Министерство образования Российской Федерации

Пензенский государственный университет

Кафедра «Системы автоматизированного проектирования»

**Отчет**

По лабораторной работе №6

По курсу «Автоматизация конструкторского проектирования ЭА»

На тему «Разработка программы последовательного алгоритма размещения одногабаритных элементов на печатной плате»

Выполнил студент гр.20ВВ2.1

Горбунов Н.А.  
Исаев С.Д.  
Кривцов Н.А.

Проверили:

Гудков А.А.  
Гудков П.А.

Пенза, 2022

Листинг

using System;

using System.Collections;

using System.Collections.Generic;

namespace Lab6

{

class Program

{

public static List<int> busyStrings = new List<int>();

const int CountOfPodGraphs = 3;

static void Main(string[] args)

{

//int[,] Matrix = new int[15, 15];

/\*

int[] X1 = { 0, 3, 1, 4, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };

int[] X2 = { 3, 0, 0, 1, 2, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0 };

int[] X3 = { 1, 0, 0, 0, 0, 2, 2, 2, 0, 0, 0, 0, 0, 0, 0 };

int[] X4 = { 4, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 1 };

int[] X5 = { 3, 2, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 1 };

int[] X6 = { 0, 0, 2, 0, 0, 0, 1, 0, 0, 2, 0, 0, 0, 1, 0 };

int[] X7 = { 0, 0, 2, 0, 0, 1, 0, 1, 1, 3, 0, 0, 2, 0, 0 };

int[] X8 = { 0, 0, 2, 0, 0, 0, 1, 0, 0, 4, 0, 0, 0, 0, 0 };

int[] X9 = { 0, 1, 0, 0, 2, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0 };

int[] X10 = { 0, 0, 0, 0, 0, 2, 3, 4, 1, 0, 0, 0, 0, 0, 0 };

int[] X11 = { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 3 };

int[] X12 = { 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 2, 0, 0, 1, 3 };

int[] X13 = { 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 1 };

int[] X14 = { 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 3 };

int[] X15 = { 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 3, 3, 1, 3, 0 };

\*/

int[,] Matrix = {

{ 0, 3, 1, 4, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },

{ 3, 0, 0, 1, 2, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0 },

{ 1, 0, 0, 0, 0, 2, 2, 2, 0, 0, 0, 0, 0, 0, 0 },

{ 4, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 1 },

{ 3, 2, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 1 },

{ 0, 0, 2, 0, 0, 0, 1, 0, 0, 2, 0, 0, 0, 1, 0 },

{ 0, 0, 2, 0, 0, 1, 0, 1, 1, 3, 0, 0, 2, 0, 0 },

{ 0, 0, 2, 0, 0, 0, 1, 0, 0, 4, 0, 0, 0, 0, 0 },

{ 0, 1, 0, 0, 2, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0 },

{ 0, 0, 0, 0, 0, 2, 3, 4, 1, 0, 0, 0, 0, 0, 0 },

{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 3 }, // 10

{ 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 2, 0, 0, 1, 3 },

{ 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 1 },

{ 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 3 },

{ 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 3, 3, 1, 3, 0 } // 14

};

Alghoritm(Matrix);

}

static void Alghoritm(int[,] Matrix)

{

int countInOnePodraphs = (int)(Matrix.GetLength(0) / CountOfPodGraphs);

List<List<int>> result = new List<List<int>>();

for (int ii = 0; ii < CountOfPodGraphs; ii++)

{

// Нашли сумму каждой строки

int[] sumEachString = new int[Matrix.GetLength(1)];

for (int i = 0; i < sumEachString.Length; i++)

{

sumEachString[i] = GetSumStringWithoutBusyString(Matrix, i);

}

int maxIndex = GetMaxIndex(sumEachString);

int maxIndexOfStolbec = GetMaxIndexOfStroke(Matrix, maxIndex);

busyStrings.Add(maxIndexOfStolbec);

while (busyStrings.Count < countInOnePodraphs \* (ii + 1))

{

GetMaxResOfBusy(Matrix);

}

List<int> resultOfCycle = new List<int>();

for (int i = countInOnePodraphs \* ii; i < busyStrings.Count; i++)

{

resultOfCycle.Add(busyStrings[i]);

}

result.Add(resultOfCycle);

PrintResult(resultOfCycle);

}

}

public static void PrintResult(List<int> resultOfCycle)

{

for (int i = 0; i < resultOfCycle.Count; i++)

