**4 ЛАБОРАТОРНАЯ РАБОТА №4**

**«МЕТОД АНАЛИЗА ИЕРАРХИЙ»**

**4.1 Цель работы**

Углубление теоретических знаний в области системного анализа, приобретение навыков создания и описания иерархических структур, а также изучение понятий цель, критерий, альтернатива.

**4.2 Вариант задания – 16 (4)**

Требуется решить задачу методом анализа иерархий вручную, а затем написать программу, которая решает МАИ задачу любой размерности.

Фирме нужно купить оборудование для очистки воды помещений. Нужно выбрать из возможных альтернатив (табл. 4.1).

Таблица 4.1 ­– Критерии альтернатив для оборудования для очистки воды

|  |  |  |  |
| --- | --- | --- | --- |
| **Критерии** | **A** | **B** | **C** |
| Стоимость | 100 | 80 | 50 |
| Степень очистки | 98 | 95 | 90 |
| Надежность | 85 | 85 | 70 |
| Производительность | 0.2 | 0.1 | 0.05 |
| Долговечность | 4 | 5 | 5 |

**4.3 Ход работы**

4.3.1 Для начала с помощью метода анализа иерархий в ручную были проведены исследования в ходе которых было выявлено, что вариант “A” получает наивысший вес и следовательно является наиболее оптимальным. Все вычисления представлены на рисунках 4.1-4.6.

