|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ «Информатика и системы управления»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Лабораторная работа № \_\_**1**\_\_**

**Дисциплина Методы вычислений**

|  |  |
| --- | --- |
| **Тема Венгерский метод решения задачи о назначениях**  **Вариант №2**  **Студент \_Блохин Д.М.\_\_\_\_\_\_\_\_**  **Группа \_ИУ7-11М\_\_\_\_\_\_\_\_\_\_\_\_\_**  **Оценка (баллы) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **Преподаватель \_Власов П.А.** |  |

Москва.

2023 г.

**Цель работы:** изучение венгерского метода решения задачи о назначениях.

**Содержательная и математическая постановка задачи**

В распоряжении работодателя имеется n работ и n исполнителей. Стоимость выполнения i-ой работы j-ым исполнителем составляет единиц.

Требуется распределить все работы между исполнителями так, чтобы:

1. каждый исполнитель выполнял ровно 1 работу;
2. общая стоимость выполнения всех работ была минимальна.

.

Матрица назначений:

Введём управляющие переменные:

,

.

Общая стоимость выполнения всех работ:

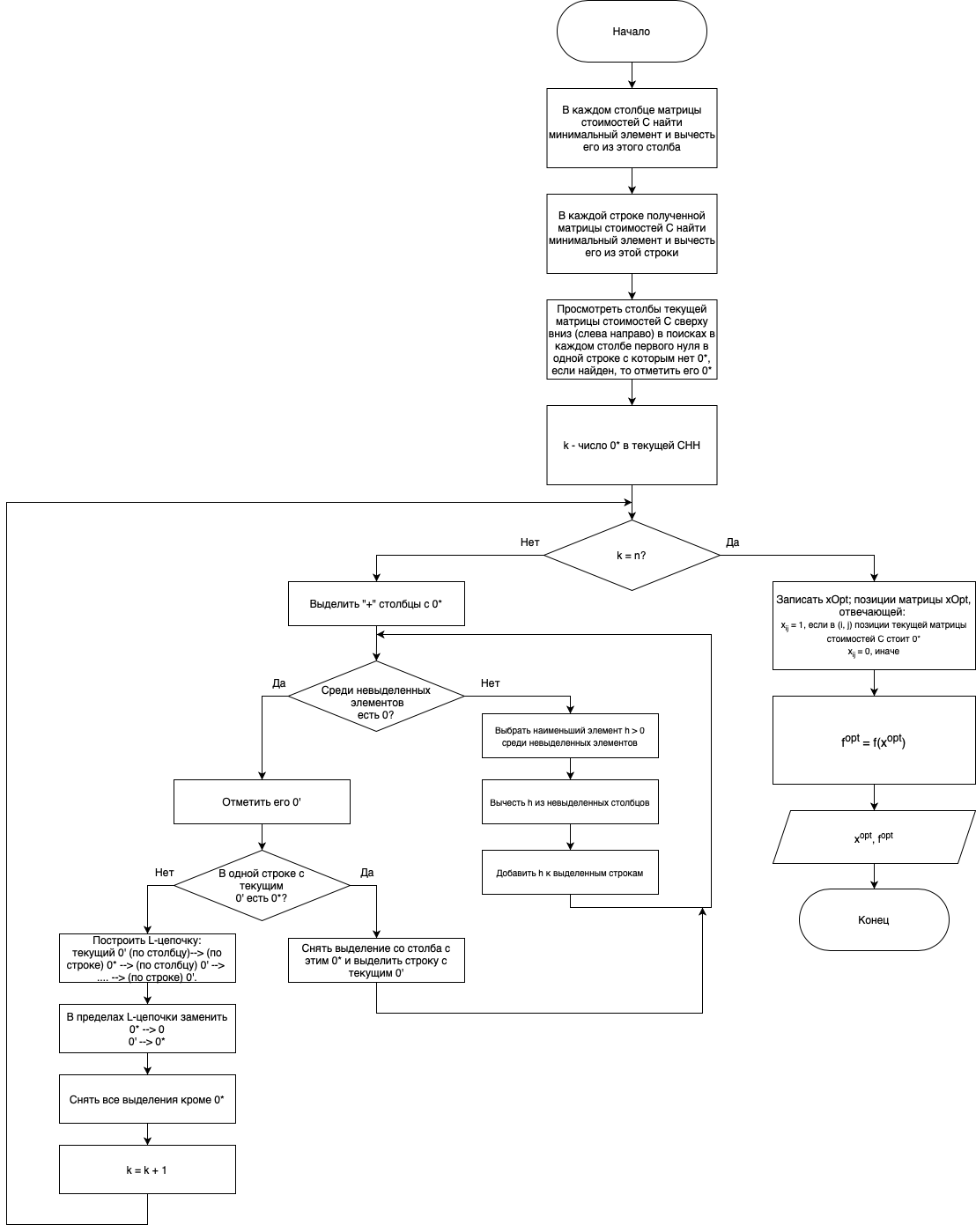
Условие того, что j-ый исполнитель выполняет ровно 1 работу:

