DSA ASSIGNMENT-1

TANISHA KARMAKAR

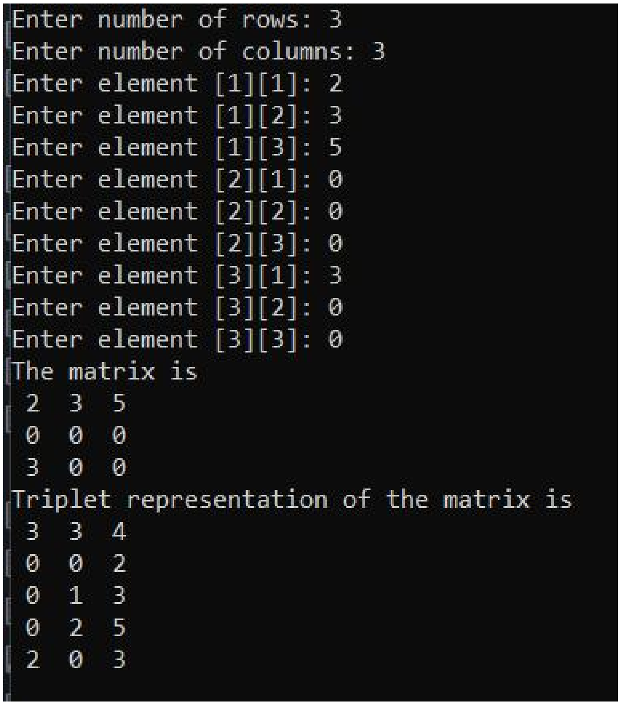
21051950

CSE 37

Q1. WAP for entering a sparse matrix and covert in triplet format.

```
#include<stdio.h>
int main()
{
int M[100][100], m, n, k=1, size=0;
printf("Enter number of rows: ");
scanf("%d", &m);
printf("Enter number of columns: ");
scanf("%d", &n);
for(int i=0; i<m; i++){
  for(int j=0; j<n; j++){
    printf("Enter element [%d][%d]: ", i+1, j+1);
    scanf("%d", &M[i][j]);
    if (M[i][j] != 0)
      size++;
  }
}
    printf("The matrix is \n");
    for (int i = 0; i < m; i++) {
        for (int j = 0; j < n; j++) {
            printf(" %d ",M[i][j]);

        }
        printf("\n");
    }
int T[size+1][3];
T[0][0]=m;
T[0][1]=n;
T[0][2]=size;

for(int i=0; i<m; i++){
 for(int j=0; j<n; j++){
  if (M[i][j]!=0)
  {
   T[k][0]=i;
   T[k][1]=j;
   T[k][2]=M[i][j];
   k++;

  }
 }
}

  printf("Triplet representation of the matrix is \n");
  for (int i=0; i<size+1; i++)
  {
    for (int j=0; j<3; j++)
      printf(" %d ", T[i][j]);
    printf("\n");
  }

  return 0;
}
```

Output:



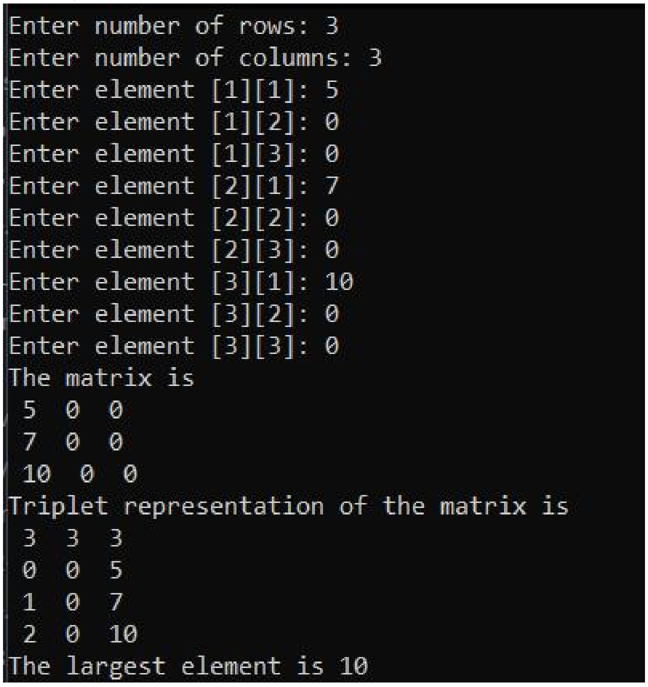
```
Enter number of rows: 3
Enter number of columns: 3
Enter element [1][1]: 2
Enter element [1][2]: 3
Enter element [1][3]: 5
Enter element [2][1]: 0
Enter element [2][2]: 0
Enter element [2][3]: 0
Enter element [3][1]: 3
Enter element [3][2]: 0
Enter element [3][3]: 0
The matrix is
 2  3  5
 0  0  0
 3  0  0
Triplet representation of the matrix is
 3  3  4
 0  0  2
 0  1  3
 0  2  5
 2  0  3
```