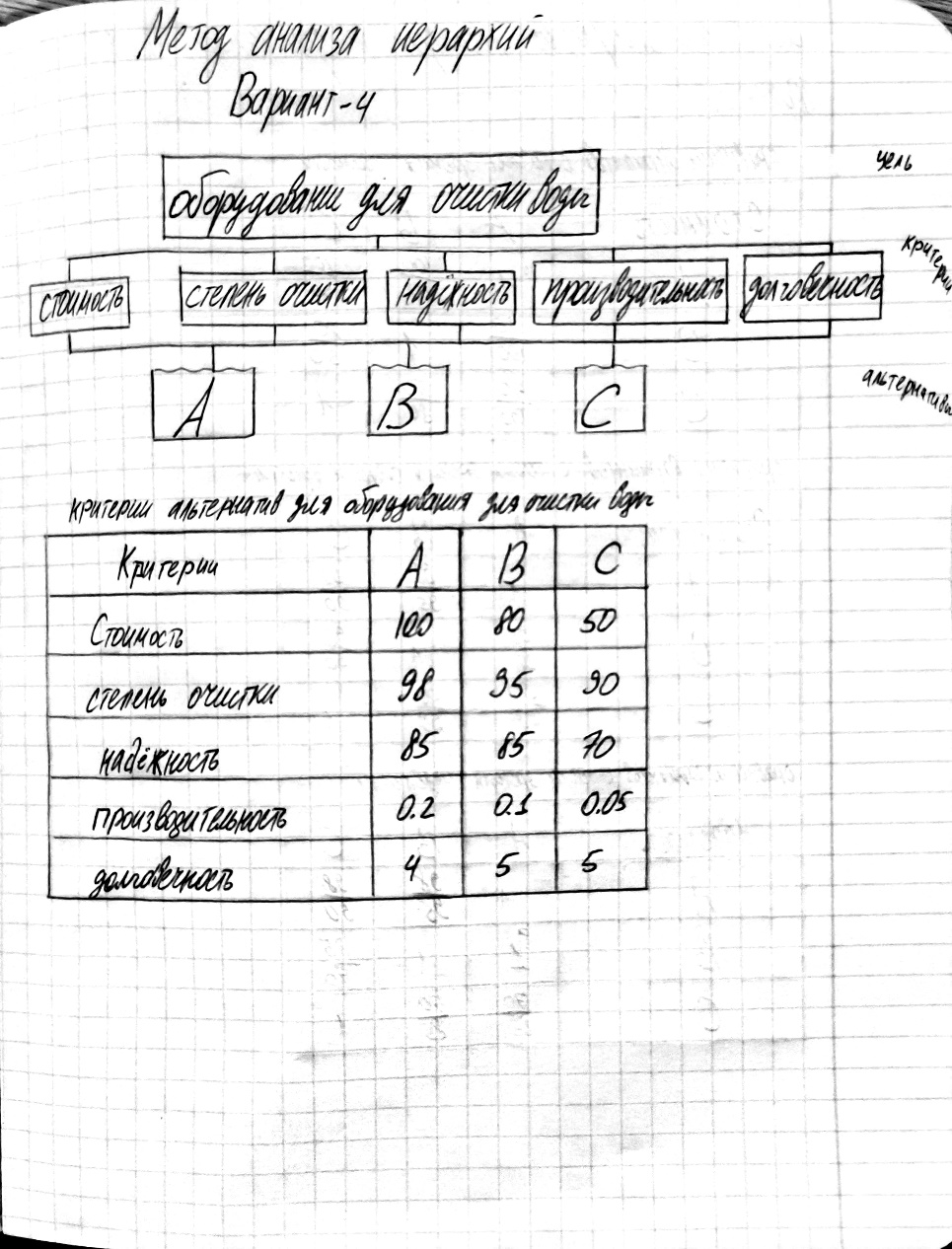


Рисунок 1.1 – Первая часть ручных вычислений

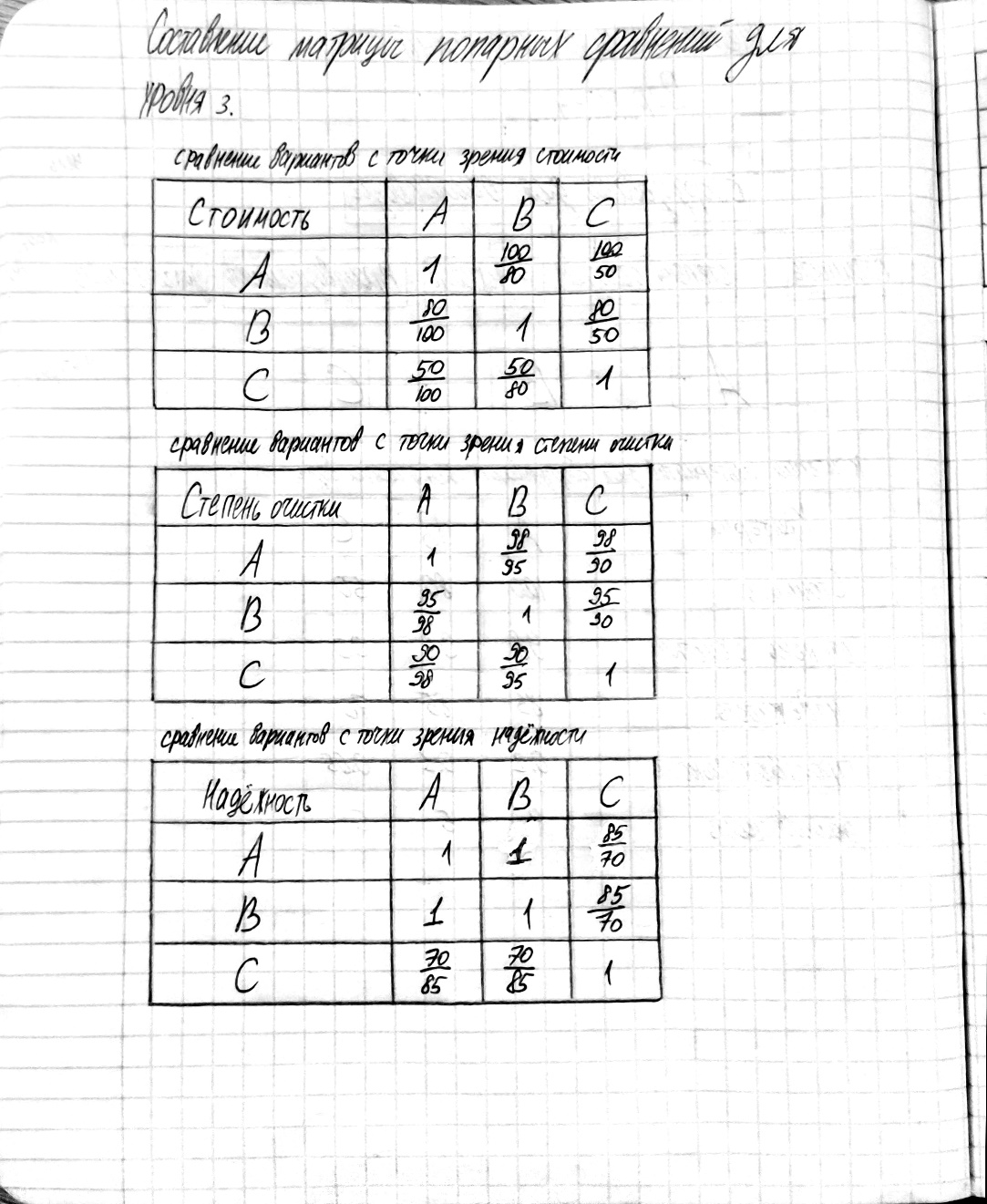


Рисунок 1.2 – Вторая часть ручных вычислений

