Double Dimension Arrays

1. Print the transpose of Matrix

import java.io.\*;

public class Transpose

{

public static void main(String args[]) throws IOException

{

int i, j;

System.out.println("Enter total rows and columns: ");

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

int row = Integer.parseInt(br.readLine());

int column = Integer.parseInt(br.readLine());

int array[][] = new int[row][column];

System.out.println("Enter matrix:");

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

{

for(j = 0; j < column; j++)

{

array[i][j] = Integer.parseInt(br.readLine());

System.out.print(" ");

}

}

System.out.println("The above matrix before Transpose is ");

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

{

for(j = 0; j < column; j++)

{

System.out.print(array[i][j]+" ");

}

System.out.println(" ");

}

System.out.println("The above matrix after Transpose is ");

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

{

for(j = 0; j < row; j++)

{

System.out.print(array[j][i]+" ");

}

System.out.println(" ");

}

}

}

Input:

Enter total rows and columns:

3

3

Enter matrix:

1

2

3

4

5

6

7

8

9

Output:

The above matrix before Transpose is

1 2 3

4 5 6

7 8 9

The above matrix after Transpose is

1 4 7

2 5 8

3 6 9

2. Matrix Addition

import java.io.\*;

class AddMatrix

{

public static void main(String args[])throws IOException

{

int m, n, c, d;

BufferedReader br=new BufferedReader(new InputStreamReader (System.in));

System.out.println("Enter the number of rows and columns of matrix");

m = Integer.parseInt(br.readLine());

n = Integer.parseInt(br.readLine());

int first[][] = new int[m][n];

int second[][] = new int[m][n];

int sum[][] = new int[m][n];

System.out.println("Enter the elements of first matrix");

for (c = 0; c < m; c++)

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

first[c][d] = Integer.parseInt(br.readLine());

System.out.println("Enter the elements of second matrix");

for (c = 0 ; c < m ; c++)

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

second[c][d] = Integer.parseInt(br.readLine());

for (c = 0; c < m; c++)

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

sum[c][d] = first[c][d] + second[c][d];

System.out.println("Sum of the matrices:");

for (c = 0; c < m; c++)

{

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

System.out.print(sum[c][d]+"\t");

System.out.println();

}

}

}

Input:

Enter the number of rows and columns of matrix

3

3

Enter the elements of first matrix

1

2

3

4

5

6

7

8

9

Enter the elements of second matrix

2

4

6

8

10

12

14

16

18

Output:

Sum of the matrices:

3 6 9

12 15 18

21 24 27

3. Matrix Multiplication

import java.io.\*;

class Multiplication

{

public static void main(String args[])throws IOException

{

int m, n, p, q, sum = 0, c, d, k;

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

System.out.println("Enter the number of rows and columns of first matrix");

m = Integer.parseInt(br.readLine());

n = Integer.parseInt(br.readLine());

int first[][] = new int[m][n];

System.out.println("Enter elements of first matrix");

for (c = 0; c < m; c++)

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

first[c][d] = Integer.parseInt(br.readLine());

System.out.println("Enter the number of rows and columns of second matrix");

p = Integer.parseInt(br.readLine());

q = Integer.parseInt(br.readLine());

if (n != p)

System.out.println("The matrices can't be multiplied with each other.");

else

{

int second[][] = new int[p][q];

int multiply[][] = new int[m][q];

System.out.println("Enter elements of second matrix");

for (c = 0; c < p; c++)

for (d = 0; d < q; d++)

second[c][d] = Integer.parseInt(br.readLine());

for (c = 0; c < m; c++)

{

for (d = 0; d < q; d++)

{

for (k = 0; k < p; k++)

{

sum = sum + first[c][k]\*second[k][d];

}

multiply[c][d] = sum;

sum = 0;

}

}

System.out.println("Product of the matrices:");

for (c = 0; c < m; c++)

{

for (d = 0; d < q; d++)

System.out.print(multiply[c][d]+"\t");

System.out.print("\n");

}

}

}

}

Input:

Enter the number of rows and columns of first matrix

3

2

Enter elements of first matrix

1

2

3

4

5

6

Enter the number of rows and columns of second matrix

2

2

Enter elements of second matrix

2

4

6

8

Output:

Product of the matrices:

14 20

30 44

46 68

4. A wondrous square is an n by n grid which fulfils the following conditions:

* It contains integers from 1 to n2, where each integer appear only once
* The sum of integer in any row or column must add up to 0.5x n x (n2 + 1)

Write a program to read n (2 <= n <=10) and the values stored in these n by n cells and output if the grid represents a wondrous square or not

Also, output all the prime numbers in the grid along with their row index and column index as shown in the output.

import java.io.\*;

class WondrousSq

{

int arr[][],arr1[];;

int n,i,j,x=0,r,c;

int flag;

void WonSq()throws IOException

{

BufferedReader br=new BufferedReader (new InputStreamReader (System.in));

System.out.println("Enter the size of array(row and column same):");

n=Integer.parseInt(br.readLine());

arr=new int[n][n];

arr1=new int[2\*n];

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

{

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

{

System.out.print("Enter the value:");

arr[i][j]=Integer.parseInt(br.readLine());

}

}

System.out.println("The matrix is");

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

{

r=0;

c=0;

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

{

System.out.print(arr[i][j]+" ");

r=r+arr[i][j];

c=c+arr[j][i];

}

System.out.println();

arr1[x]=r;

arr1[x+n-1]=c;

x++;

