Лабораторная работа №13.

Многоступенчатые вычислительные процессы.

Цель: средствами языка С научиться работать с многоступенчатыми вычислительными процессами и решить поставленные задачи.
Оборудование: ПК, Visual Studio Code

Задача 1: В систему двух связанных колебательных контуров относительная взаимная проводимость, т.е. отношение тока во втором контуре к величине ЭДС в первом контуре выражается следующей формулой:

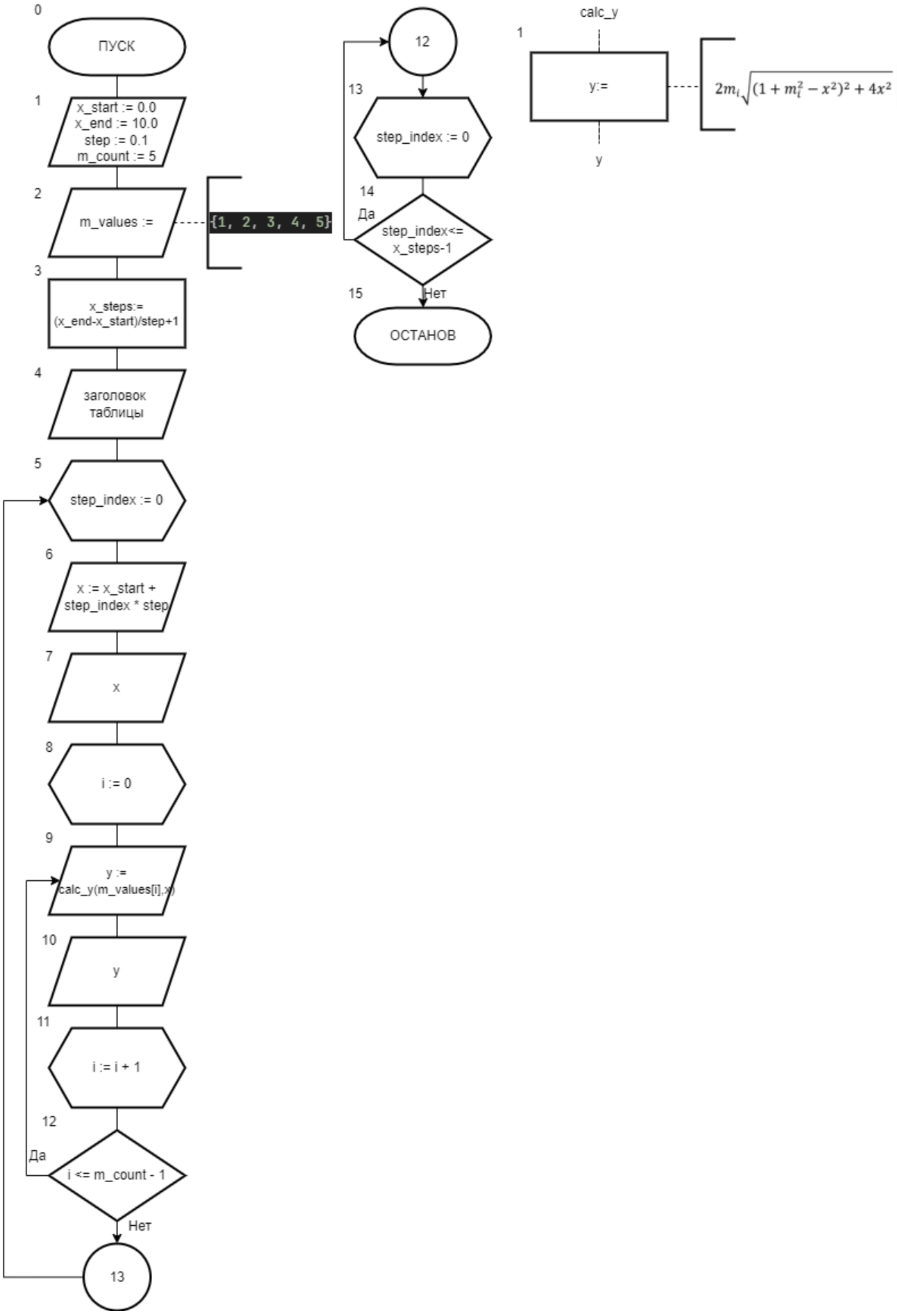
$$y_{\text{отн}} = 2m_i\sqrt{(1 + m_i^2 - x^2)^2 + 4x^2}$$

Требуется рассчитать зависимость уотн от обобщенной расстройки x в интервале от 0 до хотн с шагом Rx при n различных факторах связи mi, i меняет значения от 1 до n. В данном случае переменная mi является элементом массива M = {mi}.Для элементов массива, попавших в заданный диапазон, вычислить y.

