**3 ЛАБОРАТОРНАЯ РАБОТА №3**

**«Метод решающих матриц»**

**3.1 Цель работы**

Углубление теоретических знаний в области системного анализа, исследование способов оценки сложных систем.

**3.2 Вариант задания – 16 (1)**

Требуется оценить влияние факторов нижнего уровня на проектирование всей системы в целом. Связи между уровнями указаны для каждого варианта отдельно. Веса первого уровня для всех вариантов едины:

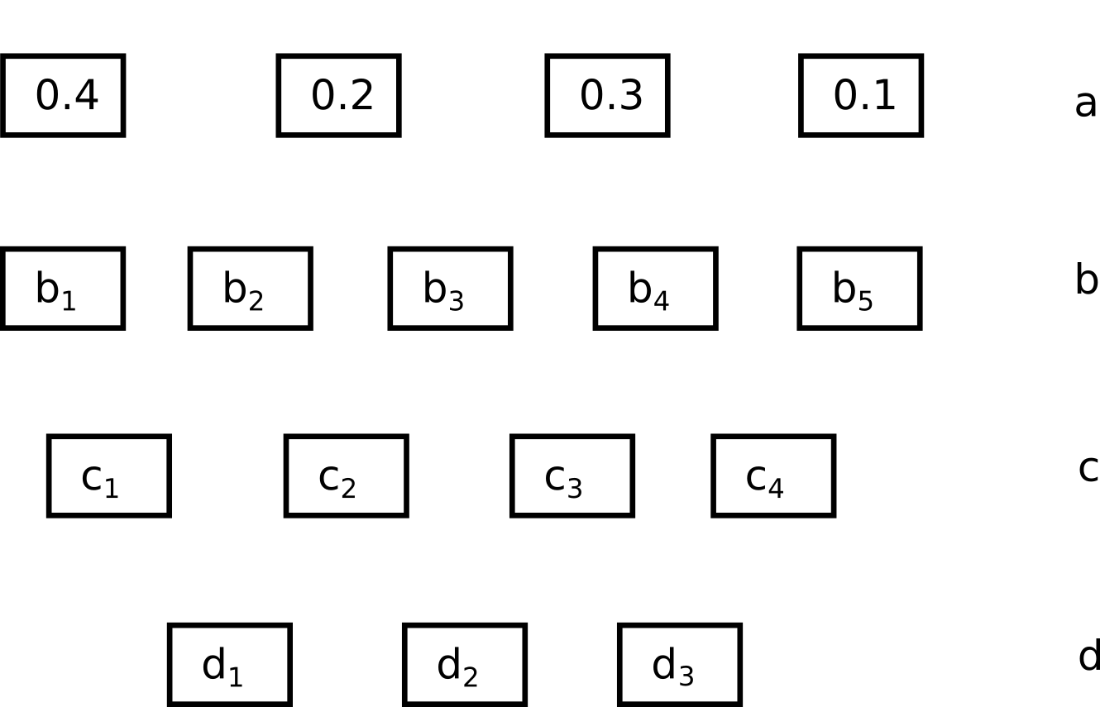


Рисунок 3.1 – Иерархия декомпозиции проблемы

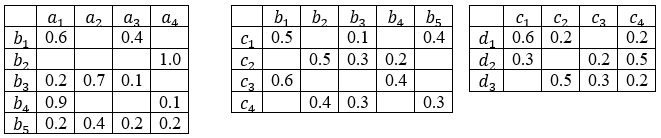


Рисунок 3.2 – Задание по варианту

**3.0.3 Ход работы**

1.3.1 Для начала была составлена схема с исходной постановкой задачи и представлена на рисунке (рис.3.3).

