clust_count; i++)
	{
		printf("\nCluster C%d={", (i + 1));
		for (j = 1; j <= c[i][0].element; j++)
			printf(" %s, ", c[i][j].cnm);
		printf("}");
```

```
                printf("\nCentroid of C%d =<", (i + 1));
                for (j = 0; j < no2; j++)
                        printf("%.2f, ", c[i][0].terms[j]);
                printf(" >");
        }

        printf("\n\nThe total number of clusters are %d", clust_count);
}

void create_cluster(int doc_term[10][10], int no1, int no2)
{
        cluster c[10][10];
        char str[5], str2[3];
        int i, j, m, n, threshold, cluster_count, count, temp[10];
        float sim, sum;
        printf("\nEnter the threshold value ");
        scanf("%d", &threshold);
        cluster_count = 0;
        for (i = 0; i < no1; i++)     //Number of documents
        {
                strcpy(str, "Doc");
                sprintf(str2, "%d", (i + 1));
                strcat(str, str2);
                if (cluster_count == 0)    //initially there are no cluster
                {
                        m = 0;
                        n = 0;
                        c[m][n].element = 1;
                        n++;
                        strcpy(c[m][n].cnm, str);
                        for (j = 0; j < no2; j++)     //fetch first row
                                c[m][n].terms[j] = doc_term[i][j];
                        cluster_count++;
                        for (j = 0; j < no2; j++)     //to calculate cluster representative
                        {
                                sum = 0;
                                count = 0;
                                for (n = 1; n <= c[m][0].element; n++)
                                {
                                        sum = sum + c[m][n].terms[j];
                                        count++;
                                }

                                c[m][0].terms[j] = sum / count;  //calculate average
                        }
                }
                else
                {
                        for (m = 0; m < cluster_count; m++)        //iterate for every cluster
                        {
                                for (j = 0; j < no2; j++)
                                        temp[j] = doc_term[i][j];
                                printf("Doc%d - ", (i + 1));
```

```
                        for (j = 0; j < no2; j++)
                                printf("%d ", temp[j]);
                        sim = similarity(c[m][0].terms, temp, no2);         //calculate similarity
                        printf("\nSimilarity(Doc%d, C%d) - %f", (i + 1), (m + 1), sim);
                        printf("\n");
                        if (sim > threshold)        //condition to belong to cluster
                        {
                                c[m][0].element++;
                                n = c[m][0].element;
                                strcpy(c[m][n].cnm, str);
                                for (j = 0; j < no2; j++)
                                        c[m][n].terms[j] = doc_term[i][j];
                                //calculate cluster representative using mean
                                for (j = 0; j < no2; j++)
                                {
                                        sum = 0;
                                        count = 0;
                                        for (n = 1; n <= c[m][0].element; n++)
                                        {
                                                sum = sum + c[m][n].terms[j];
                                                count++;
                                        }

                                        c[m][0].terms[j] = sum / count;
                                }

                                break;
                        }       //if
                }       //for
                if (m == cluster_count)  //if it doesnot belong to any cluster
                {
                        cluster_count++;
                        c[m][0].element = 1;
                        n = c[m][0].element;
                        strcpy(c[m][n].cnm, str);
                        for (j = 0; j < no2; j++)
                                c[m][n].terms[j] = doc_term[i][j];
                        for (j = 0; j < no2; j++)    //calculate cluster representative using mean
                        {
                                sum = 0;
                                count = 0;
                                for (n = 1; n <= c[m][0].element; n++)
                                {
                                        sum = sum + c[m][n].terms[j];
                                        count++;
                                }

                                c[m][0].terms[j] = sum / count;
                        }
                }       //if
        }       //else
}       //for
display_cluster(c, cluster_count, no2);
```

```
}

void main()
{
	int i, j, no1, no2, doc_term[10][10];
	printf("\nEnter the number of documents ");
	scanf("%d", &no1);
	printf("\nEnter the number of terms ");
	scanf("%d", &no2);
	for (i = 0; i < no1; i++)
	{
		for (j = 0; j < no2; j++)
		{
			printf("\nEnter the values of Doc%d X T%d ", (i + 1), (j + 1));
			scanf("%d", &doc_term[i][j]);
		}
	}

	display(doc_term, no1, no2);
	create_cluster(doc_term, no1, no2);
}
```

Output 1:

```
ubuntu@ubuntu:~/Desktop$ gcc clust.c
ubuntu@ubuntu:~/Desktop$ ./a.out

Enter the number of documents 4
Enter the number of terms 4
Enter the values of Doc1 X T1 1
Enter the values of Doc1 X T2 2
Enter the values of Doc1 X T3 0
Enter the values of Doc1 X T4 0
Enter the values of Doc2 X T1 3
Enter the values of Doc2 X T2 1
Enter the values of Doc2 X T3 2
Enter the values of Doc2 X T4 3
Enter the values of Doc3 X T1 3
Enter the values of Doc3 X T2 0
Enter the values of Doc3 X T3 0
Enter the values of Doc3 X T4 1
Enter the values of Doc4 X T1 2
Enter the values of Doc4 X T2 1
Enter the values of Doc4 X T3 0
Enter the values of Doc4 X T4 3
Document Term Matrix is :
        T1      T2      T3      T4
Doc1    1       2       0       0
Doc2    3       1       2       3
Doc3    3       0       0       1
Doc4    2       1       0       3

Enter the threshold value 10
Doc2 - 3 1 2 3
Similarity(Doc2, C1) - 5.000000
Doc3 - 3 0 0 1
Similarity(Doc3, C1) - 3.000000
Doc3 - 3 0 0 1
Similarity(Doc3, C2) - 12.000000
Doc4 - 2 1 0 3
Similarity(Doc4, C1) - 4.000000
Doc4 - 2 1 0 3
Similarity(Doc4, C2) - 12.500000

Cluster C1={ Doc1, }
Centroid of C1 = < 1.00, 2.00, 0.00, 0.00, >
Cluster C2={ Doc2, Doc3, Doc4, }
Centroid of C2 = < 2.67, 0.67, 0.67, 2.33, >

The total number of clusters are 2
```

Output 2:

```
ubuntu@ubuntu:~/Desktop$ gcc clust.c
```

```
ubuntu@ubuntu:~/Desktop$ gcc clust.c
ubuntu@ubuntu:~/Desktop$ ./a.out

Enter the number of documents 6
Enter the number of terms 6
Enter the values of Doc1 X T1 4
Enter the values of Doc1 X T2 3
Enter the values of Doc1 X T3 6
Enter the values of Doc1 X T4 1
Enter the values of Doc1 X T5 1
Enter the values of Doc1 X T6 5
Enter the values of Doc2 X T1 2
Enter the values of Doc2 X T2 6
Enter the values of Doc2 X T3 2
Enter the values of Doc2 X T4 0
Enter the values of Doc2 X T5 0
Enter the values of Doc2 X T6 1
Enter the values of Doc3 X T1 2
Enter the values of Doc3 X T2 5
Enter the values of Doc3 X T3 3
Enter the values of Doc3 X T4 7
Enter the values of Doc3 X T5 6
Enter the values of Doc3 X T6 0
Enter the values of Doc4 X T1 1
Enter the values of Doc4 X T2 2
Enter the values of Doc4 X T3 6
Enter the values of Doc4 X T4 4
Enter the values of Doc4 X T5 4
Enter the values of Doc4 X T6 7
Enter the values of Doc5 X T1 2
Enter the values of Doc5 X T2 0
Enter the values of Doc5 X T3 6
Enter the values of Doc5 X T4 4
Enter the values of Doc5 X T5 7
Enter the values of Doc5 X T6 2
Enter the values of Doc6 X T1 3
Enter the values of Doc6 X T2 7
Enter the values of Doc6 X T3 1
Enter the values of Doc6 X T4 5
Enter the values of Doc6 X T5 0
Enter the values of Doc6 X T6 6

Document Term Matrix is:
```

	T1	T2	T3	T4	T5	T6
Doc1	4	3	6	1	1	5
Doc2	2	6	2	0	0	1
Doc3	2	5	3	7	6	0
Doc4	1	2	6	4	4	7
Doc5	2	0	6	4	7	2
Doc6	3	7	1	5	0	6

```
Enter the threshold value 60
Doc2 - 2 6 2 0 0 1
```

Similarity(Doc2, C1) - 43.000000
Doc3 - 2 5 3 7 6 0
Similarity(Doc3, C1) - 54.000000
Doc3 - 2 5 3 7 6 0
Similarity(Doc3, C2) - 40.000000
Doc4 - 1 2 6 4 4 7
Similarity(Doc4, C1) - 89.000000
Doc5 - 2 0 6 4 7 2
Similarity(Doc5, C1) - 80.500000
Doc6 - 3 7 1 5 0 6
Similarity(Doc6, C1) - 67.666664

Cluster C1={ Doc1, Doc4, Doc5, Doc6, }
Centroid of C1 = < 2.50, 3.00, 4.75, 3.50, 3.00, 5.00, >
Cluster C2={ Doc2, }
Centroid of C2 = < 2.00, 6.00, 2.00, 0.00, 0.00, 1.00, >
Cluster C3={ Doc3, }
Centroid of C3 = < 2.00, 5.00, 3.00, 7.00, 6.00, 0.00, >

The total number of clusters are 3