{

Console.Write($"{resultOfCycle[i]} ");

}

Console.WriteLine("\n");

}

public static void GetMaxResOfBusy(int[,] Matrix)

{

int maxRes = -1;

int maxIndex = -1;

for (int i = 0; i < Matrix.GetLength(1); i++)

{

int sum = 0;

if (busyStrings.Contains(i)) continue;

for (int j= 0; j< busyStrings.Count; j++)

{

sum += Matrix[busyStrings[j], i];

}

if(sum > maxRes)

{

maxRes = sum;

maxIndex = i;

}

}

busyStrings.Add(maxIndex);

}

/// <summary>

/// Сумма строки с учетом Busy

/// </summary>

/// <returns></returns>

public static int GetSumString(int[,] Matrix, int numOfStroke)

{

int res = 0;

for (int i = 0; i < Matrix.GetLength(1); i++)

{

if(!busyStrings.Contains(i))

res += Matrix[numOfStroke,i];

}

return res;

}

/// <summary>

/// Сумма строки Без учета Busy

/// </summary>

/// <returns></returns>

public static int GetSumStringWithoutBusyString(int[,] Matrix, int numOfStroke)

{

int res = 0;

for (int i = 0; i < Matrix.GetLength(1); i++)

{

res += Matrix[numOfStroke, i];

}

return res;

}

public static int GetMaxIndex(int[] array)

{

int maxRes = -1;

int maxIndex = -1;

for (int i = 0; i < array.Length; i++)

{

if (busyStrings.Contains(i)) continue;

if(maxRes < array[i])

{

maxRes = array[i];

maxIndex = i;

}

}

return maxIndex;

}

public static int GetMaxIndexOfStroke(int[,] Matrix, int numOfStroke)

{

int maxRes = -1;

int maxIndex = -1;

for (int i = 0; i < Matrix.GetLength(1); i++)

{

if (busyStrings.Contains(i)) continue;

if (maxRes < Matrix[numOfStroke,i])

{

maxRes = Matrix[numOfStroke, i];

maxIndex = i;

}

}

// Добавляем строку в "занятые строки"

busyStrings.Add(numOfStroke);

int minSum = int.MaxValue;

int minSumIndex = -1;

for (int i = 0; i < Matrix.GetLength(1); i++)

{

if (maxRes == Matrix[numOfStroke, i])

{

int uuuu = GetSumString(Matrix, i);

if(minSum > uuuu)

{

minSum = uuuu;

minSumIndex = i;

}

}

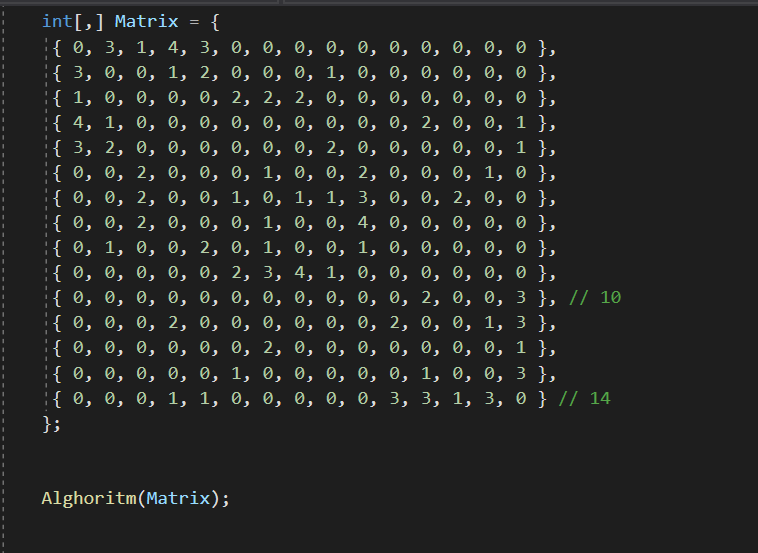
}

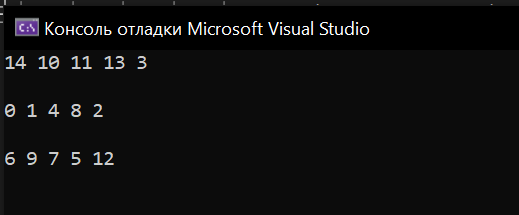
return minSumIndex;

}

}

}





Вывод: Написало, шо, скрипт код программу, разделяет граф с любым кол-вом вершин на любое колзво подграфов.