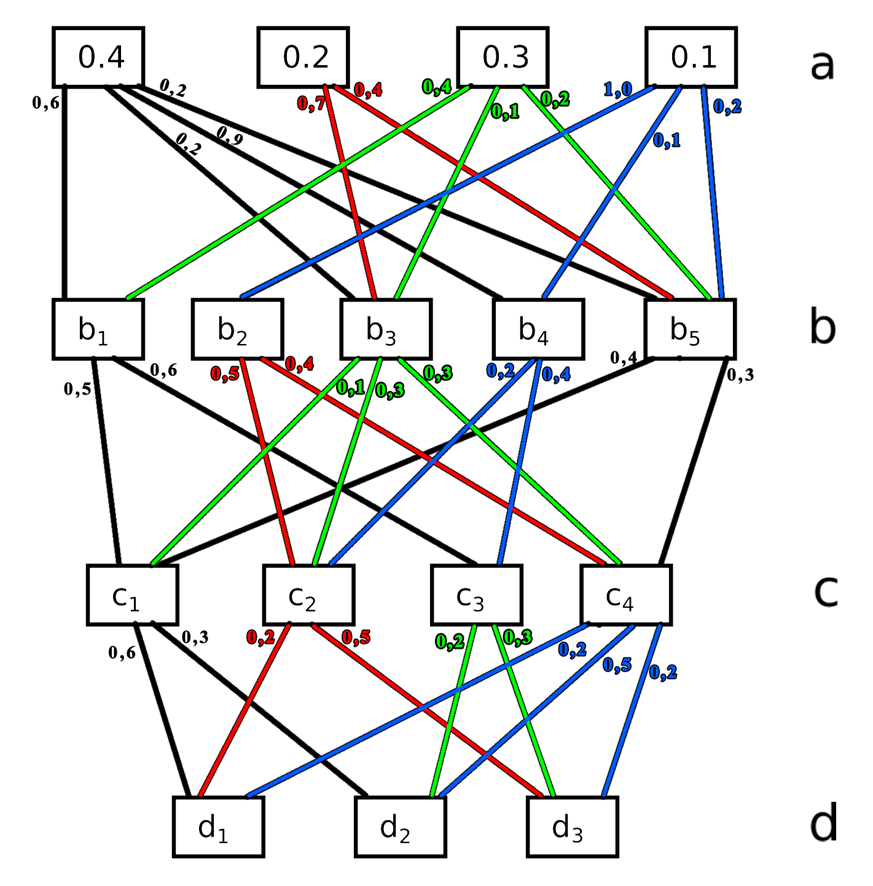


Рисунок 3.3 – Исходные данные

С помощью метода решающих матриц в ручную были проведены исследования и был оценён вклад самого нижнего уровня на проектирование системы. Все вычисления представлены на рисунках 3.4 и 3.5.

Так же была создана программа на языке С++, вычисляющая вклады самых нижних уровней системы:

#include <iostream>

#include <windows.h>

using namespace std;

void gotoxy(int x, int y) {

COORD coord;

coord.X = x; coord.Y = y;

SetConsoleCursorPosition(GetStdHandle(STD\_OUTPUT\_HANDLE), coord);

}

// Класс - матрица из числел double

class Matrix {

double \*\*matrix;

int rows, cols;

char ch1, ch2;

public:

Matrix();

Matrix(int \_rows, int \_cols, char \_ch1, char \_ch2);

~Matrix();

void showMatrix(int level);

float getElem(int i, int j) {

float elem = matrix[i][j];

return elem;

}

};

// Конструктор по умолчанию класса Matrix

Matrix :: Matrix() {

cout << "Первая буква:";

cin >> ch1;

cout << "Вторая буква:";

cin >> ch2;

cout << "Введите количество строк матрицы:";

cin >> rows;

cout << "Введите количество столбцов матрицы:";

cin >> cols;

// выделяем память

matrix = new double\* [rows];

for (int i = 0; i < rows; i++) {

matrix[i] = new double [cols];

}

// заполняем массив

for (int j = 0; j < cols; j++) { //первая буква

system("cls");

for (int i = 0; i < rows; i++) { //вторая буква

cout << "Введите " << ch1 << j+1 << ":" << ch2 << i+1 << " >> ";

cin >> matrix[i][j];

