**DSA LAB ASSIGNMENT SUBMISSION**

Name : Vanshikaa

Roll No. : 102103580

Group : 2CO21

**LAB 1A :**

Q1 a) #include<iostream>

using namespace std;

int main(){

    int arr[3] ={1,2,3} ;

    for(int i =0;i<3;i++){

        for(int j =0;j<3;j++){

            if(i!=j){

                cout<<"0";

            }

            else{

                cout<<arr[i];

            }

        }

        cout<<"\n";

    }

    return 0;

}

Output :

100

020

003

Q1b) #include<iostream>

using namespace std;

int main(){

    int arr[16] ={1,2,3,4,5,6,7,8,9,10} ;

    for(int i =0;i<4;i++){

        for(int j =0;j<4;j++){

            if(abs(i-j)>1){

                cout<<"0";

            }

            else{

                int k;

                cout<<arr[k];

                k++;

            }

        }

        cout<<"\n";

    }

    return 0;

}

Output :

1200

3450

0678

00910

Q1c)

#include<iostream>

using namespace std;

int main(){

    int arr[16] ={1,2,3,4,5,6,7,8,9,10} ;

    for(int i =0;i<4;i++){

        for(int j =0;j<4;j++){

            if(i>j){

                cout<<"0";

            }

            else{

                int k;

                cout<<arr[k];

                k++;

            }

        }

        cout<<"\n";

    }

    return 0;

}

Output :

1234

0567

0089

00010

Q1d) #include<iostream>

using namespace std;

int main(){

    int arr[16] ={1,2,3,4,5,6,7,8,9,10} ;

    for(int i =0;i<4;i++){

        for(int j =0;j<4;j++){

            if(i<j){

                cout<<"0";

            }

            else{

                int k;

                cout<<arr[k];

                k++;

            }

        }

        cout<<"\n";

    }

    return 0;

}

Output :

1000

2300

4560

78910

Q1e)

#include<iostream>

using namespace std;

int main(){

    int arr[10] ={1,2,3,4,5,6,7,8,9,10} ;

    for(int i =0;i<4;i++){

        for(int j =0;j<4;j++){

            if(i==j){

                cout<<arr[i];

            }

           else {

                int k;

                cout<<arr[k];

                k++;

            }

        }

        cout<<"\n";

    }

    return 0;

}

Q2a)

#include<stdio.h>

#define MAX 20

void printsparse(int[][3]);

void readsparse(int[][3]);

void transpose(int[][3],int[][3]);

int main()

{

int b1[MAX][3],b2[MAX][3],m,n;

printf("Enter the size of matrix (rows,columns):");

scanf("%d%d",&m,&n);

b1[0][0]=m;

b1[0][1]=n;

readsparse(b1);

transpose(b1,b2);

printsparse(b2);

}

void readsparse(int b[MAX][3])

{

int i,t;

printf("\nEnter no. of non-zero elements:");

scanf("%d",&t);

b[0][2]=t;

for(i=1;i<=t;i++)

{

printf("\nEnter the next triple(row,column,value):");

scanf("%d%d%d",&b[i][0],&b[i][1],&b[i][2]);

}

}

void printsparse(int b[MAX][3])

{

int i,n;

n=b[0][2];

printf("\nAfter Transpose:\n");

printf("\nrow\t\tcolumn\t\tvalue\n");

for(i=0;i<=n;i++)

printf("%d\t\t%d\t\t%d\n",b[i][0],b[i][1],b[i][2]);

}

void transpose(int b1[][3],int b2[][3])

{

int i,j,k,n;

b2[0][0]=b1[0][1];

b2[0][1]=b1[0][0];

b2[0][2]=b1[0][2];

k=1;

n=b1[0][2];

for(i=0;i<b1[0][1];i++)

for(j=1;j<=n;j++)

if(i==b1[j][1])

{

b2[k][0]=i;

b2[k][1]=b1[j][0];

b2[k][2]=b1[j][2];

k++;

}

}

Output :

Enter the size of matrix (rows,columns):3 3

Enter no. of non-zero elements:4 5 6 6 7 8 9 7

Enter the next triple(row,column,value):

Enter the next triple(row,column,value):

Enter the next triple(row,column,value):7 8 9

Enter the next triple(row,column,value):7 8 9

After Transpose:

row column value

3 3 4

1 2 1

2 1 3

3 2 2

Q2b)

#include <stdio.h>

#include<stdlib.h>

struct Element

{

 int i;

 int j;

 int x;

};

struct Sparse

{

 int m;

 int n;

 int num;

 struct Element \*ele;

};

void create(struct Sparse \*s)

{

 int i;

 printf("Eneter Dimensions");

 scanf("%d%d",&s->m,&s->n);

 printf("Number of non-zero");

 scanf("%d",&s->num);

 s->ele=(struct Element \*)malloc(s->num\*sizeof(struct

Element));

 printf("Eneter non-zero Elements");

 for(i=0;i<s->num;i++)

 scanf("%d%d%d",&s->ele[i].i,&s->ele[i].j,&s->ele[i].x);

