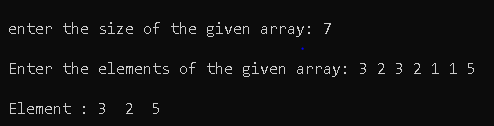
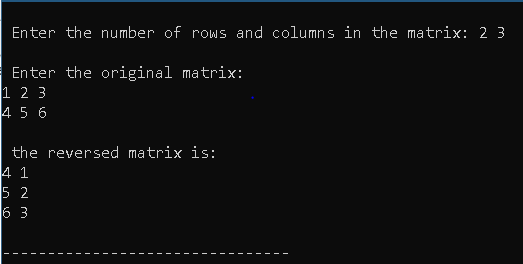
**1.** // FIND THE FREQUENCY OF EACH ELEMENT IN THE GIVEN ARRAY.  
#include <stdio.h>  
#include <stdlib.h>  
int main()  
{  
 int n, i, count = 1, j;  
 printf("\n enter the size of the given array: ");  
 scanf("%d", &n);  
 int a[n], f[n];  
 printf("\n Enter the elements of the given array: ");  
 for(i = 0; i < n; i++)  
 scanf("%d", &a[i]);  
 for(i = 0; i < n; i++)  
 {  
 count = 1;  
 for(j = i + 1; j < n; j++)  
 if(a[i] == a[j])  
 {  
 count++;  
 f[j] = -1;  
 }  
 if (f[i] != -1)  
 printf("%d \t %d \n", a[i], count);  
 }  
 return 0;  
}  
**2.** // FIND THE DTSTINCT ELEMENT IN THE GIVEN ARRAY.  
#include <stdio.h>  
#include <stdlib.h>  
int main()  
{  
 int n, i, j;  
 printf("\n enter the size of the given array: ");  
 scanf("%d", &n);  
 int a[n], f[n];  
 printf("\n Enter the elements of the given array: ");  
 for(i = 0; i < n; i++)  
 scanf("%d", &a[i]);  
 for(i = 0; i < n; i++)  
 {  
// count = 1;  
 for(j = i + 1; j < n; j++)  
 if(a[i] == a[j])  
 {  
// count++;  
 f[j] = -1;  
 }  
//if (f[i] != -1)  
// f[i] = count;  
 }  
 printf("\n Element : ");  
 for(i = 0; i < n; i++)  
 if(f[i] != -1)  
 printf("%d ", a[i]);  
 return 0;  
}

**OUTPUT:** 

**3, 4.** // ROTATE THE ARRAY IN CLOCKWISE OR ANTI-CLOCKWISE ACCORDING TO THE USER'S CHOICE.  
#include <stdio.h>  
#include <stdlib.h>  
void anti\_clockwise(int [], int);  
void clockwise(int [], int);  
int main()  
{  
 int i, n, ch;  
 printf("\n Enter the number of elements to be entered: ");  
 scanf("%d", &n);  
 int a[n];  
 printf("Enter the elements : ");  
 for(i = 0; i < n; i++)  
 scanf("%d", &a[i]);  
 printf("\n Enter 1- CLOCWISE & 2 - ANTICLOCKWISE rotations: ");  
 scanf("%d", &ch);  
 switch(ch)  
 {  
 case 2:  
 printf("anti - clockwise rotate");  
 anti\_clockwise(a, n);  
 break;  
 case 1:  
 printf("clockwise rotate");  
 clockwise(a, n);  
 break;  
 default:  
 printf("\n InvaLid choice \n");  
 }  
 return 0;  
}  
void anti\_clockwise(int a[], int n)  
{  
 int p, i, j;  
 printf("\n Enter the number of times rotation is to be done: ");  
 scanf("%d", &p);  
 for(i = 1; i <= p; i++)  
 {  
 int x = a[0];  
 for(j = 0; j < n - 1; j++)  
 a[j] = a[j+1];  
 a[n-1] = x;  
 }  
 printf("\n After anti-clockwise rotations: ");  
 for(i = 0; i < n; i++)  
 printf("%d ", a[i]);  
}  
void clockwise(int a[], int n)  
{  
 int i, p, j;  
 printf("\n Enter the number of times rotation is to be done: ");  
 scanf("%d", &p);  
 int final[n];  
 for(i = 0; i < n; i++)  
 final[(i + p) % n] = a[i];  
 printf("\n array after rotion: ");  
 for(i = 0; i < n; i++)  
 printf("%d ", final[i]);  
}