}

}

}

// Конструтор с параметрами (кол-во строк, столбцов матрицы) класса Matrix

Matrix :: Matrix(int \_rows, int \_cols, char \_ch1, char \_ch2) {

ch1 = \_ch1;

ch2 = \_ch2;

rows = \_rows;

cols = \_cols;

// выделяем память

matrix = new double\* [rows];

for (int i = 0; i < rows; i++) {

matrix[i] = new double [cols];

}

// заполняем массив

for (int j = 0; j < cols; j++) { //первая буква

system("cls");

for (int i = 0; i < rows; i++) { //вторая буква

cout << "Введите " << ch1 << j+1 << ":" << ch2 << i+1 << " >> ";

cin >> matrix[i][j];

}

}

}

// Деструктор класса Matrix

Matrix :: ~Matrix() {

// очистить память выделенную матрице

for (int i = 0; i < rows; i++) {

delete [] matrix[i];

}

delete [] matrix;

}

// Показать матрицу класса Matrix

void Matrix :: showMatrix(int level) {

level = level\*10;

gotoxy(6, 1+level);

for (int i = 0; i < cols; i++) {

gotoxy(7+i\*7, 1+level);

cout << ch1 << i+1;

}

for (int i = 0; i < rows; i++) {

gotoxy(2, i+3+level);

cout << ch2 << i+1;

}

for (int i = 0; i < rows; i++) {

gotoxy(7,i+3+level);

for (int j = 0; j < cols; j++) {

gotoxy(7+j\*7, i+3+level);

if (matrix[i][j] != 0)

cout << matrix[i][j];

}

cout << endl;

}

}

//========================================MAIN=======================================

int main() {

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

system("color B0");

int sizeA = 4;

float arrA[sizeA] = {}; // Элементы I уровня

int sizeB = 5;

float arrB[sizeB] = {}; // Относительные веса II уровня

float arrBn[sizeB] = {}; // Нормирование элементов b

int sizeC = 4;

float arrC[sizeC] = {}; // Относительные веса III уровня

float arrCn[sizeC] = {}; // Нормирование элементов c

int sizeD = 3;

float arrD[sizeD] = {}; // Относительные веса VI уровня

float arrDn[sizeD] = {}; // Нормирование элементов d

float sumB = 0, sumC = 0, sumD = 0;

cout << "Введите первый уровень заданный в виде вектора: " << endl;

for (int i = 0; i < sizeA; i++) {

cout << "a" << i+1 << ": "; cin >> arrA[i];

}

cout << endl; system("pause");

// Создание матриц (ввод)

Matrix \*matrixAB = new Matrix(sizeB, sizeA, 'a', 'b');

Matrix \*matrixBC = new Matrix(sizeC, sizeB, 'b', 'c');

Matrix \*matrixCD = new Matrix(sizeD, sizeC, 'c', 'd');

system("cls");

//==========================================================================

// Вычисляем относительные веса II уровня

for (int i = 0; i < sizeB; i++) {

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

arrB[i] = arrB[i] + (matrixAB->getElem(i,j) \* arrA[j]);

}

// Сумма относительных весов II уровня

for (int i = 0; i < sizeB; i++) {

sumB = sumB + arrB[i];

}

// Нормирование элементов II уровня

for (int i = 0; i < sizeB; i++) {

arrBn[i] = arrB[i] / sumB;

}

//==========================================================================

// Вычисляем относительные веса III уровня

for (int i = 0; i < sizeC; i++) {

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

arrC[i] = arrC[i] + (matrixBC->getElem(i,j) \* arrBn[j]);

}

// Сумма относительных весов III уровня

for (int i = 0; i < sizeC; i++) {

sumC = sumC + arrC[i];

}

// Нормирование элементов III уровня

for (int i = 0; i < sizeC; i++) {

arrCn[i] = arrC[i] / sumC;

