using System;

namespace v15

{

class Program

{

public static double GetDouble(string word)

{

Console.Write($"Введите {word}: ");

return Double.Parse(Console.ReadLine().Trim());

}

public static int GetInt(string word)

{

return Convert.ToInt32(GetDouble(word));

}

public static (double[,], int, int) GetEmptyMatrix(int raws = -1, int columns = -1)

{

if (raws == -1) raws = GetInt("строки");

if (columns == -1) columns = GetInt("столбцы");

double[,] matrix = new double[raws, columns];

return (matrix, raws, columns);

}

public static (double[,], int, int) GetMatrix(int width = -1, int height = -1)

{

(double[,] matrix, int raws, int columns) = GetEmptyMatrix(width, height);

for (int i = 0; i < raws; i++)

{

for (int j = 0; j < columns; j++)

{

matrix[i, j] = GetDouble($"{i} строка {j} столбец");

}

}

return (matrix, raws, columns);

}

public static (double[,], int, int) GetRandomMatrix(int start, int end, int width = -1, int heght = -1)

{

(double[,] matrix, int raws, int columns) = GetEmptyMatrix(width, heght);

Random rand = new Random();

for (int i = 0; i < raws; i++)

{

for (int j = 0; j < columns; j++)

{

matrix[i, j] = rand.Next(start, end);

}

}

return (matrix, raws, columns);

}

public static void PrintArr(double[] arr, int start = 0, int end = -1)

{

end = end == -1 ? arr.Length : end;

for (int i = start; i < end; i++)

{

Console.Write(arr[i]);

}

Console.WriteLine();

Console.WriteLine();

}

public static void PrintMatrix(double[,] matrix, int raws, int columns)

{

int[] lengths = new int[columns];

for (int i = 0; i < columns; i++)

{

int maxLen = Convert.ToString(matrix[0, i]).Length;

for (int j = 0; j < raws; j++)

{

int cLen = Convert.ToString(matrix[j, i]).Length;

if (cLen > maxLen) maxLen = cLen;

}

lengths[i] = maxLen;

}

for (int i = 0; i < raws; i++)

{

for (int j = 0; j < columns; j++)

{

int maxLen = lengths[j];

double c = matrix[i, j];

int cLen = Convert.ToString(c).Length;

int d = maxLen - cLen;

for (int k = 0; k < d; k++)

{

Console.Write(" ");

}

Console.Write($" {c} ");

}

Console.WriteLine();

}

Console.WriteLine();

}

public static int FindMin(double[] arr, int start = 0, int end = -1)

{

end = end == -1 ? arr.Length : end;

int idx = 0;

double min = arr[0];

for (int i = start; i < end; i++)

{

if (arr[i] < min)

{

min = arr[i];

idx = i;

}

}

return idx;

}

public static int FindMax(double[] arr, int start = 0, int end = -1)

{

end = end == -1 ? arr.Length : end;

int idx = 0;

double max = arr[0];

for (int i = start; i < end; i++)

{

if (arr[i] > max)

{

max = arr[i];

idx = i;

}

}

return idx;

}

public static int FindFirstPositive(double[] arr, int start = 0, int end = -1)

{

end = end == -1 ? arr.Length : end;

int idx = 0;

double num = arr[idx];

for (int i = start; i < end; i++)

{

if (arr[i] > 0)

{

idx = i;

num = arr[i];

break;

}

}

return num > 0 ? idx : -1;

}

public static int FindFirstNegative(double[] arr, int start = 0, int end = -1)

{

end = end == -1 ? arr.Length : end;

int idx = 0;

double num = arr[idx];

for (int i = start; i < end; i++)

{

if (arr[i] < 0)

{

idx = i;

num = arr[i];

break;

}

}

return num < 0 ? idx : -1;

}

public static int FindLastPositive(double[] arr, int start = 0, int end = -1)

{

end = end == -1 ? arr.Length : end;

int idx = 0;

double num = arr[idx];

for (int i = end - 1; i >= start; i--)

{

if (arr[i] > 0)

{

idx = i;

num = arr[i];

break;

}

}

return num > 0 ? idx : -1;

}

public static int FindLastNegative(double[] arr, int start = 0, int end = -1)

{

end = end == -1 ? arr.Length : end;

int idx = end - 1;

double num = arr[idx];

for (int i = end - 1; i >= start; i--)

{

if (arr[i] < 0)

{

idx = i;

num = arr[i];

break;

}

}

return num < 0 ? idx : -1;

}

public static double[] GetRaw(double[,] matrix, int raw, int columns)

{

double[] arr = new double[columns];

for (int i = 0; i < columns; i++)

{

arr[i] = matrix[raw, i];

}

return arr;

}

public static double[] GetColumn(double[,] matrix, int column, int raws)

{

double[] arr = new double[raws];

for (int i = 0; i < raws; i++)

{

arr[i] = matrix[i, column];

}

return arr;

}

public static void SwapColumns(double[,] matrix, int from, int raws, int to = -1)

{

if (to == -1) to = from + 1;

double[] temp = GetColumn(matrix, to, raws);

for (int i = 0; i < raws; i++)

{

matrix[i, to] = matrix[i, from];

matrix[i, from] = temp[i];

}

}

public static void Task1()

{

//чтобы получить матрицу с ручным вводом заменить GetRandomMatrix на GetMatrix

(double[,] matrix, int raws, int columns) = Program.GetRandomMatrix(-2, 5, 3, 4);

Program.PrintMatrix(matrix, raws, columns);

//Обход по строкам

for (int i = 0; i < raws; i++)

{

double[] raw = GetRaw(matrix, i, columns);

int firstPositive = FindFirstPositive(raw);

//Если 1 положительный 1 элемент строки

//То до него ничего нет и поэтому пропускаем

//Или вся строка отрицательная

if (firstPositive <= 0) continue;

int min = FindMin(raw, 0, firstPositive);

int lastNegative = FindLastNegative(raw);

//Значит нет в строке отрицательного

if (lastNegative < 0) continue;

//Долгожданная замена

matrix[i, min] = matrix[i, lastNegative];

matrix[i, lastNegative] = raw[min];

}

Program.PrintMatrix(matrix, raws, columns);

}

public static void Task2()

{

//чтобы получить матрицу с ручным вводом заменить GetRandomMatrix на GetMatrix

(double[,] matrix, int raws, int columns) = Program.GetRandomMatrix(-2, 5, 4, 5);

Program.PrintMatrix(matrix, raws, columns);

double[] negatives = new double[columns];

//Пойдем по столбцам посчитаем отрицательные в каждом

for (int i = 0; i < columns; i++)

{

int n = 0;

for (int j = 0; j < raws; j++)

{

if (matrix[j, i] < 0) n++;

}

negatives[i] = n;

}

PrintArr(negatives);

//Теперь сортируя массив с отрицательными

//Будем перествавлять столбцы

double temp;

for (int i = 0; i < negatives.Length - 1; i++)

{

for (int j = i + 1; j < negatives.Length; j++)

{

if (negatives[i] > negatives[j])

{

//Тут меняем числа массива

temp = negatives[i];

negatives[i] = negatives[j];

negatives[j] = temp;

//Тут меняем столбцы

SwapColumns(matrix, i, raws, j);

}

}

}

PrintArr(negatives);

Program.PrintMatrix(matrix, raws, columns);

}

static void Main(string[] args)

{

Task1();

Console.WriteLine("================");

Task2();

}

}

}