}

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

{

if(arr1[i]!= 0.5 \* n \* (n\*n + 1))

break;

}

if(i==x)

System.out.println("YES IT REPRESENTS A WONDROUS SQUARE.");

else

System.out.println("IT IS NOT A WONDROUS SQUARE.");

System.out.println("PRIME ROW COLUMN");

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

{

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

{

if(prime(arr[i][j]))

System.out.println(" "+arr[i][j]+ "\t"+i+ "\t"+j);

}

}

}

boolean prime(int no)

{

int index;

for(index=2;index<no;index++)

{

if(no%index==0)

break;

}

if(index==no)

return true;

else

return false;

}

public static void main(String args[])throws IOException

{

WondrousSq W=new WondrousSq();

W.WonSq();

}

}

Input:

Enter the size of array(row and column same):

4

Enter the value:16

Enter the value:15

Enter the value:1

Enter the value:2

Enter the value:6

Enter the value:4

Enter the value:10

Enter the value:14

Enter the value:9

Enter the value:8

Enter the value:12

Enter the value:5

Enter the value:3

Enter the value:7

Enter the value:11

Enter the value:13

Output:

The matrix is

16 15 1 2

6 4 10 14

9 8 12 5

3 7 11 13

YES IT REPRESENTS A WONDROUS SQUARE.

PRIME ROW COLUMN

2 0 3

5 2 3

3 3 0

7 3 1

5. Class name : DDA1

Data member : int a[][]=new int[5][5]

DDA1() : Constructor to initialize data members

void Accept() : To accept 5x5 numbers in the array

void Display1() : Display all the elements above the left diagonal

void Display2() : Display all the elements below the left diagonal

void Display3() : Display all the elements above the right diagonal

void Display4() : Display all the elements above the right diagonal

void Swap1() : Swaps the elements above the left diagonal with the elements below the left diagonal

void Swap2() : Swaps the elements above the right diagonal with the elements below the right diagonal

class DDA1

{

int a[][]=new int [5][5];

DDA1()

{

int i,j;

for(i=0;i<5;i++)

{

for(j=0;j<5;j++)

a[i][j]=0;

}

}

void Accept()

{

int i,j,c=11;

for(i=0;i<5;i++)

{

for(j=0;j<5;j++)

a[i][j]=c++;

}

System.out.println("Original Matrix: ");

for(i=0;i<5;i++)

{

for(j=0;j<5;j++)

System.out.print(a[i][j]+" ");

System.out.println("");

}

System.out.println("");

}

void Display1()

{

int i,j;

System.out.println("Display 1: ");

for(i=0;i<5;i++)

{

for(j=i;j<5;j++)

System.out.print(a[i][j]+" ");

System.out.println("");

}

System.out.println("");

}

void Display2()

{

int i,j;

System.out.println("Display 2: ");

for(i=0;i<5;i++)

{

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

System.out.print(a[i][j]+" ");

System.out.println("");

}

System.out.println("");

}

void Display3()

{

int i,j;

System.out.println("Display 3: ");

for(i=0;i<5;i++)

{

for(j=0;j<=(4-i);j++)

System.out.print(a[i][j]+" ");

System.out.println("");

}

System.out.println("");

}

void Display4()

{

int i,j;

System.out.println("Display 4: ");

for(i=0;i<5;i++)

{

for(j=(4-i);j<=4;j++)

System.out.print(a[i][j]+" ");

System.out.println("");

}

System.out.println("");

}

void Swap1()

{

int i,k,t;

System.out.println("Swap 1: ");

for(i=0;i<5;i++)

{

for(k=i+1;k<5;k++)

{

t=a[k][i];

a[k][i]=a[i][k];

a[i][k]=t;

}

}

for(i=0;i<5;i++)

{

for(k=0;k<5;k++)

{

System.out.print(a[i][k]+" ");

}

System.out.println("");

}

System.out.println("");

}

void Swap2()

{

int i,k,t;

System.out.println("Swap 2: ");

for(i=0;i<5;i++)

{

for(k=0;k<4-i;k++)

{

t=a[i][k];

a[i][k]=a[4-i][4-k];

a[4-i][4-k]=t;

}

}

for(i=0;i<5;i++)

{

for(k=0;k<5;k++)

{

System.out.print(a[i][k]+" ");

}

System.out.println("");

}

for(i=0;i<5;i++)

{

for(k=0;k<4-i;k++)

{

t=a[i][k];

a[i][k]=a[4-i][4-k];

a[4-i][4-k]=t;

}

}

System.out.println("");

}

public static void main (String args [])

{

DDA1 D=new DDA1();

D.Accept();

D.Display1();

D.Display2();

D.Display3();

D.Display4();

D.Swap1();

D.Swap2();

}

}

Output:

Original Matrix:

11 12 13 14 15

16 17 18 19 20

21 22 23 24 25

26 27 28 29 30

31 32 33 34 35

Display 1:

11 12 13 14 15

17 18 19 20

23 24 25

29 30

35

Display 2:

11

16 17

21 22 23

26 27 28 29

31 32 33 34 35

Display 3:

11 12 13 14 15

16 17 18 19

21 22 23

26 27

31

Display 4:

15

19 20

23 24 25

27 28 29 30

31 32 33 34 35

Swap 1:

11 16 21 26 31

12 17 22 27 32

13 18 23 28 33

14 19 24 29 34

15 20 25 30 35

Swap 2:

35 30 25 20 31

34 29 24 27 14

33 28 23 18 13

32 19 22 17 12

15 26 21 16 11