Условие того, что i-ую работу выполняет ровно 1 исполнитель:

Таким образом, математическая постановка задачи о назначениях:

**Вариант 2**

C =

Схема алгоритма: 

Текст программы представлен на Листинге 1

Листинг 1

|  |
| --- |
| function lab01()  clc;  debugFlag = 1;  maximizationFlag = 0;  matr = [  4 10 10 3 6;  5 6 2 7 4;  9 5 6 8 3;  2 3 5 4 8;  8 5 4 9 3];  disp('2 вариант. Матрица:');  disp(matr);  C = matr;  if maximizationFlag == 1  C = convertToMin(matr);  if debugFlag == 1  disp('Матрица после приведения к задаче минимизации:');  disp(C);  end  end  % 1.1 Вычитание из каждого столбца матрицы наименьшего элемента этого столбца  C = updateColumns(C);  if debugFlag == 1  disp('Матрица после преобразования столбцов:');  disp(C);  end  % 1.2 Вычитание из каждой строки матрицы наименьшего элемента этой строки  C = updateRows(C);  if debugFlag == 1  disp('Матрица после преобразования строк:');  disp(C);  end  % 1.3 Построение начальной СНН  [numColumns, numRows] = size(C);  addMatr = initAddMatr(numRows, numColumns, C);  % 1.4 Вывод начальной СНН  if debugFlag == 1  disp('Начальная СНН:');  printCurMatr(C, addMatr, numRows, numColumns);  end  % 1.5 подсчет 0\* в текущей СНН  k = sum(addMatr, 'all');  if debugFlag == 1  fprintf('Число нулей в построенной СНН: k = %d\n\n', k);  end  iteration = 1;  while k < numRows  if debugFlag == 1  fprintf('---------------- Итерация №%d ----------------\n', iteration);  end  % Матрица для выделения 0'  primeMatr = zeros(numRows, numColumns);  % Массивы для выделения строк и столбцов  selectedRows = zeros(numRows);  selectedColumns = sum(addMatr);  % Выделение столбцов с 0\*  selection = getSelection(numRows, numColumns, selectedColumns);  if debugFlag == 1  disp('Результат выделения столбцов, в которых стоит 0\*:');  printMarkedMatr(C, addMatr, primeMatr, selectedColumns, selectedRows, numRows, numColumns);  end  flag = true;  primePnt = [-1 -1];  while flag  if debugFlag == 1  disp('Поиск 0 среди невыделенных элементов');  end  % 2.1 поиск невыделенных 0  primePnt = findPrime(C, selection, numRows, numColumns);  if primePnt(1) == -1  C = updateMatrNoZero(C, numRows, numColumns, selection, selectedRows, selectedColumns);  if debugFlag == 1  disp('Т.к. среди невыделенных элементов нет нулей, матрица была преобразована:');  printMarkedMatr(C, addMatr, primeMatr, selectedColumns, selectedRows, numRows, numColumns);  end  primePnt = findPrime(C, selection, numRows, numColumns);  end  primeMatr(primePnt(1), primePnt(2)) = 1;  if debugFlag == 1  disp('Матрица с найденным 0-штрих');  printMarkedMatr(C, addMatr, primeMatr, selectedColumns, selectedRows, numRows, numColumns);  end  zeroWithStarInRow = getZeroWithStarInRow(primePnt, numColumns, addMatr);  if zeroWithStarInRow(1) == -1  flag = false;  else  % Перенос выделения со столбца на строку  selection(:, zeroWithStarInRow(2)) = selection(:, zeroWithStarInRow(2)) - 1;  selectedColumns(zeroWithStarInRow(2)) = 0;    selection(zeroWithStarInRow(1), :) = selection(zeroWithStarInRow(1), :) + 1;  selectedRows(zeroWithStarInRow(1)) = 1;  if debugFlag == 1  disp("Т.к. в одной строке с 0' есть 0\*, было переброшено выделение:");  printMarkedMatr(C, addMatr, primeMatr, selectedColumns, selectedRows, numRows, numColumns);  end  end  end  if debugFlag == 1  disp('L-цепочка: ');  end  [primeMatr, addMatr] = createL(numRows, numColumns, primePnt, primeMatr, addMatr);  k = sum(addMatr, 'all');  if debugFlag == 1  disp('Текущая СНН:');  printCurMatr(C, addMatr, numRows, numColumns);  fprintf('Итого, k = %d\n', k);  end    iteration = iteration + 1;  disp('----------------------------------------------');  end  if debugFlag == 1  disp('Конечная СНН:');  printCurMatr(C, addMatr, numRows, numColumns);  end  disp('X =');  disp(addMatr);  fOpt = getFOpt(matr, addMatr, numRows, numColumns);  fprintf('Ответ: fOpt = %d\n', fOpt);  end  % Приведение задачи максимизации к задаче минимизации  function matr = convertToMin(matr)  maxEl = max(max(matr));  matr = matr \* (-1) + maxEl;  end  % Начальное изменение столбцов  function matr = updateColumns(matr)  minEl = min(matr);  for i = 1 : length(minEl)  matr(:, i) = matr(:, i) - minEl(i);  end  end  % Начальное изменение строк  function matr = updateRows(matr)  minEl = min(matr,[],2);  for i = 1 : length(minEl)  matr(i, :) = matr(i, :) - minEl(i);  end  end  % Создание СНН  function addMatr = initAddMatr(numRows, numColumns, matr)  addMatr = zeros(numRows, numColumns);  for j = 1 : numColumns  for i = 1 : numRows  if matr(i, j) == 0  counter = 0;  for k = 1 : numColumns  counter = counter + addMatr(i, k);  end  for k = 1 : numRows  counter = counter + addMatr(k, j);  end  if counter == 0  addMatr(i, j) = 1;  end  end  end  end  end  % Вывод СНН  function [] = printCurMatr(matr, addMatr, numRows, numColumns)  fprintf("\n");  for i = 1 : numRows  for j = 1 : numColumns  if addMatr(i, j) == 1  fprintf("\t%d\*\t", matr(i, j));  else  fprintf("\t%d\t", matr(i, j));  end  end  fprintf("\n");  end  fprintf("\n");  end  % Выделение столбцов с 0\*  function [selection] = getSelection(numRows, numColumns, selectedColumns)  selection = zeros(numRows, numColumns);  for i = 1 : numColumns  if selectedColumns(i) == 1  selection(:, i) = selection(:, i) + 1;  end  end  end  % Вывод текущей СНН с выделениями столбцов и строк  function [] = printMarkedMatr(matr, addMatr, primeMatr, selectedColumns, selectedRows, numRows, numColumns)  for i = 1 : numRows  if selectedRows(i) == 1  fprintf("+");  end  for j = 1 : numColumns  fprintf("\t%d", matr(i, j));  if addMatr(i, j) == 1  fprintf("\*\t");  elseif primeMatr(i, j) == 1  fprintf("'\t");  else  fprintf("\t");  end  end  fprintf('\n');  end  for i = 1 : numColumns  if selectedColumns(i) == 1  fprintf("\t+\t")  else  fprintf(" \t\t")  end  end  fprintf('\n\n');  end  % Нахождение невыделенного 0  function [primePnt] = findPrime(matr, selection, numRows, numColumns)  primePnt = [-1 -1];  for j = 1 : numColumns  for i = 1 : numRows  if matr(i, j) == 0 && selection(i, j) == 0  primePnt(1) = i;  primePnt(2) = j;  return;  end  end  end  end  % Изменение матрицы в случае, если среди невыделенных элементов нет нуля  function [matr] = updateMatrNoZero(matr, numRows, numColumns, selection, selectedRows, selectedColumns)  h = 1e5; % Наименьший элемент среди невыделенных  for j = 1 : numColumns  for i = 1 : numRows  if selection(i, j) == 0 && matr(i, j) < h  h = matr(i, j);  end  end  end  for j = 1 : numColumns  if selectedColumns(j) == 0  matr(:, j) = matr(:, j) - h;  end  end  for i = 1 : numRows  if selectedRows(i) == 1  matr(i, :) = matr(i, :) + h;  end  end  end  % Нахождение 0\* в одной строке с 0'  function [zeroWithStarInRow] = getZeroWithStarInRow(primePnt, numColumns, addMatr)  primeI = primePnt(1);  zeroWithStarInRow = [-1 -1];  for j = 1 : numColumns  if addMatr(primeI, j) == 1  zeroWithStarInRow(1) = primeI;  zeroWithStarInRow(2) = j;  break;  end  end  end  % Построение L-цепочки  function [primeMatr, addMatr] = createL(numRows, numColumns, primePnt, primeMatr, addMatr)  i = primePnt(1);  j = primePnt(2);  while i > 0 && j > 0 && i <= numRows && j <= numColumns  % Снятие '  primeMatr(i, j) = 0;  % Замена ' на \*  addMatr(i, j) = 1;  fprintf("[%d, %d] ", i, j);  % Дойти до 0\* по столбцу от 0'  kRow = 1;  while kRow <= numRows && (addMatr(kRow, j) ~= 1 || kRow == i)  kRow = kRow + 1;  end  if (kRow <= numRows)  % Дойти до 0' по строке от 0\*  lCol = 1;  while lCol <= numColumns && (primeMatr(kRow, lCol) ~= 1 || lCol == j)  lCol = lCol + 1;  end  if lCol <= numColumns  addMatr(kRow,j) = 0;  fprintf("-> [%d, %d] -> ", kRow, j);  end  j = lCol;  end  i = kRow;  end  fprintf("\n");  end  % Вычисление fOpt  function fOpt = getFOpt(matr, addMatr, numRows, numColumns)  fOpt = 0;  for i = 1 : numRows  for j = 1 : numColumns  if addMatr(i, j) == 1  fOpt = fOpt + matr(i, j);  end  end  end  end |

