Practical 02

Name: Shruti Rajesh Kumbhare Roll no: 11

Batch: B1

Aim: To study array ADT and to implement various operations on matrix(addition, multiplication, transpose, row major, column major)

**Code:** #include<stdio.h> struct Matrix{

int row,column,arr[10][10];

};

void rowmajor(struct Matrix M)

{

printf("no. of rows?\n");

scanf("%d",&M.row); printf("no. of columns?\n"); scanf("%d",&M.column); printf("enter elements\n"); for(int i=0;i<M.row;i++)

{

for(int j=0;j<M.column;j++)

{

scanf("%d",&M.arr[i][j]);

}

} printf("row major order :\n"); for(int i=0;i<M.row;i++)

{

for(int j=0;j<M.column;j++)

{

printf("%d ",M.arr[i][j]);

}

}

}

void columnmajor(struct Matrix M)

{

printf("no. of rows?\n");

scanf("%d",&M.row); printf("no. of columns?\n"); scanf("%d",&M.column); printf("enter elements\n"); for(int i=0;i<M.row;i++)

{

for(int j=0;j<M.column;j++)

{

scanf("%d",&M.arr[i][j]);

}

}

printf("column major order :\n"); for(int i=0;i<M.row;i++)

{

for(int j=0;j<M.column;j++)

{

printf("%d ",M.arr[j][i]);

}

}

}

void transpose(struct Matrix M)

{ int t[5][5];

printf("no. of rows?\n");

scanf("%d",&M.row); printf("no. of columns?\n"); scanf("%d",&M.column);

printf("enter elements\n"); for(int i=0;i<M.row;i++)

{

for(int j=0;j<M.column;j++)

{

scanf("%d",&M.arr[i][j]);

}

}

for(int i=0;i<M.row;i++)

{

for(int j=0;j<M.column;j++)

{

t[i][j]=M.arr[j][i];

}

}

printf("transpose:\n"); for(int i=0;i<M.row;i++)

{

for(int j=0;j<M.column;j++)

{

printf("%d ",t[i][j]);

}

printf("\n");

}

}

void add(struct Matrix M,struct Matrix N)

{ int P[5][5];

printf("no. of rows?\n");

scanf("%d",&M.row); printf("no. of columns?\n");

scanf("%d",&M.column); printf("enter elements of M\n"); for(int i=0;i<M.row;i++)

{

for(int j=0;j<M.column;j++)

{

scanf("%d",&M.arr[i][j]);

}

}

printf("enter elements of N\n"); for(int i=0;i<M.row;i++)

{

for(int j=0;j<M.column;j++)

{

scanf("%d",&N.arr[i][j]);

}

}

for(int i=0;i<M.row;i++)

{

for(int j=0;j<M.column;j++)

{

P[i][j]=M.arr[i][j]+N.arr[i][j];

}

}

printf("added matrix:\n"); for(int i=0;i<M.row;i++)

{

for(int j=0;j<M.column;j++)

{

printf("%d ",P[i][j]);

}

printf("\n");

}

}

void multiply(struct Matrix M, struct Matrix N)

{

int P[5][5];

printf("no. of rows of M?\n"); scanf("%d",&M.row);

printf("no. of columns of M?\n"); scanf("%d",&M.column); printf("enter elements of M\n"); for(int i=0;i<M.row;i++)

{

for(int j=0;j<M.column;j++)

{

scanf("%d",&M.arr[i][j]);

}

}

printf("no. of rows of N?\n"); scanf("%d",&N.row);

printf("no. of columns of N?\n"); scanf("%d",&N.column); printf("enter elements of N\n"); for(int i=0;i<N.row;i++)

{

for(int j=0;j<N.column;j++)

{

scanf("%d",&N.arr[i][j]);

}

}

if(M.column!=N.row)

{

printf(" multiplication cannot be done,\n");

}

else{

for(int i=0;i<M.row;i++)

{

for(int j=0;j<N.column;j++)

{

P[i][j]=0;

// Multiplying i’th row with j’th column

for(int k=0;k<M.row;k++)

{

P[i][j]+=M.arr[i][k]\*N.arr[k][j];

}

}

}

printf("Multiplied matrix\n"); for(int i=0;i<M.row;i++)

{

for(int j=0;j<N.column;j++)

{

printf("%d ",P[i][j]);

}

printf("\n");

}

}

}

int main()

{ int choice;

printf("enter choice\n 1.Row major\n 2.Column major\n 3.Transpose\n 4.Add\n 5.Multiply\n"); scanf("%d",&choice);

struct Matrix M; struct Matrix N; switch(choice)

{

case 1:

{

rowmajor(M); break;

}

case 2:

{

columnmajor(M); break;

}

case 3:

{

transpose(M); break;

}

case 4:

{

add(M,N); break;

}

case 5:

{

multiply(M,N); break;

}

default:

{

printf("invalid choice"); break;

}

}

}

**Output:**