**5.** // TO CLOCKWISE ROTATE A 2D ARRAY USING ANOTHER ARRAY.  
#include <stdio.h>  
#include <stdlib.h>  
int main()  
{  
 int r, c, i, j, a, b;  
 printf("\n Enter the number of rows and columns in the matrix: ");  
 scanf("%d%d", &r, &c);  
 int m1[r][c], rev[c][r];  
 printf("\n Enter the original matrix: \n");  
 for(i = 0; i < r; i++)  
 for(j = 0; j < c; j++)  
 scanf("%d", &m1[i][j]);  
 b = r - 1;  
 for(i = 0; i < r; i++)  
 {  
 a = 0;  
 for(j = 0; j < c; j++)  
 rev[a++][b] = m1[i][j];  
 b--;  
 }  
 printf("\n the reversed matrix is: \n");  
 for(i = 0; i < c; i++)  
 {  
 for(j = 0; j < r; j++)  
 printf("%d ", rev[i][j]);  
 printf("\n");  
 }  
 return 0;  
}

**OUTPUT:**

**NAME – SPANDAN CHAUDHURY**

**SECTION – CSE C2**

**ROLL: 02**

**SIC: 190310410**

Open with

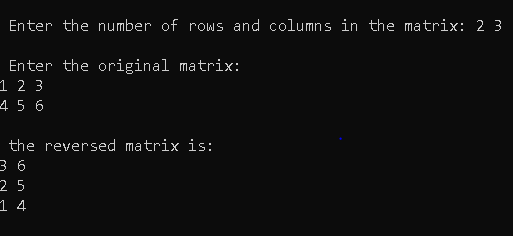


distinct

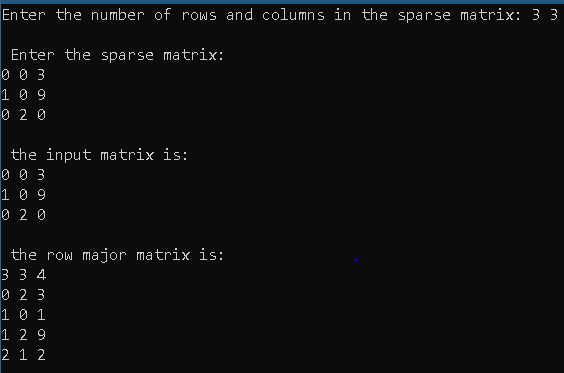
Open with



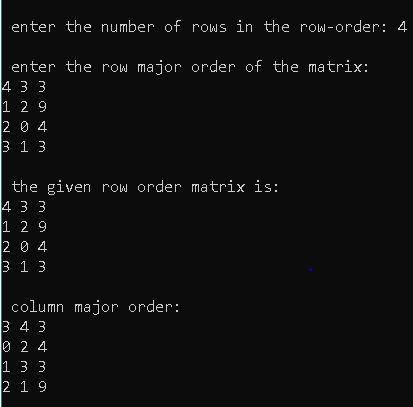
**1.** // TO ANTI-CLOCKWISE ROTATE A 2D ARRAY USING ANOTHER ARRAY.  
#include <stdio.h>  
#include <stdlib.h>  
int main()  
{  
 int r, c, i, j, a, b;  
 printf("\n Enter the number of rows and columns in the matrix: ");  
 scanf("%d%d", &r, &c);  
 int m1[r][c], rev[c][r];  
 printf("\n Enter the original matrix: \n");  
 for(i = 0; i < r; i++)  
 for(j = 0; j < c; j++)  
 scanf("%d", &m1[i][j]);  
 b = 0;  
 for(i = 0; i < r; i++)  
 {  
 a = c - 1;  
 for(j = 0; j < c; j++)  
 rev[a--][b] = m1[i][j];  
 b++;  
 }  
 printf("\n the reversed matrix is: \n");  
 for(i = 0; i < c; i++)  
 {  
 for(j = 0; j < r; j++)  
 printf("%d ", rev[i][j]);  
 printf("\n");  
 }  
 return 0;  
}