**Результаты расчетов для задач из индивидуального варианта.**

Задача минимизации

|  |
| --- |
| 2 вариант. Матрица:  4 10 10 3 6  5 6 2 7 4  9 5 6 8 3  2 3 5 4 8  8 5 4 9 3  Матрица после преобразования столбцов:  2 7 8 0 3  3 3 0 4 1  7 2 4 5 0  0 0 3 1 5  6 2 2 6 0  Матрица после преобразования строк:  2 7 8 0 3  3 3 0 4 1  7 2 4 5 0  0 0 3 1 5  6 2 2 6 0  Начальная СНН:  2 7 8 0\* 3  3 3 0\* 4 1  7 2 4 5 0\*  0\* 0 3 1 5  6 2 2 6 0  Число нулей в построенной СНН: k = 4  ---------------- Итерация №1 ----------------  Результат выделения столбцов, в которых стоит 0\*:  2 7 8 0\* 3  3 3 0\* 4 1  7 2 4 5 0\*  0\* 0 3 1 5  6 2 2 6 0  + + + +  Поиск 0 среди невыделенных элементов  Матрица с найденным 0-штрих  2 7 8 0\* 3  3 3 0\* 4 1  7 2 4 5 0\*  0\* 0' 3 1 5  6 2 2 6 0  + + + +  Т.к. в одной строке с 0' есть 0\*, было переброшено выделение:  2 7 8 0\* 3  3 3 0\* 4 1  7 2 4 5 0\*  + 0\* 0' 3 1 5  6 2 2 6 0  + + +  Поиск 0 среди невыделенных элементов  Т.к. среди невыделенных элементов нет нулей, матрица была преобразована:  0 5 8 0\* 3  1 1 0\* 4 1  5 0 4 5 0\*  + 0\* 0' 5 3 7  4 0 2 6 0  + + +  Матрица с найденным 0-штрих  0' 5 8 0\* 3  1 1 0\* 4 1  5 0 4 5 0\*  + 0\* 0' 5 3 7  4 0 2 6 0  + + +  Т.к. в одной строке с 0' есть 0\*, было переброшено выделение:  + 0' 5 8 0\* 3  1 1 0\* 4 1  5 0 4 5 0\*  + 0\* 0' 5 3 7  4 0 2 6 0  + +  Поиск 0 среди невыделенных элементов  Матрица с найденным 0-штрих  + 0' 5 8 0\* 3  1 1 0\* 4 1  5 0' 4 5 0\*  + 0\* 0' 5 3 7  4 0 2 6 0  + +  Т.к. в одной строке с 0' есть 0\*, было переброшено выделение:  + 0' 5 8 0\* 3  1 1 0\* 4 1  + 5 0' 4 5 0\*  + 0\* 0' 5 3 7  4 0 2 6 0  +  Поиск 0 среди невыделенных элементов  Матрица с найденным 0-штрих  + 0' 5 8 0\* 3  1 1 0\* 4 1  + 5 0' 4 5 0\*  + 0\* 0' 5 3 7  4 0' 2 6 0  +  L-цепочка:  [5, 2]  Текущая СНН:  0 5 8 0\* 3  1 1 0\* 4 1  5 0 4 5 0\*  0\* 0 5 3 7  4 0\* 2 6 0  Итого, k = 5  ----------------------------------------------  Конечная СНН:  0 5 8 0\* 3  1 1 0\* 4 1  5 0 4 5 0\*  0\* 0 5 3 7  4 0\* 2 6 0  X =  0 0 0 1 0  0 0 1 0 0  0 0 0 0 1  1 0 0 0 0  0 1 0 0 0  Ответ: fOpt = 15 |