}

void display(struct Sparse s)

{

 int i,j,k=0;

 for(i=0;i<s.m;i++)

 {

 for(j=0;j<s.n;j++)

 {

 if(i==s.ele[k].i && j==s.ele[k].j)

 printf("%d ",s.ele[k++].x);

 else

 printf("0 ");

 }

 printf("\n");

 }

}

struct Sparse \* add(struct Sparse \*s1,struct Sparse \*s2)

{

 struct Sparse \*sum;

 int i,j,k;

 i=j=k=0;

 if(s1->n != s2->n && s1->m != s2->m)

 return NULL;

 sum=(struct Sparse \*)malloc(sizeof(struct Sparse));

sum->ele=(struct Element \*)malloc((s1->num+s2->num)\*sizeof(struct Element));

 while(i<s1->num && j<s2->num)

 {

 if(s1->ele[i].i<s2->ele[j].i)

 sum->ele[k++]=s1->ele[i++];

 else if(s1->ele[i].i>s2->ele[j].i)

 sum->ele[k++]=s2->ele[j++];

 else

 {

 if(s1->ele[i].j<s2->ele[j].j)

 sum->ele[k++]=s1->ele[i++];

 else if(s1->ele[i].j>s2->ele[j].j)

 sum->ele[k++]=s2->ele[j++];

 else

 {

 sum->ele[k]=s1->ele[i];

 sum->ele[k++].x=s1->ele[i++].x+s2->ele[j++].x;

 }

 }

 }

 for(;i<s1->num;i++)sum->ele[k++]=s1->ele[i];

 for(;j<s2->num;j++)sum->ele[k++]=s2->ele[j];

 sum->m=s1->m;

 sum->n=s1->n;

 sum->num=k;

 return sum;

}

int main()

{

 struct Sparse s1,s2,\*s3;

 create(&s1);

 create(&s2);

 s3=add(&s1,&s2);

 printf("First Matrix\n");

 display(s1);

 printf("Second Matrix\n");

 display(s2);

 printf("Sum Matrix\n");

 display(\*s3);

 return 0;

}

Output :

Eneter Dimensions5 5

Number of non-zero5

Eneter non-zero Elements

0 0 1

1 1 1

2 2 1

3 3 1

4 4 1

Eneter Dimensions5 5

Number of non-zero5

Eneter non-zero Elements

0 0 2

1 0 2

2 0 2

3 0 2

4 0 2

First Matrix

1 0 0 0 0

0 1 0 0 0

0 0 1 0 0

0 0 0 1 0

0 0 0 0 1

Second Matrix

2 0 0 0 0

2 0 0 0 0

2 0 0 0 0

2 0 0 0 0

2 0 0 0 0

Sum Matrix

3 0 0 0 0

2 1 0 0 0

2 0 1 0 0

2 0 0 1 0

2 0 0 0 1

Q2c) #include<iostream>

using namespace std;

int main()

{

   int r1,r2,c1,c2,sw,i1,j1,i2,j2,k,p,q,r,i3,j3;

   int a[200][200],b[200][200],pro[200][200];

   cout<<"enter sparse matrix 1 rows and columns :";

   cin>>r1>>c1;

   cout<<"enter sparse matrix 2 rows and columns :";

   cin>>r2>>c2;

   if(c1!=r2)

   {

       sw=1;

   }

   else

   {

       sw=2;

   }

   switch(sw)

   {

   case 1:

        cout<<"matrix multiplication not possible...!!";

        break;

   case 2:

       cout<<"enter sparse matrix 1 elements...."<<endl;

    for(i1=0;i1<r1;i1++)

    {

        for(j1=0;j1<c1;j1++)

        {

            cin>>a[i1][j1];

        }

    }

    cout<<"enter sparse matrix 2 elements...."<<endl;

    for(i2=0;i2<r2;i2++)

    {

        for(j2=0;j2<c2;j2++)

        {

            cin>>b[i2][j2];

        }

    }

    for(p=0;p<r1;p++)

    {

        for(q=0;q<c2;q++)

        {

            pro[p][q]=0;

            for(r=0;r<c1;r++)

            {

                pro[p][q]=pro[p][q]+a[p][r]\*b[r][q];

            }

        }

    }

    cout<<"the product of the two sparse matrices is..."<<endl;

    for(i3=0;i3<r1;i3++)

    {

        for(j3=0;j3<c2;j3++)

        {

            cout<<pro[i3][j3]<<" ";

        }

        cout<<endl;

    }

    break;

   }

   return 0;

}

Output :

enter sparse matrix 1 rows and columns :2 3

enter sparse matrix 2 rows and columns :3 2

enter sparse matrix 1 elements....

1

2

3

4

5

6

enter sparse matrix 2 elements....