**OUTPUT:** 

**2.** // TO CREATE THE ROW-MAJOR MATRIX FROM THE SPARSE MATRIX.  
#include <stdio.h>  
#include <stdlib.h>  
int main()  
{  
 int r, c, i, j, a = 1, b = 0;  
 printf("Enter the number of rows and columns in the sparse matrix: ");  
 scanf("%d%d", &r, &c);  
 int m[r][c], row[r\*c][3];  
 printf("\n Enter the sparse matrix: \n");  
 for(i = 0; i < r; i++)  
 for(j = 0; j < c; j++)  
 scanf("%d", &m[i][j]);  
 printf("\n the input matrix is: \n");  
 for(i = 0; i < r; i++)  
 {  
 for(j = 0; j < c; j++)  
 printf("%d ", m[i][j]);  
 printf("\n");  
 }  
 for(i = 0; i < r; i++)  
 for(j = 0; j < c; j++)  
 if(m[i][j] != 0)  
 {  
 row[a][0] = i;  
 row[a][1] = j;  
 row[a][2] = m[i][j];  
 a++;  
 }  
 row[0][0] = r;  
 row[0][1] = c;  
 row[0][2] = a - 1;  
 printf("\n the row major matrix is: \n");  
 for(i = 0; i < a; i++)  
 {  
 for(j = 0; j < 3; j++)  
 printf("%d ", row[i][j]);  
 printf("\n");  
 }  
 return 0;  
}  
  
**OUTPUT:**



**3.** // TO CREATE THE COLUMN-MAJOR MATRIX FROM THE SPARSE MATRIX.  
#include <stdio.h>  
#include <stdlib.h>  
int main()  
{  
 int r, c, i, j, a = 1;  
 printf("Enter the number of rows and columns in the sparse matrix: ");  
 scanf("%d%d", &r, &c);  
 int m[r][c], col[r\*c][3];  
 printf("\n Enter the sparse matrix: \n");  
 for(i = 0; i < r; i++)  
 for(j = 0; j < c; j++)  
 scanf("%d", &m[i][j]);  
 printf("\n the input matrix is: \n");  
 for(i = 0; i < r; i++)  
 {  
 for(j = 0; j < c; j++)  
 printf("%d ", m[i][j]);  
 printf("\n");  
 }  
 for(j = 0; j < c; j++)  
 for(i = 0; i < r; i++)  
 if(m[i][j] != 0)  
 {  
 col[a][0] = j;  
 col[a][1] = i;  
 col[a][2] = m[i][j];  
 a++;  
 }  
 col[0][0] = c;  
 col[0][1] = r;  
 col[0][2] = a - 1;  
 printf("\n the column major matrix is: \n");  
 for(i = 0; i < a; i++)  
 {  
 for(j = 0; j < 3; j++)  
 printf("%d ", col[i][j]);  
 printf("\n");  
 }  
 return 0;  
}  
  
**4.** // TO CHANGE THE ROW ORDER MATRIX TO ITS COLUMN ORDER.  
#include <stdio.h>  
#include <stdlib.h>  
int main()  
{  
 int r, c = 3, i, j, a, b, t;  
 printf("\n enter the number of rows in the row-order: ");  
 scanf("%d", &r);  
 int m[r][3], col[r][3];  
 printf("\n enter the row major order of the matrix: \n");  
 for(i = 0; i < r; i++)  
 for(j = 0; j < 3; j++)  
 scanf("%d", &m[i][j]);  
 printf("\n the given row order matrix is: \n");  
 for(i = 0; i < r; i++)  
 {  
 for(j = 0; j < 3; j++)  
 printf("%d ", m[i][j]);  
 printf("\n");  
 }  
 a = 1;  
 t = m[0][1]; // total number of columns present in the sparse matrix.  
  
 for(c = 0; c < t; c++)  
 {  
 for(i = 0; i < r; i++)  
 if(m[i][1] == c)  
 {  
 col[a][0] = c;  
 col[a][1] = m[i][0];  
 col[a][2] = m[i][2];  
 a++;  
 }  
 }  
   
 col[0][0] = m[0][1];  
 col[0][1] = m[0][0];  
 col[0][2] = m[0][2];  
 printf("\n column major order: \n");  
 for(i = 0; i < a; i++)  
 {  
 for(j = 0; j < 3; j++)  
 printf("%d ", col[i][j]);  
 printf("\n");  
 }  
 return 0;  
}

