|  |  |  |  |
| --- | --- | --- | --- |
|  | | | |
| Федеральное государственное бюджетное  образовательное учреждение высшего образования «Новосибирский государственный технический университет» | | | |
|  | | | |
| Кафедра прикладной математики | | | |
|  | | | |
|  | | | |
| Курсовой проект по курсу | | | |
| **«Численные методы»** | | | |
|  | | | |
|  | Группа | ПМ-12 |
|  |  |
| Студент | Овчинников и. р. |
|  |  |
|
|  |
| Новосибирск | | | |

2023

1. **Формулировка задания**

МКЭ для двумерной краевой задачи для эллиптического уравнения в декартовой системе координат. Базисные функции линейные на треугольниках. Краевые условия всех типов. Коэффициент диффузии λ разложить по линейным базисным функциям. Матрица СЛАУ генерировать в разреженном строчном формате. Для решения СЛАУ использовать МСГ или ЛОС с неполной факторизацией.

1. **Постановка задачи**

Уравнение эллиптической краевой задачи функции u в области Ω с границей S = S1 ∪ S2 ∪ S3:



Краевые условия:

 - краевое условие 1-го рода

 - краевое условие 2-го рода

 - краевое условие 3-го рода

Уравнение в декартовой системе координат:



1. **Теоретическая часть**
2. **Описания разработанных программ**
   1. Структуры данных

|  |  |
| --- | --- |
| **Структура** | **Описание** |
| Vertex | Хранит координаты x и y узла |
| Triangle | Хранит номера узлов конечного элемента и область, к которой он относится |
| FirstBoundaryCondition | Хранит номер узла и номер выражения для вычисления значения |
| SecondBoundaryCondition | Хранит номера узлов ребра и номер выражения для вычисления значения θ |
| ThirdBoundaryCondition | Хранит номера узлов ребра и номера выражений для вычисления значений β и uβ |

* 1. Входные файлы

|  |  |
| --- | --- |
| **Название** | **Описание** |
| Vertices.txt | Первое значение – количество узлов, далее пары координат x и y |
| Triangles.txt | Первое значение – количество конечных элементов, далее для каждого элемента тройка номеров узлов и номер области |
| BoundaryConditions.txt | Первое значение – количество краевых условий, далее для каждого условия тип, ребро/узел и номера выражений для вычисления значений |

* 1. Структура модулей программы

vector<Vertex> vertices; // Узлы системы

vector<Triangle> tris; // Конечные элементы

vector<FirstBoundaryCondition> firstBoundary; // Первые краевые условия

vector<SecondBoundaryCondition> secondBoundary; // Вторые краевые условия

vector<ThirdBoundaryCondition> thirdBoundary; // Третьи краевые условия

double\* q; // Вектор решения

void Input(); // Ввод данных из файла

void Solve(); // Общая функция запуска решения

int regionsNum, globalN; // Количество областей и узлов

int\* ig, \* jg; // Глобальная матрица

double\* ggl, \* ggu, \* di, \* b;

double G[3][3]{}; // Пустая матрица G

double M[3][3]{}; // Пустая матрица M

const double pureM[3][3] = { {2, 1, 1}, {1, 2, 1}, {1, 1, 2} }; // Шаблон матрицы M для возвращения ее в исходное состояние на каждой итерации

double localB[3]{}; // Локальный вектор b

double Lamda(int vert, int region); // Вычисление значения лямбда

double Gamma(int vert, int region); // Вычисление значения гамма

double Function(int vert, int region); // Вычисление значения функции f

double Beta(int vert, int eqNum); // Вычисление значения бета

double Ubeta(int vert, int eqNum); // Вычисление значения U бета для 3 краевого условия

double Theta(int vert, int eqNum); // Вычисление значения тета для 2 краевого условия

double Ug(int vert, int eqNum); // Вычисление значения в узле для 1 краевого условия

double GetAverageLamda(Triangle tri); // Вычисление среднего лямбда на элементе

double GetAverageGamma(Triangle tri); // Вычисление среднего гамма на элементе

double DetD(Triangle tri); // Вычисление определителя D (удвоенной площади) элемента

double Alpha(Triangle tri, int k, int i); // Вычисление значения альфа для построения матрицы G

double EdgeLength(int vert1, int vert2); // Вычисление длины ребра

int IndexOfUnknown(Triangle tri, int i); // Получение глобального номера узла из локального у элемента