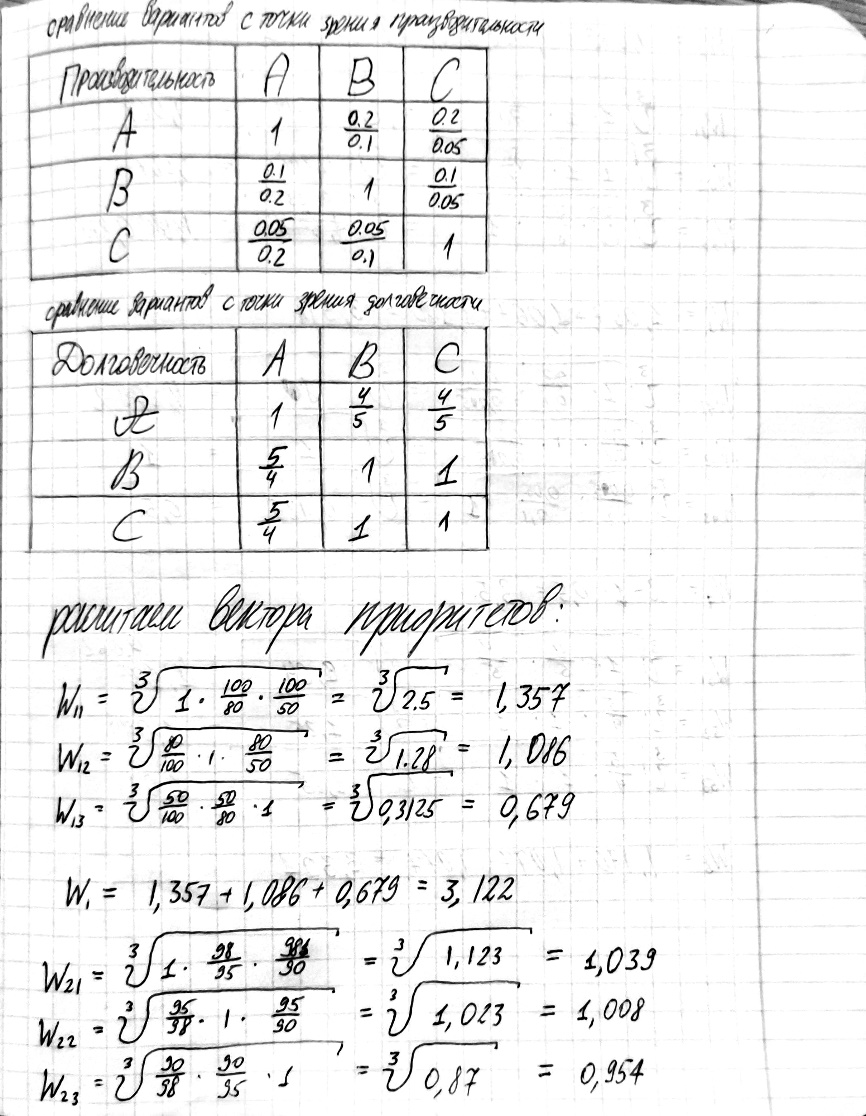


Рисунок 1.3 – Третья часть ручных вычислений

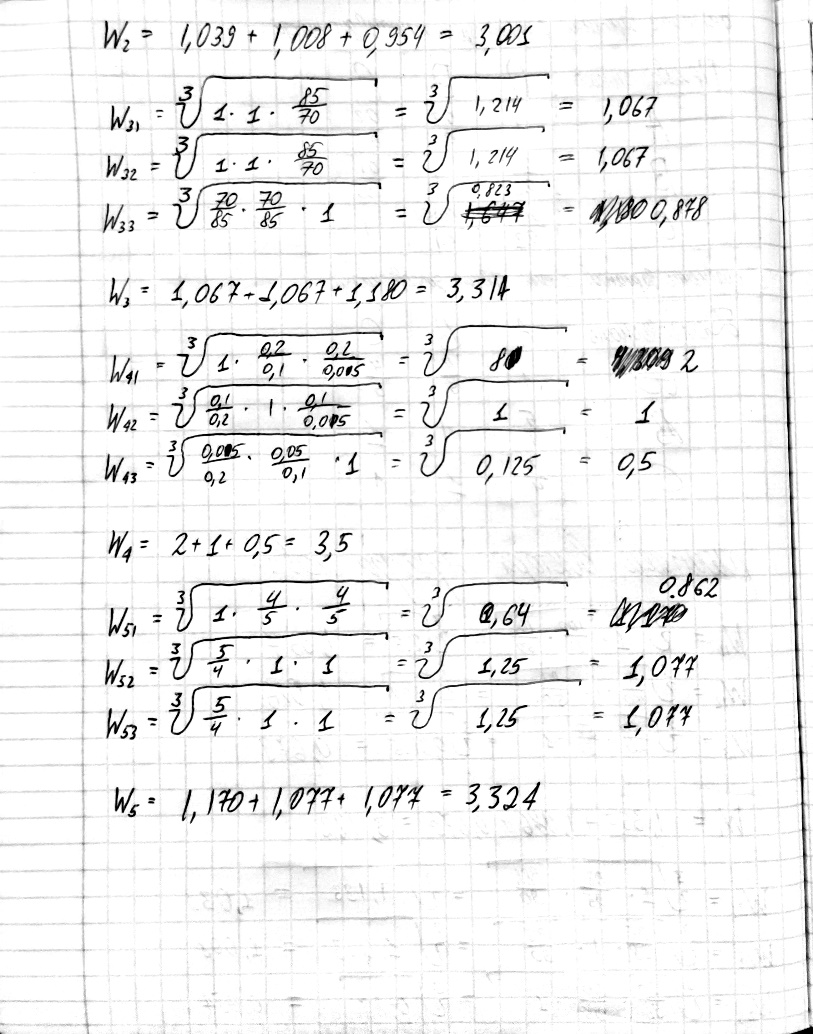


