Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

«Пермский национальный исследовательский политехнический университет»

Электротехнический факультет

Кафедра Информационные технологии и автоматизированные системы

Направление: «Программная инженерия»

Профиль: «Разработка информационных систем»

**ЛАБОРАТОРНАЯ РАБОТА № 3**

Дисциплина: «Технологии блокчейн и распределенные информационные системы»

Тема «Решение СЛАУ с помощью OpenMP»

Выполнил:

студент группы

РИС-20-2б

Вязов А.С.

Проверил:

к. т. н., доцент кафедры ИТАС

Щапов В.А.

Пермь 2024

1. **Цель**

Цель данной работы является изучение параллельного программирования с использованием OpenMP. В процессе работы должен быть реализован алгоритм решения системы линейных уравнений с помощью метода Гаусса.

1. **Ход работы**
2. #include <iostream>
3. #include "/opt/homebrew/Cellar/libomp/17.0.6/include/omp.h"
4. using namespace std;
5. const int lengthMatrix = 3; // Размерность системы
6. // Функция для вывода матрицы
7. void printMatrix(double\*\* matrix, double\* answerMatrix) {
8. for (int i = 0; i < lengthMatrix; i++) {
9. for (int j = 0; j < lengthMatrix; j++) {
10. cout << matrix[i][j] << " ";
11. }
12. cout << " = " << answerMatrix[i] << endl;
13. }
14. }
15. void matrixTransformation(double\*\* matrix, double\* answerMatrix) {
16. for (int k = 0; k < lengthMatrix - 1; k++) {
17. #pragma omp parallel for shared(matrix, answerMatrix)//распараллелить цикл, используя общие данные matrix и answerMatrix, которые могут быть доступны для чтения и записи из всех потоков, участвующих в этой параллельной области
18. for (int i = k + 1; i < lengthMatrix; i++) {
19. double temp = matrix[i][k] / matrix[k][k];
20. for (int j = k; j < lengthMatrix; j++) {
21. matrix[i][j] -= temp \* matrix[k][j];
22. }lab
23. answerMatrix[i] -= temp \* answerMatrix[k];
24. }
25. }
26. }
27. void calculationSolution(double\*\* matrix, double\* answerMatrix, double\* solution) {
28. solution[lengthMatrix - 1] = answerMatrix[lengthMatrix - 1] / matrix[lengthMatrix - 1][lengthMatrix - 1];
29. for (int i = lengthMatrix - 2; i >= 0; i--) {
30. double sum = answerMatrix[i];
31. for (int j = i + 1; j < lengthMatrix; j++) {
32. sum -= matrix[i][j] \* solution[j];
33. }
34. solution[i] = sum / matrix[i][i];
35. }
36. }
37. double\*\* createMatrixSecond(int length) {
38. double\*\* matrix = new double\* [length];
39. for (int i = 0; i < length; i++) {
40. matrix[i] = new double[length];
41. for (int j = 0; j < length; j++) {
42. matrix[i][j] = rand() % 10;
43. }
44. }
45. return matrix;
46. }
47. double\* createMatrixFirst(int length) {
48. double\* matrix = new double [length];
49. for (int i = 0; i < length; i++) {
50. matrix[i] = rand() % 10;
51. }
52. return matrix;
53. }
54. int main() {
55. double\*\* matrix = createMatrixSecond(lengthMatrix);
56. double\* answerMatrix = createMatrixFirst(lengthMatrix);
57. double\* solution = new double [3];
58. cout << "Исходная система уравнений:" << endl;
59. printMatrix(matrix, answerMatrix);
60. matrixTransformation(matrix, answerMatrix);
61. cout << "\nМатрица после прямого хода:" << endl;
62. printMatrix(matrix, answerMatrix);
63. calculationSolution(matrix, answerMatrix, solution);
64. cout << "\nРешение СЛАУ:" << endl;
65. for (int i = 0; i < lengthMatrix; i++) {
66. cout << "x" << i + 1 << " = " << solution[i] << endl;
67. }
68. return 0;
69. }

Листинг 1 – Код решения СЛАУ методом Гаусса

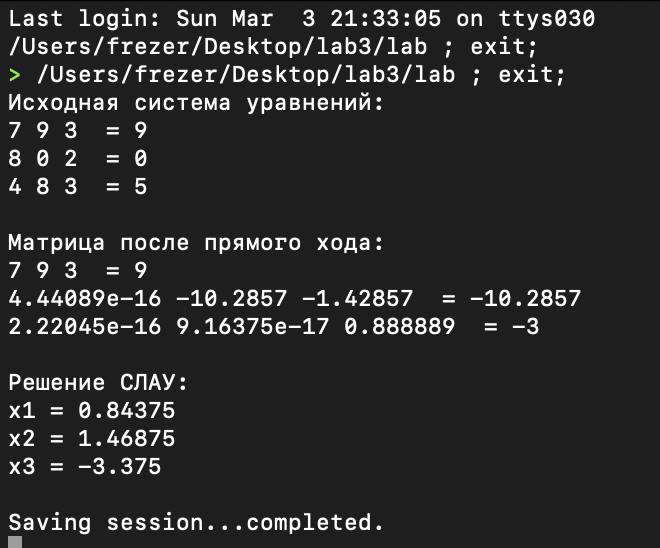


Рисунок 1 – Результат выполнения программы