|  |  |
| --- | --- |
| 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 №3**  **Студент \_Брянская Е.В.\_\_\_\_\_\_\_\_**  **Группа \_ИУ7-11М\_\_\_\_\_\_\_\_\_\_\_\_\_**  **Оценка (баллы) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **Преподаватель \_Власов П.А.** |  |

Москва.

2022 г.

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

**Содержательная и математическая постановка задачи**

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

Требуется распределить все работы между исполнителями так, чтобы каждый из них выполнял ровно 1 работу. А общая стоимость выполнения всех работ была минимальна.

.

Матрица назначений:

Введём управляющие переменные:

,

.

Общая стоимость всех работ:

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

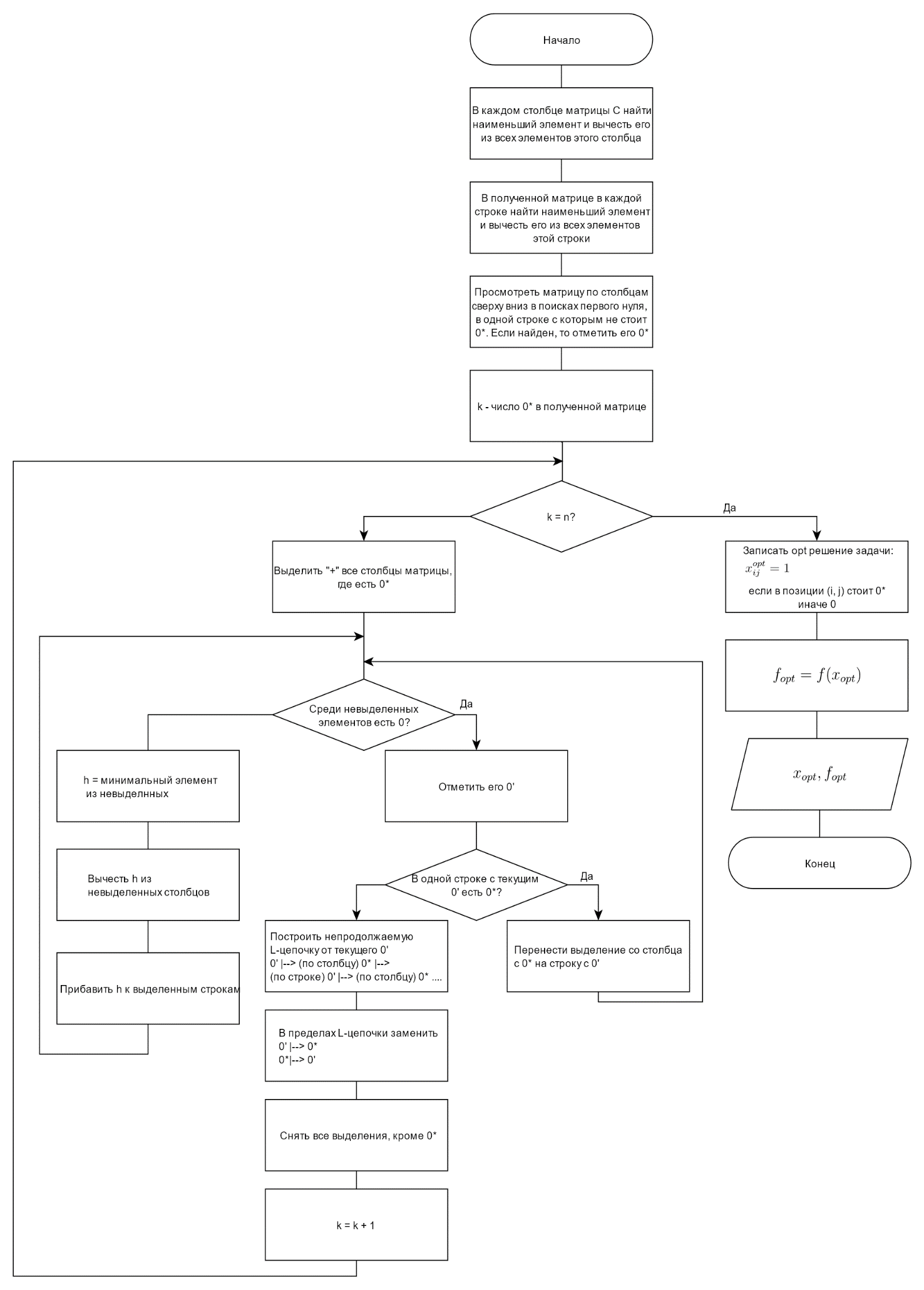
Условие того, что i-ую работу выполняет ровно 1 исполнитель:

Таким образом, математическая постановка задачи о назначениях:

**Вариант 3**

C =

Схема алгоритма:



Текст программы представлен на Листинге 1

Листинг 1

|  |
| --- |
| function lab1()  clc;  debugFlg = 1;  findMax = 0;  matr = [  1 4 7 9 4;  9 3 8 7 4;  3 4 6 8 2;  8 2 4 6 7;  7 6 9 8 5];  disp('3 вариант. Матрица:');  disp(matr);  C = matr;  if findMax == 1  C = convertToMin(matr);  if debugFlg == 1  disp('Матрица после приведения к задаче минимизации:');  disp(C);  end  end  C = updateColumns(C);  if debugFlg == 1  disp('Результат вычитания наименьшего элемента по столбцам:');  disp(C);  end  C = updateRows(C);  if debugFlg == 1  disp('Результат вычитания наименьшего элемента по строкам:');  disp(C);  end  [numRows,numCols] = size(C);  matrSIZ = getSIZInit(C);  if debugFlg == 1  disp('Начальная СНН:');  printSIZ(C, matrSIZ);  end  k = sum(matrSIZ, 'all');  if debugFlg == 1  fprintf('Число нулей в построенной СНН: k = %d\n\n', k);  end  while k < numCols  matrStreak = zeros(numRows, numCols);  selectedColumns = sum(matrSIZ);  selectedRows = zeros(numRows);  selection = getSelection(numRows, numCols, selectedColumns);  if debugFlg == 1  disp('Результат выделения столбцов, в которых стоит 0\*:');  printMarkedMatr(C, matrSIZ, matrStreak, selectedColumns, selectedRows);  end  flag = true;  streakPnt = [-1 -1];  while flag  if debugFlg == 1  disp('Поиск 0 среди невыделенных элементов');  end  streakPnt = findStreak(C, selection);  if streakPnt(1) == -1  C = updateMatrNoZero(C, numRows, numCols, selection, selectedRows, selectedColumns);  if debugFlg == 1  disp('Т.к. среди невыделенных элементов нет нулей, матрица была преобразована:');  printMarkedMatr(C, matrSIZ, matrStreak, selectedColumns, selectedRows);  end  streakPnt = findStreak(C, selection);  end  matrStreak(streakPnt(1), streakPnt(2)) = 1;  if debugFlg == 1  disp('Матрица с найденным 0"');  printMarkedMatr(C, matrSIZ, matrStreak, selectedColumns, selectedRows);  end  zeroStarInRow = getZeroStarInRow(streakPnt, numCols, matrSIZ);  if zeroStarInRow(1) == -1  flag = false;  else  selection(:, zeroStarInRow(2)) = selection(:, zeroStarInRow(2)) - 1;  selectedColumns(zeroStarInRow(2)) = 0;  selection(zeroStarInRow(1), :) = selection(zeroStarInRow(1), :) + 1;  selectedRows(zeroStarInRow(1)) = 1;  if debugFlg == 1  disp('Т.к. в одной строке с 0" есть 0\*, было переброшено выделение:');  printMarkedMatr(C, matrSIZ, matrStreak, selectedColumns, selectedRows);  end  end  end  if debugFlg == 1  disp('L-цепочка: ');  end  [matrStreak, matrSIZ] = createL(numRows, numCols, streakPnt, matrStreak, matrSIZ);  k = sum(matrSIZ, 'all');  if debugFlg == 1  disp('Текущая СНН:');  printSIZ(C, matrSIZ);  fprintf('Итого, k = %d\n', k);  end  end  disp('Конечная СНН:');  printSIZ(C, matrSIZ);  disp('X =');  disp(matrSIZ);  fOpt = getFOpt(matr, matrSIZ);  fprintf("Результат = %d\n", fOpt);  end  function [streakPnt] = findStreak(matr, selection)  streakPnt = [-1 -1];  [numRows,numCols] = size(matr);  for i = 1 : numCols  for j = 1 : numRows  if selection(j, i) == 0 && matr(j, i) == 0  streakPnt(1) = j;  streakPnt(2) = i;  return;  end  end  