Рисунок 1.4 – Четвертая часть ручных вычислений

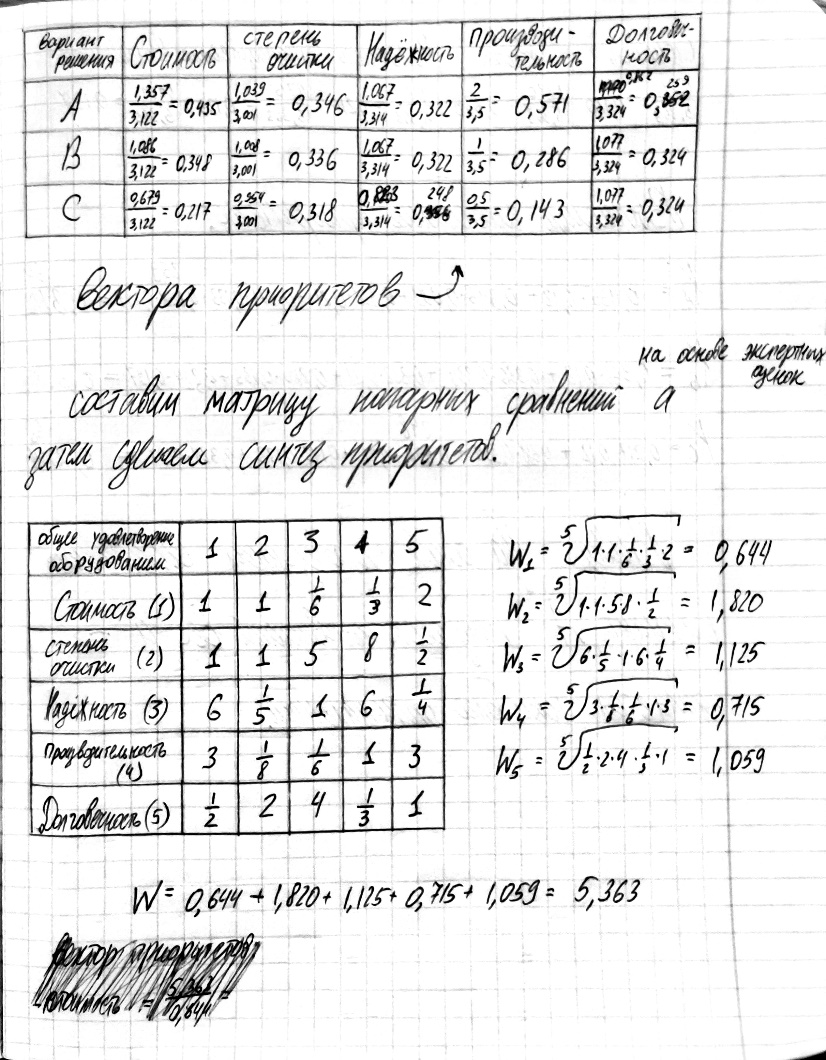


Рисунок 1.5 – Пятая часть ручных вычислений