Q2. WAP to find the largest element of a sparse matrix using triplet format.

```
#include<stdio.h>
int main()
{
int M[100][100], m, n, k=1, size=0, largest;
printf("Enter number of rows: ");
scanf("%d", &m);
printf("Enter number of columns: ");
scanf("%d", &n);
for(int i=0; i<m; i++){
  for(int j=0; j<n; j++){
    printf("Enter element [%d][%d]: ", i+1, j+1);
    scanf("%d", &M[i][j]);
    if (M[i][j] != 0)
      size++;
  }
}
```

```
    printf("The matrix is \n");
    for (int i = 0; i < m; i++) {
        for (int j = 0; j < n; j++) {
            printf(" %d ",M[i][j]);

        }
        printf("\n");
    }
int T[size+1][3];
T[0][0]=m;
T[0][1]=n;
T[0][2]=size;

for(int i=0; i<m; i++){
  for(int j=0; j<n; j++){
    if (M[i][j]!=0)
    {
      T[k][0]=i;
      T[k][1]=j;
      T[k][2]=M[i][j];
      k++;

    }
  }
}

    printf("Triplet representation of the matrix is \n");
    for (int i=0; i<size+1; i++)
    {
        for (int j=0; j<3; j++)
            printf(" %d ", T[i][j]);
        printf("\n");
    }

for (m=1; m<(size+1); m++){
  if (T[m][2]>T[m+1][2]){
    largest=T[m][2];
  }
}
printf("The largest element is %d", largest);
    return 0;
}
```

Output:



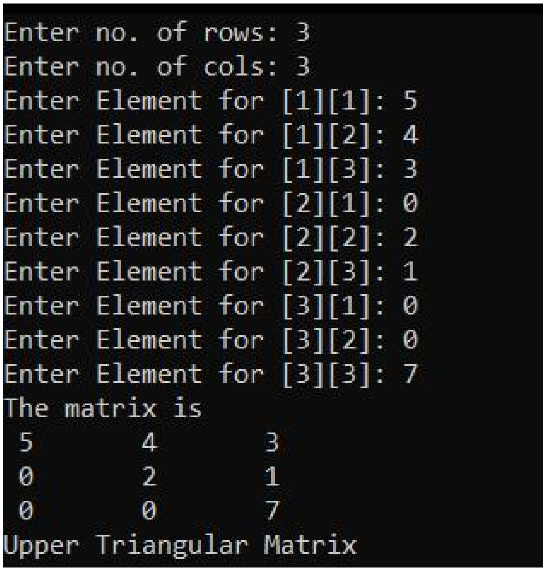
```
Enter number of rows: 3
Enter number of columns: 3
Enter element [1][1]: 5
Enter element [1][2]: 0
Enter element [1][3]: 0
Enter element [2][1]: 7
Enter element [2][2]: 0
Enter element [2][3]: 0
Enter element [3][1]: 10
Enter element [3][2]: 0
Enter element [3][3]: 0
The matrix is
 5  0  0
 7  0  0
 10  0  0
Triplet representation of the matrix is
 3  3  3
 0  0  5
 1  0  7
 2  0  10
The largest element is 10
```

Q3. WAP to check if a matrix is lower triangular or upper triangular matrix.

```
#include<stdio.h>
int main()
{
int m[100][100], r, c, flag=0, temp2=0, temp3=0;
printf("Enter no. of rows: ");
scanf("%d", &r);
printf("Enter no. of cols: ");
scanf("%d", &c);

for( int i=0; i<r; i++){
 for( int j=0; j<c; j++){
  printf("Enter Element for [%d][%d]: ", i+1, j+1);
  scanf("%d", &m[i][j]);
 }
}
printf("The matrix is \n");
for( int i = 0; i < r; i++) {
  for (int j = 0; j < c; j++) {
    printf(" %d \t",m[i][j]);
    if (i==j && m[i][j]==0)
          flag=1;
  }
  printf("\n");
}
if (r==c)
  {
  for (int i=0;i<r;i++)
  {
    for (int j=0;j<c;j++)
    {
      if (flag==1)
      printf("Wrong Input");
      else
      {
        if(i>j && m[i][j]!=0)
        temp2++;
        else if(j>i && m[i][j]!=0)
        temp3++;
      }
    }
  }

  if (temp2==0 && temp3==0)
  printf("Both Upper and Lower Triangular Matrix");
  else if(temp2==0)
```

```
    printf("Upper Triangular Matrix");
    else if(temp3==0)
    printf("Lower Triangular Matrix");
    else
    printf("Not of any type");
    }

    else
    printf("Wrong Input");

    return 0;
}
```