end  end  function [] = printSIZ(matr, matrSIZ)  [numRows,numCols] = size(matr);  for i = 1 : numRows  for j = 1 : numCols  if matrSIZ(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  function [] = printMarkedMatr(matr, matrSIZ, matrStreak, selectedCols, selectedRows)  [numRows,numCols] = size(matr);  for i = 1 : numRows  if selectedRows(i) == 1  fprintf("+")  end  for j = 1 : numCols  fprintf("\t%d", matr(i, j))  if matrSIZ(i, j) == 1  fprintf("\*\t");  elseif matrStreak(i, j) == 1  fprintf("'\t")  else  fprintf("\t");  end  end  fprintf('\n');  end  for i = 1 : numCols  if selectedCols(i) == 1  fprintf("\t+\t")  else  fprintf(" \t\t")  end  end  fprintf('\n\n');  end  function matr = convertToMin(matr)  maxElem = max(max(matr));  matr = matr \* (-1) + maxElem;  end  function matr = updateColumns(matr)  minElemArr = min(matr);  for i = 1 : length(minElemArr)  matr(:, i) = matr(:, i) - minElemArr(i);  end  end  function matr = updateRows(matr)  minElemArr = min(matr, [], 2);  for i = 1 : length(minElemArr)  matr(i, :) = matr(i, :) - minElemArr(i);  end  end  function matrSIZ = getSIZInit(matr)  [numRows,numCols] = size(matr);  matrSIZ = zeros(numRows, numCols);  for i = 1: numCols  for j = 1 : numRows  if matr(j, i) == 0  count = 0;  for k = 1 : numCols  count = count + matrSIZ(j, k);  end  for k = 1 : numRows  count = count + matrSIZ(k, i);  end  if count == 0  matrSIZ(j, i) = 1;  end  end  end  end  end  function [selection] = getSelection(numRows, numCols, selectedColumns)  selection = zeros(numRows, numCols);  for i = 1 : numCols  if selectedColumns(i) == 1  selection(:, i) = selection(:, i) + 1;  end  end  end  function [matr] = updateMatrNoZero(matr, numRows, numCols, selection, selectedRows, selectedColumns)  h = 1e5; % Наименьший элемент среди невыделенных  for i = 1 : numCols  for j = 1 : numRows  if selection(j, i) == 0 && matr(j, i) < h  h = matr(j, i);  end  end  end  for i = 1 : numCols  if selectedColumns(i) == 0  matr(:, i) = matr(:, i) - h;  end  end  for i = 1 : numRows  if selectedRows(i) == 1  matr(i, :) = matr(i, :) + h;  end  end  end  function [zeroStarInRow] = getZeroStarInRow(streakPnt, numCols, matrSIZ)  j = streakPnt(1);  zeroStarInRow = [-1 -1];  for i = 1 : numCols  if matrSIZ(j, i) == 1  zeroStarInRow(1) = j;  zeroStarInRow(2) = i;  break  end  end  end  function [matrStreak, matrSIZ] = createL(numRows, numCols, streakPnt, matrStreak, matrSIZ)  i = streakPnt(1);  j = streakPnt(2);  while i > 0 && j > 0 && i <= numRows && j <= numCols  matrStreak(i, j) = 0;  matrSIZ(i, j) = 1;  fprintf("[%d, %d] ", i, j);  kRow = 1;  while kRow <= numRows && (matrSIZ(kRow, j) ~= 1 || kRow == i)  kRow = kRow + 1;  end  if (kRow <= numRows)  lCol = 1;  while lCol <= numCols && (matrStreak(kRow, lCol) ~= 1 || lCol == j)  lCol = lCol + 1;  end  if lCol <= numCols  matrSIZ(kRow,j) = 0;  fprintf("-> [%d, %d] -> ", kRow, j);  end  j = lCol;  end  i = kRow;  end  end  function [fOpt] = getFOpt(matr, matrSIZ)  fOpt = 0;  [numRows,numCols] = size(matr);  for i = 1 : numCols  for j = 1 : numRows  if matrSIZ(j, i) == 1  fOpt = fOpt + matr(j, i);  end  end  end  end |