**OUTPUT:** 

**5.** // TO PERFORM ADDITION OF TWO SPARSE MATRICES.  
#include <stdio.h>  
#include <stdlib.h>  
void read\_matrix(int a[][3], int);  
int main()  
{  
 int r, i, j, k = 0, t1, t2;  
 printf("\n enter the number of rows in both the row-major matrices: ");  
 scanf("%d", &r);  
 int m1[r][3], m2[r][3], m3[r+ r][3];  
 printf("\n enter the first row-major order matrix: \n");  
 read\_matrix(m1, r);  
 printf("\n enter the second row-major order matrix: \n");  
 read\_matrix(m2, r);  
   
 t1 = m1[0][2];  
 t2 = m2[0][2];  
 i = 0, j = 0;  
 while (i != t1 && j != t2)  
 {  
 if(m1[i][0] != m2[j][0]) // if the rows are not equal  
 {  
 if(m1[i][0] > m2[j][0]) //   
 {  
 m3[k][0] = m2[j][0];  
 m3[k][1] = m2[j][1];  
 m3[k][2] = m2[j][2];  
 k++;  
 j++;  
 }  
 else  
 {  
 m3[k][0] = m1[i][0];  
 m3[k][1] = m1[i][1];  
 m3[k][2] = m2[i][2];  
 k++;  
 i++;  
 }  
 }  
 else // if the rows are equal, then check the column number  
 {  
 if(m1[i][1] > m2[j][1]) // if the column number of the 2nd matrix is smaller than that of the 1st one.  
 {  
 m3[k][0] = m2[j][0];  
 m3[k][1] = m2[j][1];  
 m3[k][2] = m2[j][2];  
 k++;  
 j++;  
 }  
 else if(m1[i][1] < m2[j][1]) // if the column number of the first matrix is smaller than that of the 2nd one  
 {  
 m3[k][0] = m1[i][0];  
 m3[k][1] = m1[i][1];  
 m3[k][2] = m1[i][2];  
 i++;  
 k++;  
 }  
 else // if both the column and row number matches.  
 {  
 m3[k][0] = m1[i][0];  
 m3[k][1] = m1[i][1];  
 m3[k][2] = m1[i][2] + m2[j][2];  
 i++;  
 j++;  
 k++;  
 }  
 }  
 }  
  
 if(i < t1)  
 {  
 while(i != t1)  
 {  
 m3[k][0] = m1[i][0];  
 m3[k][1] = m1[i][1];  
 m3[k][2] = m1[i][2];  
 i++;  
 k++;  
 }  
 }  
 else if(j < t2)  
 {  
 while(j != t2)  
 {  
 m3[k][0] = m2[j][0];  
 m3[k][1] = m2[j][1];  
 m3[k][2] = m2[j][2];  
 j++;  
 k++;  
 }  
 }  
 printf("\n the addition of the two sparse natrices: \n");  
 for(i = 0; i < k; i++)  
 {  
 for(j = 0; j < 3; j++)  
 printf("%d ", m3[i][j]);  
 printf("\n");  
 }  
 return 0;  
}  
void read\_matrix(int a[][3], int r)  
{  
 int i, j;  
 for(i = 0; i < r; i++)  
 for(j = 0; j < 3; j++)  
 scanf("%d", &a[i][j]);  
}

**NAME: SPANDAN CHAUDHURY**

**SECTION – CSE C2**

**ROLL – 02**

**SIC – 190310410**

**1.** // TO CONVERT FROM ROW ORDER MATRIX TO SPARSE MATRIX.  
#include <stdio.h>  
#include <stdlib.h>  
int main()  
{  
 int n, i, j, r, c;  
 printf("\n enter the number of rows in the row-major matrix: ");  
 scanf("%d", &n);  
 int m[n][3];  
 printf("\n enter the row-major order matrix: \n");  
 for(i = 0; i < n; i++)  
 for(j = 0; j < 3; j++)  
 scanf("%d", &m[i][j]);  
 r = m[0][0];  
 c = m[0][1];  
 int sparse[r][c];  
 for(i = 0; i < r; i++)  
 for(j = 0; j < c; j++)  
 sparse[i][j] = 0;  
 for(i = 1; i < n; i++)  
 sparse[m[i][0]][m[i][1]] = m[i][2];  
 printf("\n the sparse matrix is: \n");  
 for(i = 0; i < r; i++)  
 {  
 for(j = 0; j < c; j++)  
 printf(" %d ", sparse[i][j]);  
 printf("\n");  
 }  
 return 0;  
}