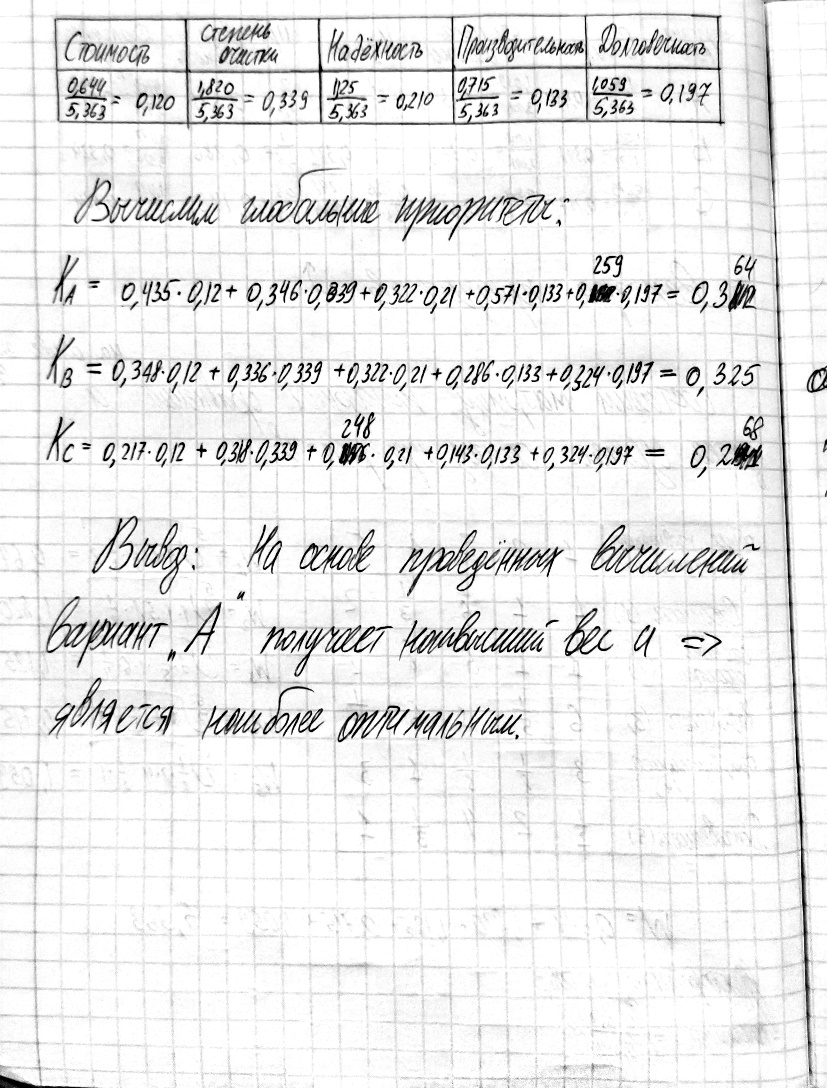


Рисунок 1.6 – Шестая часть ручных вычислений

4.3.2 Так же была создана программа на языке С++, вычисляющая МАИ любой размерности:

#include <iostream>

#include <windows.h>

#include <cmath>

class Matrix {

double \*\*mainMatrix; // критерии альтернатив

double \*\*matrixSravneniy; // матрица попарных сравнений на основе экспертных оценок

// количество критериев и количество альтернатив

int kolKrit, kolAlter; // строки = kolKrit; столбцы = kolAlter;

//можно ввести 15 имен длиной в 19 символов

char nameKrit[15][20]; // имена критериев

char nameAlter[26]; // имена альтернатив

char nameCeli[255]; // имя цели

public:

Matrix();

~Matrix();

void zapolniaem();

void reshaem();

};

Matrix :: Matrix() {

std::cout << "Введите количество критериев (kol <= 15): ";

std::cin >> this->kolKrit;

std::cout << "Введите количество альтернатив (kol <= 25): ";

std::cin >> this->kolAlter;

std::cout << "Вместо пробела используйте символ нижнего подчеркивания \"\_\"" << std::endl;

std::cout << "Введите имя ЦЕЛИ: ";

std::cin >> nameCeli;

for (int i = 1; i <= kolKrit; i++) {

std::cout << "Введите имя " << i << " критерия: ";

std::cin >> nameKrit[i-1];

}

char ch = 65;

for (int i = 1; i <= kolAlter; i++) {

// имя альтернативы:

nameAlter[i-1] = ch;

ch++;

}

std::cout << std::endl;

// выделяем память

mainMatrix = new double\* [kolKrit];

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

mainMatrix[i] = new double [kolAlter];

}

matrixSravneniy = new double\* [kolKrit];

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

matrixSravneniy[i] = new double [kolKrit];

}

system("pause");

system("cls");

}

void Matrix :: zapolniaem(){

std::cout << "ЩА будем вводить сравнительные характеристики" << std::endl;

std::cout << "для этой цели ->\"" << nameCeli << "\"" << std::endl << std::endl;

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

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

std::cout << "Введите [" << nameAlter[j] << "] [" << nameKrit[i] << "] - ";

std::cin >> mainMatrix[i][j];

}

}

std::cout << std::endl;

system("pause");

system("cls");

std::cout << "А теперь будем вводить элементы матрицы попарных" << std::endl;

std::cout << "сравнений на освнове экспертных оценок" << std::endl << std::endl;

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

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

std::cout << "Введите [" << nameKrit[i] << "] [" << j+1 << "] - ";

std::cin >> matrixSravneniy[i][j];

}

}

std::cout << std::endl;

system("pause");

system("cls");

}

void Matrix :: reshaem(){

double arrVectoraPr[kolKrit]; // вектор приоритетов для главной матрицы (меняется)

double arrPrioritetov[kolKrit]; // вектор приоритетов для матриы попарных сравнений

std::cout << "Глобальные приоритеты:" << std::endl << std::endl;

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

//сделаем синтез приоритетов из матрицы попарных сравнений

double tempArr[kolKrit]; //временный массив размера кол-ва критериев

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

tempArr[i] = 1;

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

tempArr[i] = tempArr[i] \* matrixSravneniy[i][j];

}

tempArr[i] = pow(tempArr[i], (1.0/kolKrit));

} //вычислили w1, w2, w3, ...

double sum = 0.0;

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

sum = sum + tempArr[i];

} //вычислили W

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

arrPrioritetov[i] = tempArr[i] / sum;

} //сделали синтез приоритетов

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

double matrVectoraPr[kolKrit][kolAlter];

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

tempArr[i] = 0.0;

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

double tempChislitel = mainMatrix[i][j];

matrVectoraPr[i][j] = 1.0;

for(int j2 = 0; j2<kolAlter; j2++) {

matrVectoraPr[i][j] = matrVectoraPr[i][j] \* (tempChislitel / mainMatrix[i][j2]);

}

matrVectoraPr[i][j] = pow(matrVectoraPr[i][j], 1.0/kolAlter);

// std::cout << "вычисленный w[" << i << "] [" << j << "] = " << matrVectoraPr[i][j] << std::endl; //-------->

tempArr[i] = tempArr[i] + matrVectoraPr[i][j];