Математическая модель:

$$y_{\text{отн}} = 2m_i\sqrt{(1 + m_i^2 - x^2)^2 + 4x^2}$$

Блок схема:


0
ПУСК
1
x_start := 0.0
x_end := 10.0
step := 0.1
m_count := 5
2
m_values :=
{1, 2, 3, 4, 5}
3
x_steps:=
(x_end-x_start)/step+1
4
заголовок
таблицы
5
step_index := 0
6
x := x_start +
step_index * step
7
x
8
i := 0
9
y :=
calc_y(m_values[i],x)
10
y
11
i := i + 1
12
Да
i <= m_count - 1
Нет
13
12
13
step_index := 0
14
Да
step_index<=
x_steps-1
15
Нет
ОСТАНОВ
calc_y
1
y:=
$2m_i\sqrt{(1+m_i^2-x^2)^2+4x^2}$
y

Список идентификаторов:

Имя	Тип	Смысл
X_start	Const	Начальное значение X
X_end	Const	Финальное значение X
Step	Const	Шаг
M_count	const	Количество значений m
M_values	double	Массив значений m
X_steps	int	Количество шагов по x
Step_index	Int	Параметр цикла
x	Double	Значение для вычисления y
i	Int	Параметр цикла
y	Double	Значение функции
Calc_y	function	Функция для вычисления y

Код программы:

```c
#include <stdio.h>
#include <math.h>

#define x_start 0      // Начальное значение x
#define x_end 10       // Конечное значение x
#define step 1         // Шаг
#define m_count 5         // Количество значений m

// Функция для расчёта y
double calc_y(double m, double x) {
    return 2 * m * sqrt(pow(1 + pow(m, 2) - pow(x, 2), 2) + 4 * pow(x, 2));
}

int main() {
    double m_values[m_count] = {1, 2, 3, 4, 5}; // Массив значений m
    int x_steps = ((x_end - x_start) / step) + 1; // Количество шагов по x

    // Заголовок таблицы
    printf("x\t");
    for (int i = 0; i < m_count; i++) {
        printf("y(m=%.1f)\t", m_values[i]);
    }
    printf("\n");

    // Расчёт и вывод значений
    for (int step_index = 0; step_index < x_steps; step_index++) {
        double x = x_start + step_index * step;
        printf("%.2f\t", x);

        for (int i = 0; i ⩽ m_count-1; i++) {
            double y = calc_y(m_values[i], x);
            printf("%.6f\t", y);
        }

        printf("\n");
    }
}
```

Результат выполнения программы:

x	y(m=1.0)	y(m=2.0)	y(m=3.0)	y(m=4.0)	y(m=5.0)
0.00	4.000000	20.000000	60.000000	136.000000	260.000000
1.00	4.472136	17.888544	55.317267	128.996124	250.798724
2.00	8.944272	16.492423	43.266615	108.811764	223.606798
3.00	18.439089	28.844410	36.496575	80.000000	180.277564
4.00	32.249031	54.405882	60.000000	64.498062	128.062485
5.00	50.159745	89.442719	108.166538	102.449988	100.498756
6.00	72.111026	132.966161	171.813853	179.777640	156.204994
7.00	98.081599	184.694342	248.620192	279.427987	269.258240
8.00	128.062485	244.524027	337.923068	397.190131	412.310563
9.00	162.049375	312.409987	439.476962	531.864644	578.705452
10.00	200.039996	388.329757	553.172667	683.005124	766.550716

Анализ вычислений:

Код выполняет вычисления значений y для различных значений x и фиксированных параметров m. Значения x меняются в заданном диапазоне с фиксированным шагом, а для каждого значения m рассчитывается соответствующее значение y. Итоговые данные представлены в виде таблицы, что удобно для анализа зависимости.

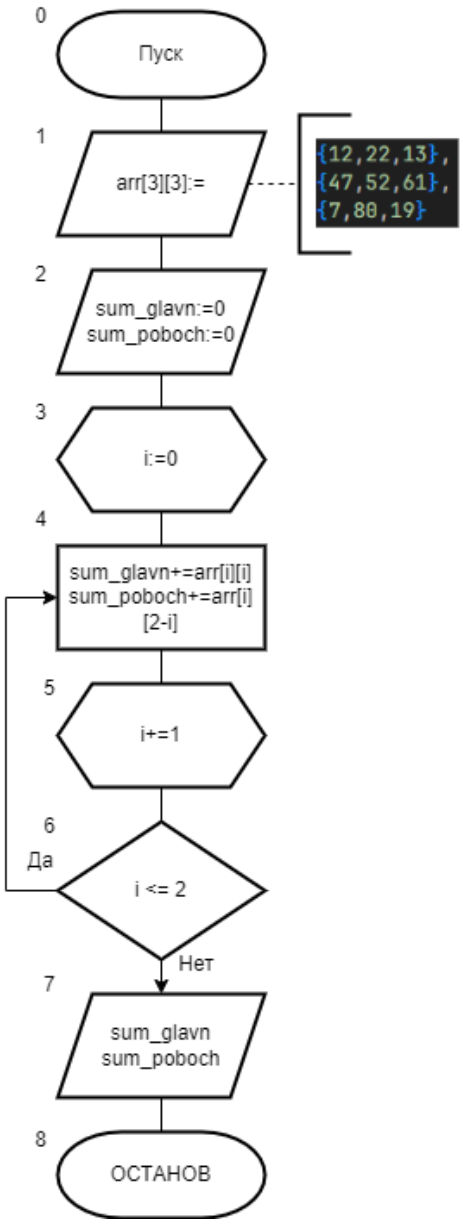
Задача 2: Дан массив 3x3. Найти сумму элементов на главной диагонали и сумму элементов побочной диагонали.