}

//==========================================================================

// Вычисляем относительные веса VI уровня

for (int i = 0; i < sizeD; i++) {

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

arrD[i] = arrD[i] + (matrixCD->getElem(i,j) \* arrCn[j]);

}

// Сумма относительных весов III уровня

for (int i = 0; i < sizeD; i++) {

sumD = sumD + arrD[i];

}

// Нормирование элементов III уровня

for (int i = 0; i < sizeD; i++) {

arrDn[i] = arrD[i] / sumD;

}

//==========================================================================

int num;

while(1) {

system("pause");

system("cls");

cout << " 1 - Вывести элементы I уровня (исходные данные)" << endl << endl;

cout << " 2 - Вывести относительные веса II (b) уровня" << endl;

cout << " 3 - Вывести сумму относительных весов II (b) уровня" << endl;

cout << " 4 - Вывести нормированные элементы II (b) уровня" << endl << endl;

cout << " 5 - Вывести относительные веса III (c) уровня" << endl;

cout << " 6 - Вывести сумму относительных весов III (c) уровня" << endl;

cout << " 7 - Вывести нормированные элементы III (c) уровня" << endl << endl;

cout << " 8 - Вывести относительные веса VI (d) уровня" << endl;

cout << " 9 - Вывести сумму относительных весов VI (d) уровня" << endl;

cout << "10 - Вывести нормированные элементы VI (d) уровня" << endl << endl;

cout << "11 - Вывести введённые матрицы (исходные данные)" << endl << endl;

cout << "12 - ВЫХОД" << endl << endl;

cout << " >> "; cin >> num;

switch (num) {

case 1: {

break;

}

case 2: {

cout << "b: " << endl << endl;

for (int i = 0; i < sizeB; i++) {

cout << "b" << i+1 << " = " << arrB[i] << endl;

}

break;

}

case 3: {

cout << "Сумма относительных весов II (b) уровня = " << sumB << endl;

break;

}

case 4: {

cout << "bn: " << endl << endl;

for (int i = 0; i < sizeB; i++) {

cout << "bn" << i+1 << " = " << arrBn[i] << endl;

}

break;

}

case 5: {

cout << "c: " << endl << endl;

for (int i = 0; i < sizeC; i++) {

cout << "c" << i+1 << " = " << arrC[i] << endl;

}

break;

}

case 6: {

cout << "Сумма относительных весов III (c) уровня = " << sumC << endl;

break;

}

case 7: {

cout << "cn: " << endl << endl;

for (int i = 0; i < sizeC; i++) {

cout << "cn" << i+1 << " = " << arrCn[i] << endl;

}

break;

}

case 8: {

cout << "d: " << endl << endl;

for (int i = 0; i < sizeD; i++) {

cout << "d" << i+1 << " = " << arrD[i] << endl;

}

break;

}

case 9: {

cout << "Сумма относительных весов VI (d) уровня = " << sumD << endl;

break;

}

case 10: {

cout << "dn: " << endl << endl;

for (int i = 0; i < sizeD; i++) {

cout << "dn" << i+1 << " = " << arrDn[i] << endl;

}

break;

}

case 11: {

system("cls");

matrixAB->showMatrix(0);

matrixBC->showMatrix(1);

matrixCD->showMatrix(2);

break;

}

case 12: {

return 0;

}

}

}

}

Результат выполнения программы соответствует ожиданиям и отображен на рисунке 3.6.