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

Задача минимизации

|  |
| --- |
| 3 вариант. Матрица:  1 4 7 9 4  9 3 8 7 4  3 4 6 8 2  8 2 4 6 7  7 6 9 8 5  Результат вычитания наименьшего элемента по столбцам:  0 2 3 3 2  8 1 4 1 2  2 2 2 2 0  7 0 0 0 5  6 4 5 2 3  Результат вычитания наименьшего элемента по строкам:  0 2 3 3 2  7 0 3 0 1  2 2 2 2 0  7 0 0 0 5  4 2 3 0 1  Начальная СНН:  0\* 2 3 3 2  7 0\* 3 0 1  2 2 2 2 0\*  7 0 0\* 0 5  4 2 3 0\* 1  Число нулей в построенной СНН: k = 5  Конечная СНН:  0\* 2 3 3 2  7 0\* 3 0 1  2 2 2 2 0\*  7 0 0\* 0 5  4 2 3 0\* 1  X =  1 0 0 0 0  0 1 0 0 0  0 0 0 0 1  0 0 1 0 0  0 0 0 1 0  Результат = 18 |

Задача максимизации

|  |
| --- |
| 3 вариант. Матрица:  1 4 7 9 4  9 3 8 7 4  3 4 6 8 2  8 2 4 6 7  7 6 9 8 5  Матрица после приведения к задаче минимизации:  8 5 2 0 5  0 6 1 2 5  6 5 3 1 7  1 7 5 3 2  2 3 0 1 4  Результат вычитания наименьшего элемента по столбцам:  8 2 2 0 3  0 3 1 2 3  6 2 3 1 5  1 4 5 3 0  2 0 0 1 2  Результат вычитания наименьшего элемента по строкам:  8 2 2 0 3  0 3 1 2 3  5 1 2 0 4  1 4 5 3 0  2 0 0 1 2  Начальная СНН:  8 2 2 0\* 3  0\* 3 1 2 3  5 1 2 0 4  1 4 5 3 0\*  2 0\* 0 1 2  Число нулей в построенной СНН: k = 4  Результат выделения столбцов, в которых стоит 0\*:  8 2 2 0\* 3  0\* 3 1 2 3  5 1 2 0 4  1 4 5 3 0\*  2 0\* 0 1 2  + + + +  Поиск 0 среди невыделенных элементов  Матрица с найденным 0"  8 2 2 0\* 3  0\* 3 1 2 3  5 1 2 0 4  1 4 5 3 0\*  2 0\* 0' 1 2  + + + +  Т.к. в одной строке с 0" есть 0\*, было переброшено выделение:  8 2 2 0\* 3  0\* 3 1 2 3  5 1 2 0 4  1 4 5 3 0\*  + 2 0\* 0' 1 2  + + +  Поиск 0 среди невыделенных элементов  Т.к. среди невыделенных элементов нет нулей, матрица была преобразована:  8 1 1 0\* 3  0\* 2 0 2 3  5 0 1 0 4  1 3 4 3 0\*  + 3 0\* 0' 2 3  + + +  Матрица с найденным 0"  8 1 1 0\* 3  0\* 2 0 2 3  5 0' 1 0 4  1 3 4 3 0\*  + 3 0\* 0' 2 3  + + +  L-цепочка:  [3, 2] -> [5, 2] -> [5, 3]  Текущая СНН:  8 1 1 0\* 3  0\* 2 0 2 3  5 0\* 1 0 4  1 3 4 3 0\*  3 0 0\* 2 3  Итого, k = 5  ----------------------------------------------  Конечная СНН:  8 1 1 0\* 3  0\* 2 0 2 3  5 0\* 1 0 4  1 3 4 3 0\*  3 0 0\* 2 3  X =  0 0 0 1 0  1 0 0 0 0  0 1 0 0 0  0 0 0 0 1  0 0 1 0 0  Результат = 38 |