Output:



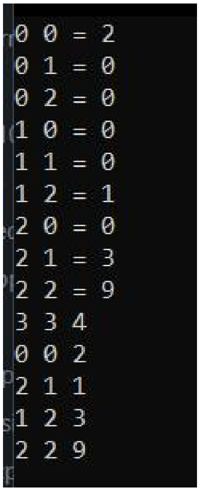
```
Enter no. of rows: 3
Enter no. of cols: 3
Enter Element for [1][1]: 5
Enter Element for [1][2]: 4
Enter Element for [1][3]: 3
Enter Element for [2][1]: 0
Enter Element for [2][2]: 2
Enter Element for [2][3]: 1
Enter Element for [3][1]: 0
Enter Element for [3][2]: 0
Enter Element for [3][3]: 7
The matrix is
 5       4       3
 0       2       1
 0       0       7
Upper Triangular Matrix
```

Q4. WAP for finding the transpose of a sparse matrix using triplet format.

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

int main(){
    int l = 0, n = 3;

    // int sparse[3][3] = {
    //     {2, 0, 0},
    //    {0, 0, 1},
    //    {0, 3, 9}
    // };

    int sparse[3][3];

    for(int i = 0; i < n; i++){
        for(int j = 0; j < n; j++){
            printf("%d %d = ", i, j);
            scanf("%d", &sparse[i][j]);
            if(sparse[i][j] != 0)
                l++;
        }
    }

    int triplet[l+1][3];

    int k = 1;
    triplet[0][0] = 3;
    triplet[0][1] = 3;
    triplet[0][2] = l;

    for(int i = 0; i < n; i++){
        for(int j = 0; j < n; j++){
            if(sparse[i][j] != 0){
                triplet[k][0] = i;
                triplet[k][1] = j;
                triplet[k][2] = sparse[i][j];
                k++;
            }
        }
    }

    int transpose[l+1][3];
```

```
    transpose[0][0] = 3;
    transpose[0][1] = 3;
    transpose[0][2] = l;

    for(int i = 1; i < l+1; i++){
        transpose[i][0] = triplet[i][1];
        transpose[i][1] = triplet[i][0];
        transpose[i][2] = triplet[i][2];
    }

    for(int i = 0; i < l+1; i++){
        for(int j = 0; j < 3; j++){
            printf("%d ", transpose[i][j]);
        }
        printf("\n");
    }



    return 0;
}
```

Output:



```
0 0 = 2
0 1 = 0
0 2 = 0
1 0 = 0
1 1 = 0
1 2 = 1
2 0 = 0
2 1 = 3
2 2 = 9
3 3 4
0 0 2
2 1 1
1 2 3
2 2 9
```

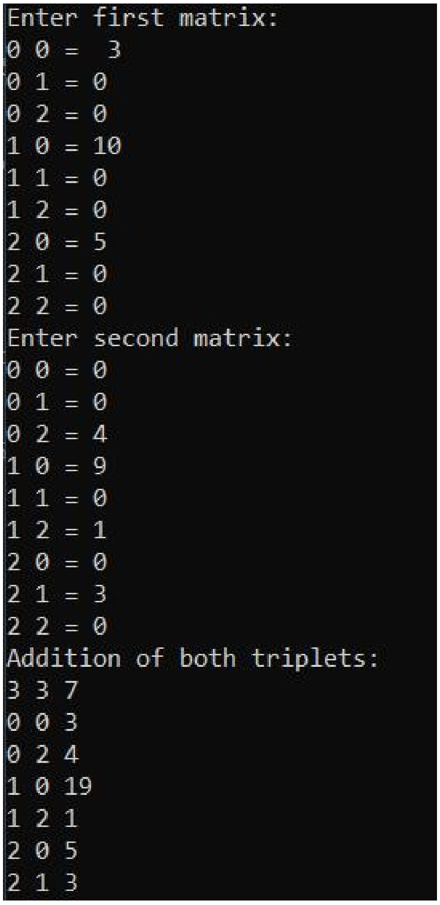
Q5. WAP for addition of two sparse matrix using triplet format.

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

int main(){
   int l1 = 0, l2 = 0, n = 3;

   // int l1 = 4, l2 = 4, n = 3;
   // int sparse1[3][3] = {
   //     {2, 0, 0},
   //     {0, 0, 1},
   //     {0, 3, 9}
   // };
   // int sparse2[3][3] = {
   //     {2, 0, 0},
   //     {0, 0, 1},
   //     {3, 3, 0}
   // };

   int sparse1[3][3];
   printf("Enter first matrix: \n");
   for(int i = 0; i < n; i++){
      for(int j = 0; j < n; j++){
         printf("%d %d = ", i, j);
         scanf("%d", &sparse1[i][j]);
         if(sparse1[i][j] != 0)
            l1++;
      }
   }

   int triplet1[l1+1][3];

   int k = 1;
   triplet1[0][0] = 3;
```

```
triplet1[0][1] = 3;
triplet1[0][2] = l1;

for(int i = 0; i < n; i++){
    for(int j = 0; j < n; j++){
        if(sparse1[i][j] != 0){
            triplet1[k][0] = i;
            triplet1[k][1] = j;
            triplet1[k][2] = sparse1[i][j];
            k++;
        }
    }
}

printf("Enter second matrix: \n");
int sparse2[3][3];

for(int i = 0; i < n; i++){
    for(int j = 0; j < n; j++){
        printf("%d %d = ", i, j);
        scanf("%d", &sparse2[i][j]);
        if(sparse2[i][j] != 0)
            l2++;
    }
}

int triplet2[l2+1][3];

k = 1;
triplet2[0][0] = 3;
triplet2[0][1] = 3;
triplet2[0][2] = l2;

for(int i = 0; i < n; i++){
    for(int j = 0; j < n; j++){
        if(sparse2[i][j] != 0){
            triplet2[k][0] = i;
            triplet2[k][1] = j;
            triplet2[k][2] = sparse2[i][j];
            k++;
        }
    }
}


int triplet[l1+l2+1][3];

k = 1;
triplet[0][0] = 3;
triplet[0][1] = 3;
triplet[0][2] = l1+l2;


int i = 1, j = 1, l = 1;
while((i < l1+1) || (j < l2+1)){

    // printf("%d %d \n", i, j);

    if(triplet1[i][0] < triplet2[j][0]){
        triplet[l][0] = triplet1[i][0];
        triplet[l][1] = triplet1[i][1];
        triplet[l][2] = triplet1[i][2];
        l++;
        i++;
    }
    else if(triplet1[i][0] > triplet2[j][0]){
        triplet[l][0] = triplet2[j][0];
        triplet[l][1] = triplet2[j][1];
        triplet[l][2] = triplet2[j][2];
        l++;
        j++;
    }
    else if(triplet1[i][0] == triplet2[j][0]){
        if(triplet1[i][1] == triplet2[j][1]){
            triplet[l][0] = triplet1[i][0];
            triplet[l][1] = triplet1[i][1];
            triplet[l][2] = triplet1[i][2] + triplet2[j][2];
            l++;
            i++;
            j++;
        }
        else if(triplet1[i][1] < triplet2[j][1]){
            triplet[l][0] = triplet1[i][0];
            triplet[l][1] = triplet1[i][1];
            triplet[l][2] = triplet1[i][2];
            l++;
            i++;
        }
        else if(triplet1[i][1] > triplet2[j][1]){
            triplet[l][0] = triplet2[i][0];
            triplet[l][1] = triplet2[j][1];
            triplet[l][2] = triplet2[j][2];
            l++;
            j++;
        }
    }
}




// for(int i = 0; i < l1+1; i++){
//     for(int j = 0; j < 3; j++){
//         printf("%d ", triplet1[i][j]);
//     }
```

```
    //     printf("\n");
    // }




    // for(int i = 0; i < l2+1; i++){
    //     for(int j = 0; j < 3; j++){
    //         printf("%d ", triplet2[i][j]);
    //     }
    //     printf("\n");
    // }




    printf("Addition of both triplets: \n");

    for(int i = 0; i < l; i++){
        for(int j = 0; j < 3; j++){
            printf("%d ", triplet[i][j]);
        }
        printf("\n");
    }




    return 0;
}
```

Output:



```
Enter first matrix:
0 0 =  3
0 1 = 0
0 2 = 0
1 0 = 10
1 1 = 0
1 2 = 0
2 0 = 5
2 1 = 0
2 2 = 0
Enter second matrix:
0 0 = 0
0 1 = 0
0 2 = 4
1 0 = 9
1 1 = 0
1 2 = 1
2 0 = 0
2 1 = 3
2 2 = 0
Addition of both triplets:
3 3 7
0 0 3
0 2 4
1 0 19
1 2 1
2 0 5
2 1 3
```