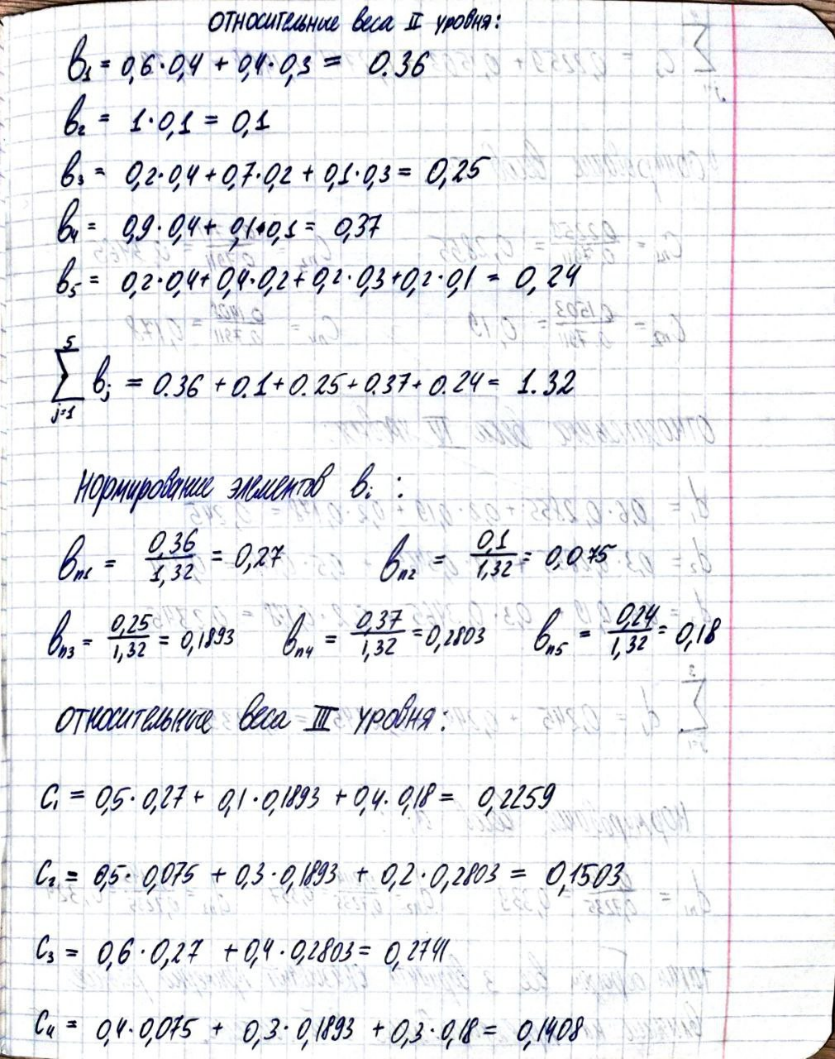


Рисунок 3.4 – Первая часть ручных вычислений

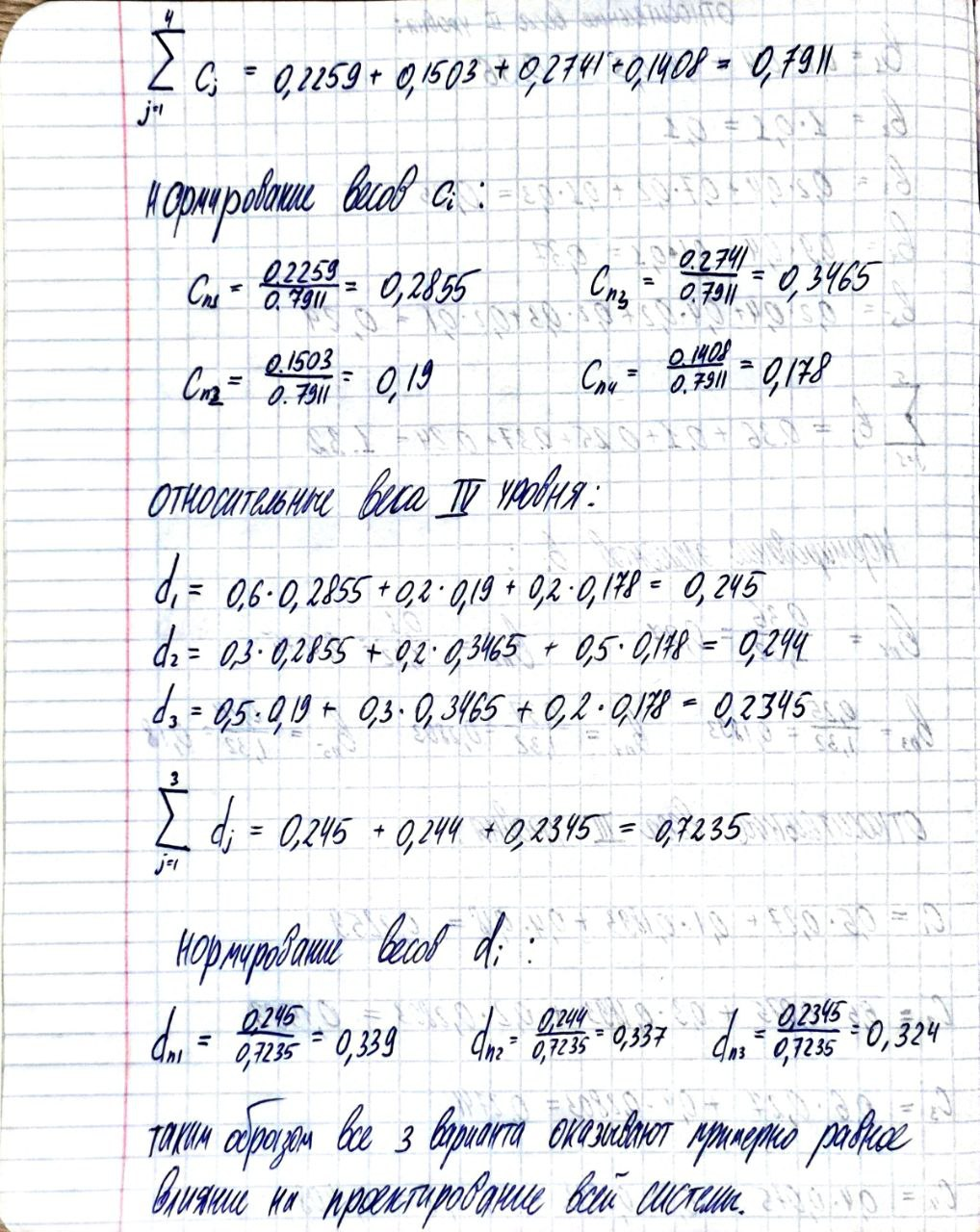


Рисунок 3.5 – Вторая часть ручных вычислений

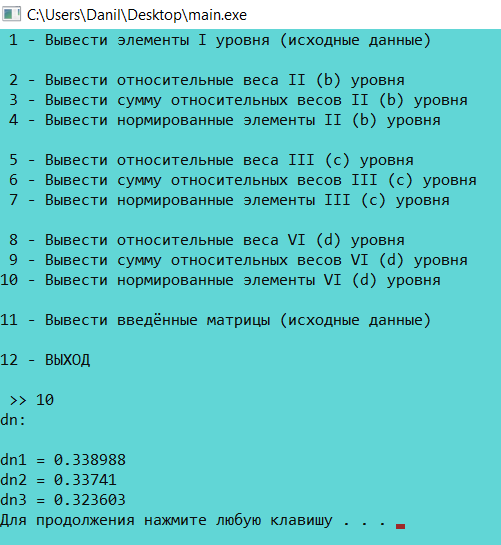


Рисунок 3.6 – Результат выполнения программы

Таким образом все 3 варианта оказывают примерно одинаковое влияние на проектирование всей системы, так как нормированные элементы 4 уровня примерно равны 0.33.

**Выводы**

В ходе выполнения данной лабораторной работы были углублены теоретические знания в области системного анализа, исследованы способы оценки сложных систем. Закреплены навыки разработки программ на языке С++. Полученные во время выполнения лабораторной работы навыки помогут в дальнейшей жизни при необходимости провести системный анализ в какой либо области.