**2.** // REVERSE A STRING WITHOUT USING ANOTHER STRING.  
#include <stdio.h>  
#include <stdlib.h>  
# define max 60  
int main()  
{  
 int i, l = 0;  
 char s[max], c;  
 printf("\n enter the string to be reversed: ");  
 scanf("%[^\n]", s);  
 for(i = 0; s[i] != '\0'; i++)  
 l++;  
 i = 0;  
 while (i < l / 2)  
 {  
 c = s[i];  
 s[i] = s[l - i - 1];  
 s[l - i - 1] = c;  
 i++;  
 }  
 printf("%s \n", s);  
 return 0;  
}

**OUTPUT:**

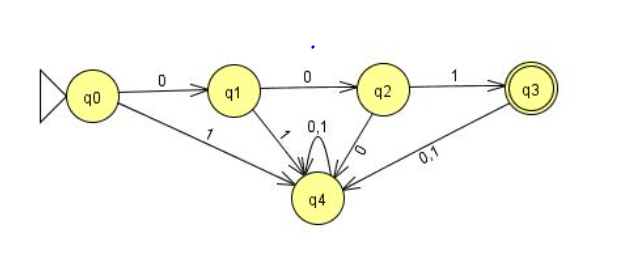
**3.** // TO CONCAT 2 STRINGS USING 3RD STRING.  
#include <stdio.h>  
#include <stdlib.h>  
#include <string.h>  
# define max 60  
int main()  
{  
 char s1[max], s2[max];  
 int i, l1 = 0, l2 = 0, j;  
 printf("\n enter the first string: ");  
 scanf("%[^\n]", s1);  
// printf("%s", s1);  
 printf("\n enter the second string: ");  
 scanf(" %[^\n]", s2);  
 for(i = 0; s1[i] != '\0'; i++)  
 l1++;  
 for(i = 0; s2[i] != '\0'; i++)  
 l2++;  
 char s3[l1 + l2];  
 for(i = 0; i < l1; i++)  
 s3[i] = s1[i];  
 for(j = 0; j < l2; j++)  
 s3[i++] = s2[j];  
 s3[i] = '\0';  
 printf("\n the concatenated string is: %s \n", s3);  
  
   
 return 0;  
}

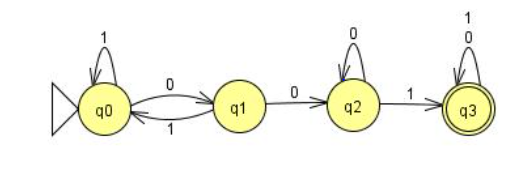
**4.** // TO CONCAT 2 STRINGS WITHOUT USING THE 3RD STRING.  
#include <stdio.h>  
#include <stdlib.h>  
# define max 100  
int main()  
{  
 char s1[2 \* max], s2[max];  
 int i, l1 = 0, l2 = 0, j;  
 printf("\n enter the first string: ");  
 scanf("%[^\n]", s1);  
 printf("\n enter the second string: ");  
 scanf(" %[^\n]", s2);  
 for(i = 0; s1[i] != '\0'; i++)  
 l1++;  
 s1[l1++] = ' ';  
 for(i = 0; s2[i] != '\0'; i++)  
 s1[l1++] = s2[i];  
 s1[l1] = '\0';  
 printf("\n the concatenated string is: %s \n", s1);  
 return 0;  
}