void FormM(Triangle tri); // Формирование матрицы G

void FormG(Triangle tri); // Формирование матрицы M

void FormPortrait(); // Формирование портрета глобальной матрицы

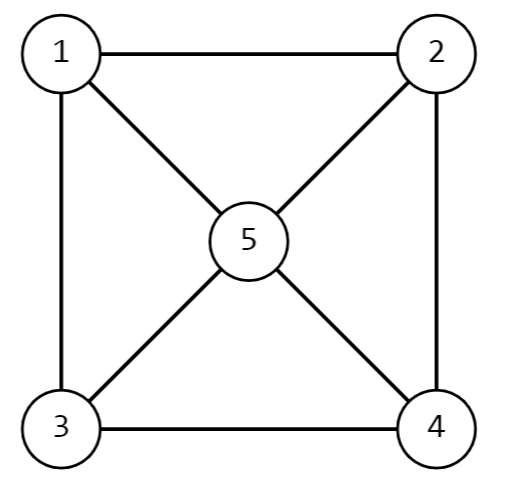
void ResolveBoundaries(); // Учет всех краевых условий

void AddToGlobal(int i, int j, double add); // Добавление значения в глобальную матрицу

void AllocateGlobalMatrix(); // Выделение памяти для глобальной матрицы

void FormB(Triangle tri); // Формирование локального вектора b

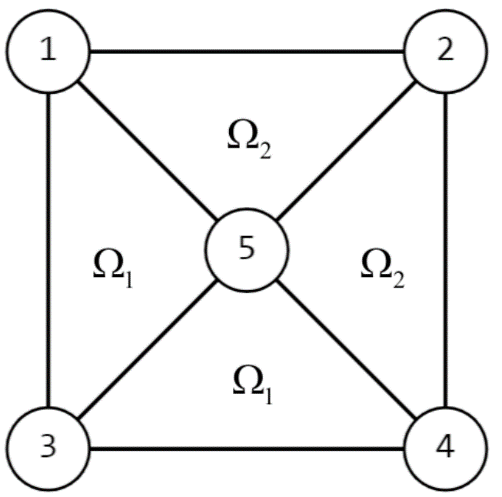
1. **Тесты**

Тест №1

|  |  |  |
| --- | --- | --- |
| Параметры | Координаты вершин: | Краевые условия: |

Решение:

|  |  |  |  |
| --- | --- | --- | --- |
| Узел | q\* | q |  |
| 1 | 6 | 6.00000000000000e+00 | 0.000E+00 |
| 2 | 8 | 8.00000000000000e+00 | 0.000E+00 |
| 3 | 4 | 4.00000000000000e+00 | 0.000E+00 |
| 4 | 6 | 6.00000000000000e+00 | 0.000E+00 |
| 5 | 6 | 6.00000000000000e+00 | 0.000E+00 |

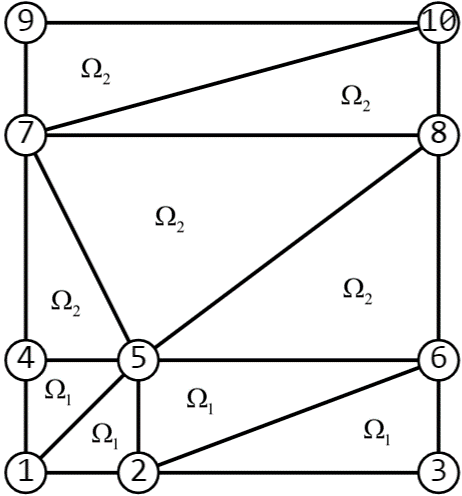
Тест №2

|  |  |  |
| --- | --- | --- |
| Параметры | Координаты вершин: | Краевые условия: |

Решение:

|  |  |  |  |
| --- | --- | --- | --- |
| Узел | q\* | q |  |
| 1 | 6 | 6.00000000000000e+00 | 0.000E+00 |
| 2 | 7 | 7.00000000000000e+00 | 0.000E+00 |
| 3 | 4 | 4.00000000000000e+00 | 0.000E+00 |
| 4 | 6 | 6.00000000000000e+00 | 0.000E+00 |
| 5 | 6 | 6.00000000000000e+00 | 0.000E+00 |

Тест №3



|  |  |  |  |
| --- | --- | --- | --- |
| Параметры | Координаты вершин: | Краевые условия: |  |

Решение:

|  |  |  |  |
| --- | --- | --- | --- |
| Узел | q\* | q |  |
| 1 | 29 | 2.90000000000000e+01 | 0.000E+00 |
| 2 | 29.5 | 2.95000000000000e+01 | 0.000E+00 |
| 3 | 32 | 3.20000000000000e+01 | 0.000E+00 |
| 4 | 35 | 3.50000000000000e+01 | 0.000E+00 |
| 5 | 35.5 | 3.55000000000000e+01 | 0.000E+00 |
| 6 | 38 | 3.80000000000000e+01 | 0.000E+00 |
| 7 | 50 | 5.00000000000000e+01 | 0.000E+00 |
| 8 | 53 | 5.30000000000000e+01 | 0.000E+00 |
| 9 | 59 | 5.90000000000000e+01 | 0.000E+00 |
| 10 | 62 | 6.20000000000000e+01 | 0.000E+00 |