Математическая модель:

$$S_{\text{главн}} = \sum_{i=0}^{2} A[i][i]$$

$$S_{\text{побочн}} = \sum_{i=0}^{2} A[i][2-i]$$

Блок схема:



Список идентификаторов:

Имя	Тип	Смысл
Arr[3][3]	Int	Массив 3 на 3
Sum_glavn	Int	Сумма элементов главной диагонали
Sum_poboch	Int	Сумма элементов побочной диагонали
i	Int	Параметр цикла

Код программы:

```c
#include <stdio.h>
#include <windows.h>

int main(void)
{

SetConsoleOutputCP(CP_UTF8);
int arr[3][3]= {
    {12,22,13},
    {47,52,61},
    {7,80,19}
};

int sum_glavn = 0;
int sum_poboch = 0;

for(int i = 0; i ⩽ 2; i++)
{
        sum_glavn += arr[i][i];
        sum_poboch += arr[i][2-i];
}
printf("Сумма эл. главной диагонали: %d\n",sum_glavn);
printf("Сумма эл. побочной диагонали: %d",sum_poboch);
}
```

Результат выполнения программы:

```
Сумма эл. главной диагонали: 83
Сумма эл. побочной диагонали: 72
```

Анализ вычислений:

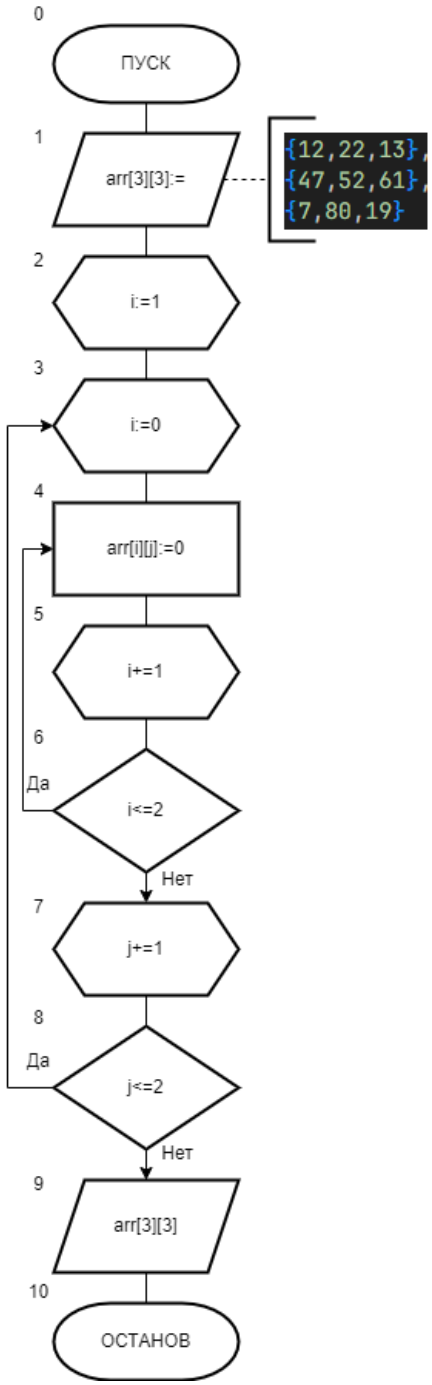
Была написана программа, которая считает сумму элементов главной диагонали и сумму элементов побочной диагонали, после чего выводит ее на экран.

Задача 3: Дан массив 3x3. Заменить элементы, стоящие ниже главной диагонали на 0.

Математическая модель:

$$A = \begin{bmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} = 0 & a_{11} & a_{12} \\ a_{20} = 0 & a_{21} = 0 & a_{22} \end{bmatrix}$$

Блок схема:



Список идентификаторов:

Имя	Тип	Смысл
Arr[3][3]	Int	Массив 3 на 3
i	Int	Параметр цикла

Код программы:

```c
#include <stdio.h>
#include <windows.h>

int main(void)
{

SetConsoleOutputCP(CP_UTF8);
int arr[3][3]= {
    {12,22,13},
    {47,52,61},
    {7,80,19}
};

for (int i = 1; i < 3; i++)
{
    for(int j = 0; j < i; j ++)
    {
        arr[i][j] = 0;
    }
}

for (int i = 0; i ⩽ 2; i++)
{
    for (int j = 0; j ⩽ 2; j++)
    {
        printf("%d\t",arr[i][j]);
    }
    printf("\n");
}

}
```

Результат выполнения программы:

```
12      22      13
0       52      61
0       0       19
```

Анализ вычислений:

Была написана программа, которая заменяет элементы, стоящие ниже главной диагонали на 0.

Вывод:

средствами языка С я научился работать с многоступенчатыми вычислительными процессами и решить поставленные задачи.