2

3

4

5

6

7

the product of the two sparse matrices is...

28 34

64 79

Q3 .

#include <iostream>

using namespace std;

#define m 4

#define n 4

void row\_sum(int arr[m][n])

{

    int i,j,sum = 0;

    cout << "\nFinding Sum of each row:\n\n";

    for (i = 0; i < m; ++i) {

        for (j = 0; j < n; ++j) {

            sum = sum + arr[i][j];

        }

        cout

            << "Sum of the row "

            << i << " = " << sum

            << endl;

        sum = 0;

    }

}

void column\_sum(int arr[m][n])

{

    int i,j,sum = 0;

    cout << "\nFinding Sum of each column:\n\n";

    for (i = 0; i < m; ++i) {

        for (j = 0; j < n; ++j) {

            sum = sum + arr[j][i];

        }

        cout

            << "Sum of the column "

            << i << " = " << sum

            << endl;

        sum = 0;

    }

}

int main()

{

    int i,j;

    int arr[m][n];

    int x = 1;

    for (i = 0; i < m; i++)

        for (j = 0; j < n; j++)

            arr[i][j] = x++;

    row\_sum(arr);

    column\_sum(arr);

    return 0;

}

Output :

Finding Sum of each row:

Sum of the row 0 = 10

Sum of the row 1 = 26

Sum of the row 2 = 42

Sum of the row 3 = 58

Finding Sum of each column:

Sum of the column 0 = 28

Sum of the column 1 = 32

Sum of the column 2 = 36

Sum of the column 3 = 40

Q4.

#include <iostream>

using namespace std;

const int MAX = 100;

bool findSaddlePoint(int mat[MAX][MAX], int n)

{

    for (int i = 0; i < n; i++)

    {

        int min\_row = mat[i][0], col\_ind = 0;

        for (int j = 1; j < n; j++)

        {

            if (min\_row > mat[i][j])

            {

                min\_row = mat[i][j];

                col\_ind = j;

            }

        }

        int k;

        for (k = 0; k < n; k++)

            if (min\_row < mat[k][col\_ind])

                break;

        if (k == n)

        {

           cout << "Value of Saddle Point " << min\_row;

           return true;

        }

    }

    return false;

}

int main()

{

    int mat[MAX][MAX] = {{1, 2, 3},

                        {4, 5, 6},

                        {7, 8, 9}};

    int n = 3;

    if (findSaddlePoint(mat, n) == false)

       cout << "No Saddle Point ";

    return 0;

}

Output :

Value of Saddle Point 7

Q5. #include <iostream>

#include <vector>

using namespace std;

void printSpiralOrder(vector<vector<int>> const &mat)

{

    if (mat.size() == 0) {

        return;

    }

    int top = 0, bottom = mat.size() - 1;

    int left = 0, right = mat[0].size() - 1;

    while (1)

    {

        if (left > right) {

            break;

        }

        for (int i = left; i <= right; i++) {

            cout << mat[top][i] << " ";

        }

        top++;

        if (top > bottom) {

            break;

        }

        for (int i = top; i <= bottom; i++) {

            cout << mat[i][right] << " ";

        }

        right--;

        if (left > right) {

            break;

        }

        for (int i = right; i >= left; i--) {

            cout << mat[bottom][i] << " ";

        }

        bottom--;

        if (top > bottom) {

            break;

        }

        for (int i = bottom; i >= top; i--) {

            cout << mat[i][left] << " ";

        }

        left++;

    }

}

int main()

{

    vector<vector<int>> mat =

    {

        { 1, 2, 3, 4, 5},

        {16, 17, 18, 19, 6},

        {15, 24, 25, 20, 7},

        {14, 23, 22, 21, 8},

        {13, 12, 11, 10, 9}

    };

    printSpiralOrder(mat);

    return 0;

}

Output :

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25

Q6. #include <iostream>

using namespace std;

int main(){

    int r,c;

    cout << "Enter the number of rows and columns: " << endl;

    cin >> r >> c;

    int arr[100][100];

    int m = r, n = c;

    int k = 0, l = 0 , i = 0;

    while(k < r && l < c){

        for(i = l; i < c; i++){

            cin >> arr[k][i];

        }

        k++;

        for(i = k; i < r; i++){

            cin >> arr[i][c - 1];

        }

        c--;

        if(k < r){

            for(i = c - 1; i >= l; i--){

                cin >> arr[r-1][i];

            }

            r--;

        }

        if(l < c){

            for(i = r - 1;i >= k; i--){

                cin >> arr[i][l];

            }

            l++;

        }

    }

    for(int i = 0; i < m; i++){

        cout << "|";

        for(int j = 0; j < n; j++){

            cout << " " << arr[i][j] << " ";

        }

        cout << "|" << endl;

    }

}

Output :

Enter the number of rows and columns:

3 3

1

2

3

4

5

6

7

8

9

| 1 2 3 |

| 8 9 4 |

| 7 6 5 |