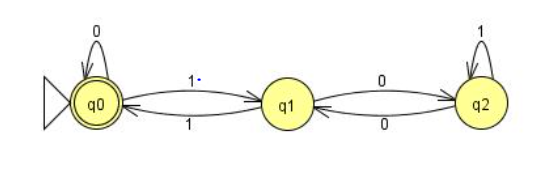
**5.** // TO CHECK IF THE INPUT STRING IS PALINDROME OR NOT.  
#include <stdio.h>  
#include <stdlib.h>  
# define max 60  
int main()  
{  
 int i, l = 0, j = 0;  
 char s[max], r[max], c;  
 printf("\n enter the string to be checked: ");  
 scanf("%[^\n]", s);  
 for(i = 0; s[i] != '\0'; i++)  
 l++;  
 for(i = l - 1; i >= 0; i--)  
 r[j++] = s[i];  
 for(i = 0; i < l; i++)  
 if(s[i] != r[i])  
 break;  
 if(i < l)  
 printf("\n not a palindrome string\n");  
 else  
 printf("\n palindrome string\n");  
 printf("%s \n", s);  
 return 0;  
}

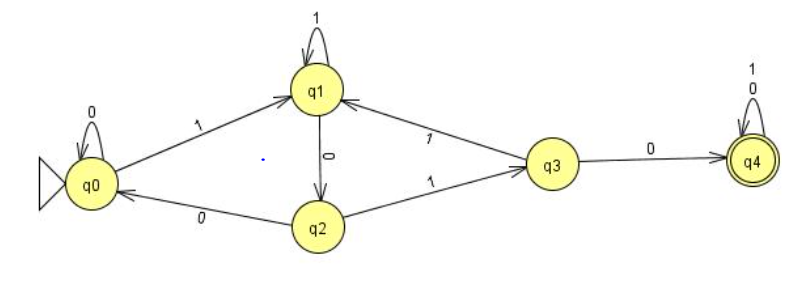
**NAME: SPANDAN CHAUDHURY** **CSE C2** **ROLL-02**

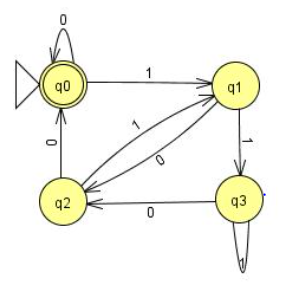
**SIC-190310410**

**1.**

**2.**

**3.**

**4,** 

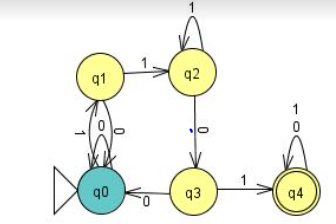
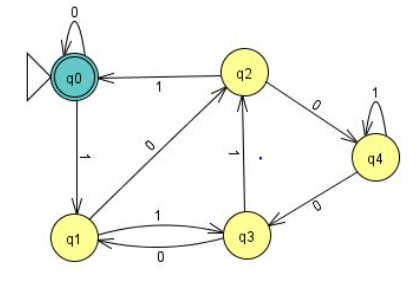
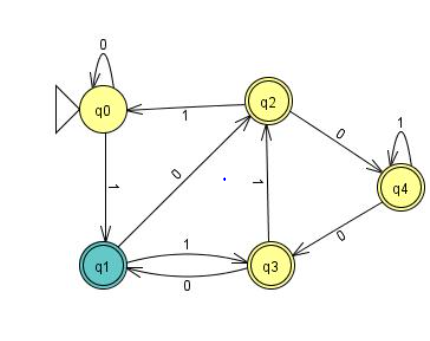
**5.** 

**NAME: SPANDAN CHAUDHURY**

**SECTION: CSE C2**

**ROLL : 02**

**SIC: 190310410**

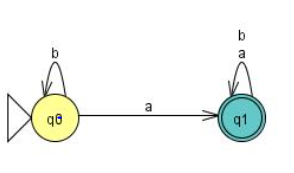
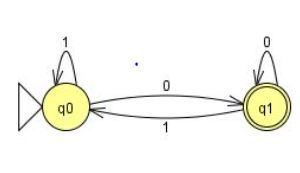
1. 
2. 
3. 

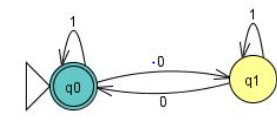
**NAME: SPANDAN CHAUDHURY**

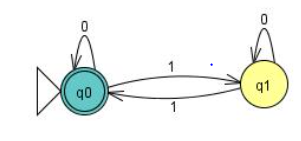
**SECTION: CSE C2**

**ROLL: 02**

**SIC: 190310410**

1. 
2. 

4. 

5. 

**NAME: SPANDAN CHAUDHURY**

**SECTION: CSE C2**

**ROLL: 02**

**SIC: 190310410**