Задача максимизации

|  |
| --- |
| 2 вариант. Матрица:  4 10 10 3 6  5 6 2 7 4  9 5 6 8 3  2 3 5 4 8  8 5 4 9 3  Матрица после приведения к задаче минимизации:  6 0 0 7 4  5 4 8 3 6  1 5 4 2 7  8 7 5 6 2  2 5 6 1 7  Матрица после преобразования столбцов:  5 0 0 6 2  4 4 8 2 4  0 5 4 1 5  7 7 5 5 0  1 5 6 0 5  Матрица после преобразования строк:  5 0 0 6 2  2 2 6 0 2  0 5 4 1 5  7 7 5 5 0  1 5 6 0 5  Начальная СНН:  5 0\* 0 6 2  2 2 6 0\* 2  0\* 5 4 1 5  7 7 5 5 0\*  1 5 6 0 5  Число нулей в построенной СНН: k = 4  ---------------- Итерация №1 ----------------  Результат выделения столбцов, в которых стоит 0\*:  5 0\* 0 6 2  2 2 6 0\* 2  0\* 5 4 1 5  7 7 5 5 0\*  1 5 6 0 5  + + + +  Поиск 0 среди невыделенных элементов  Матрица с найденным 0-штрих  5 0\* 0' 6 2  2 2 6 0\* 2  0\* 5 4 1 5  7 7 5 5 0\*  1 5 6 0 5  + + + +  Т.к. в одной строке с 0' есть 0\*, было переброшено выделение:  + 5 0\* 0' 6 2  2 2 6 0\* 2  0\* 5 4 1 5  7 7 5 5 0\*  1 5 6 0 5  + + +  Поиск 0 среди невыделенных элементов  Т.к. среди невыделенных элементов нет нулей, матрица была преобразована:  + 7 0\* 0' 8 4  2 0 4 0\* 2  0\* 3 2 1 5  7 5 3 5 0\*  1 3 4 0 5  + + +  Матрица с найденным 0-штрих  + 7 0\* 0' 8 4  2 0' 4 0\* 2  0\* 3 2 1 5  7 5 3 5 0\*  1 3 4 0 5  + + +  Т.к. в одной строке с 0' есть 0\*, было переброшено выделение:  + 7 0\* 0' 8 4  + 2 0' 4 0\* 2  0\* 3 2 1 5  7 5 3 5 0\*  1 3 4 0 5  + +  Поиск 0 среди невыделенных элементов  Матрица с найденным 0-штрих  + 7 0\* 0' 8 4  + 2 0' 4 0\* 2  0\* 3 2 1 5  7 5 3 5 0\*  1 3 4 0' 5  + +  L-цепочка:  [5, 4] -> [2, 4] -> [2, 2] -> [1, 2] -> [1, 3]  Текущая СНН:  7 0 0\* 8 4  2 0\* 4 0 2  0\* 3 2 1 5  7 5 3 5 0\*  1 3 4 0\* 5  Итого, k = 5  ----------------------------------------------  Конечная СНН:  7 0 0\* 8 4  2 0\* 4 0 2  0\* 3 2 1 5  7 5 3 5 0\*  1 3 4 0\* 5  X =  0 0 1 0 0  0 1 0 0 0  1 0 0 0 0  0 0 0 0 1  0 0 0 1 0  Ответ: fOpt = 42 |