}

}

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

for(int kolichestvoAlternativ = 0; kolichestvoAlternativ < kolAlter; kolichestvoAlternativ++) {

sum = 0.0;

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

arrVectoraPr[i] = (matrVectoraPr[i][kolichestvoAlternativ] / tempArr[i]);

}

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

sum = sum + (arrVectoraPr[i] \* arrPrioritetov[i]);

} //вычислили глобальный приоритет

std::cout << "k" << nameAlter[kolichestvoAlternativ] << " = " << sum << std::endl;

}

std::cout << std::endl << std::endl;

system("pause");

}

Matrix :: ~Matrix() {

// очистить память выделенную матрицам

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

delete [] mainMatrix[i];

}

delete [] mainMatrix;

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

delete [] matrixSravneniy[i];

}

delete [] matrixSravneniy;

}

int main() {

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

system("color B0");

//создадим объедок с нашими данными МАИ

Matrix \*MAI = new Matrix();

MAI->zapolniaem();

MAI->reshaem();

delete MAI;

return 0;

}

Результат выполнения программы соответствует ожиданиям и отображен на рисунках 4.7-

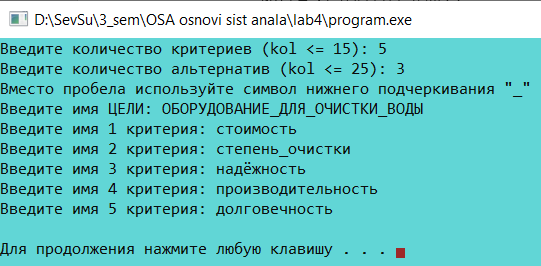


Рисунок 4.7 – Ввод размерности матрицы и имён критериев

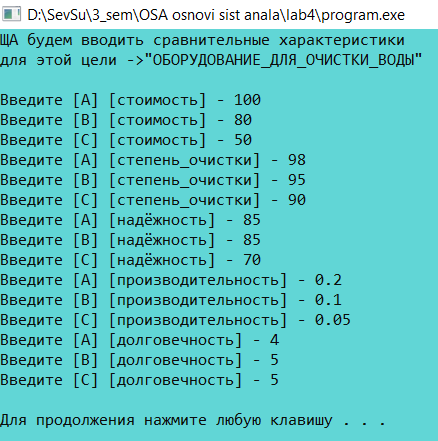


Рисунок 4.8 – Заполнение матрицы с критериями альтернатив

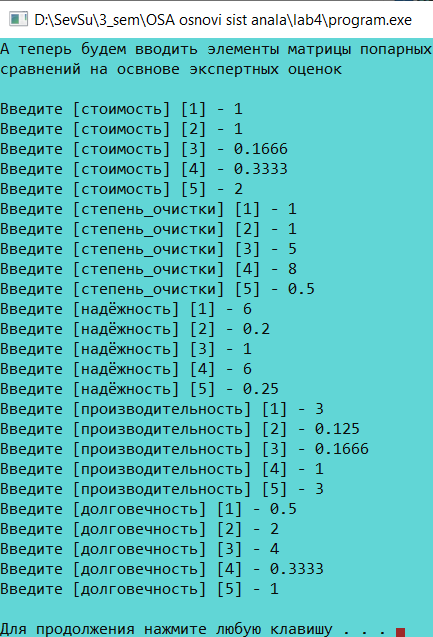


Рисунок 4.9 – Заполнение матрицы попарных сравнений

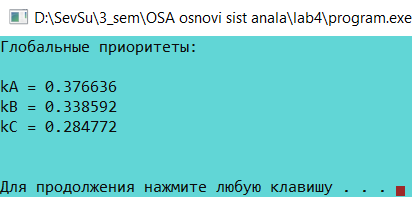


Рисунок 4.10 – Результат выполнения программы

Таким образом все программа подтверждает письменные вычисления и доказывает, что вариант “A” действительно является наиболее оптимальным.

**Выводы**

В ходе выполнения данной лабораторной работы были углублены теоретические знания в области системного анализа, приобретены навыки создания и описания иерархических структур. Изучены такие понятия как цель, критерий и альтернатива. 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.