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

#define N 3       // Размерность системы
#define EPSILON 1e-6 // Точность

void printVector(double vector[N]) {
   for (int i = 0; i < N; i++) {
      printf("x%d = %10.6lf\n", i + 1, vector[i]);
   }
}

void gaussSeidel(double matrix[N][N+1], double x[N]) {
   double x_new[N];
   double error;
   int iteration = 0;

   do {
      for (int i = 0; i < N; i++) {
         x_new[i] = matrix[i][N];
         for (int j = 0; j < N; j++) {
            if (i != j) {
               x_new[i] -= matrix[i][j] * x[j];
            }
         }
         x_new[i] /= matrix[i][i];
      }

      // Вычисление ошибки
      error = 0.0;
      for (int i = 0; i < N; i++) {
         error += fabs(x_new[i] - x[i]);
         x[i] = x_new[i];
      }

      iteration++;
   } while (error > EPSILON);

   printf("Метод Зейделя завершен за %d итераций.\n", iteration);
}

int main() {
   double matrix[N][N+1] = {
      {4, 1, 2, 4},
      {3, 5, 1, 7},
      {1, 1, 3, 3}
   };

   double x[N] = {0, 0, 0};

   printf("Исходная система уравнений:\n");
   for (int i = 0; i < N; i++) {
      for (int j = 0; j <= N; j++) {
         printf("%10.3lf ", matrix[i][j]);
      }
      printf("\n");
   }

   gaussSeidel(matrix, x);

   printf("Решение:\n");
   printVector(x);
```

```
    return 0;
}
```

Отчёт по выполненной работе

Введение

Метод Зейделя (метод последовательных приближений) — это итерационный метод решения систем линейных алгебраических уравнений. В отличие от прямого метода Гаусса, метод Зейделя использует начальные приближения и последовательно улучшает их, пока не достигнет заданной точности.

Описание кода

1. **Заголовочные файлы:**

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

Эти заголовочные файлы предоставляют функции ввода-вывода, динамического выделения памяти и математических операций.

2. **Определение параметров:**

```
#define N 3       // Размерность системы
#define EPSILON 1e-6 // Точность
```

Макрос `N` определяет размерность системы уравнений. Макрос `EPSILON` задает требуемую точность решения.

3. **Функция `printVector`:**

```
void printVector(double vector[N]) {
    for (int i = 0; i < N; i++) {
        printf("x%d = %10.6lf\n", i + 1, vector[i]);
    }
}
```

Эта функция выводит вектор решений на экран.

4. **Функция `gaussSeidel`:**

```
void gaussSeidel(double matrix[N][N+1], double x[N]) {
    double x_new[N];
    double error;
```

```
    int iteration = 0;

    do {
        for (int i = 0; i < N; i++) {
            x_new[i] = matrix[i][N];
            for (int j = 0; j < N; j++) {
                if (i != j) {
                    x_new[i] -= matrix[i][j] * x[j];
                }
            }
            x_new[i] /= matrix[i][i];
        }

        // Вычисление ошибки
        error = 0.0;
        for (int i = 0; i < N; i++) {
            error += fabs(x_new[i] - x[i]);
            x[i] = x_new[i];
        }

        iteration++;
    } while (error > EPSILON);

    printf("Метод Зейделя завершен за %d итераций.\n", iteration);
}
```

Функция `gaussSeidel` выполняет итерационный процесс решения СЛАУ методом Зейделя. Алгоритм заключается в следующем:

- **Начальные приближения:** Вектор `x` инициализируется начальными значениями (в данном случае нулями).
- **Итерационный процесс:** Для каждого уравнения вычисляются новые значения переменных на основе текущих значений.
- **Оценка ошибки:** Вычисляется сумма абсолютных значений разностей новых и старых приближений. Итерации продолжаются, пока ошибка не станет меньше заданного порога `EPSILON`.

5. **Функция `main`:**

```
int main() {
    double matrix[N][N+1] = {
        {4, 1, 2, 4},
        {3, 5, 1, 7},
        {1, 1, 3, 3}
    };

    double x[N] = {0, 0, 0};
```

```
    printf("Исходная система уравнений:\n");
    for (int i = 0; i < N; i++) {
        for (int j = 0; j <= N; j++) {
            printf("%10.3lf ", matrix[i][j]);
        }
        printf("\n");
    }

    gaussSeidel(matrix, x);

    printf("Решение:\n");
    printVector(x);

    return 0;
}
```

Эта функция:
- Определяет исходную матрицу коэффициентов и столбец свободных членов.
- Инициализирует вектор начальных приближений нулями.
- Выводит исходную систему уравнений.
- Запускает итерационный процесс методом Зейделя.
- Выводит найденное решение.

Заключение

Метод Зейделя — это эффективный итерационный метод решения систем линейных уравнений, который часто используется при больших размерностях систем и плохой обусловленности матриц. Данный код демонстрирует применение метода Зейделя на примере системы размерностью 3x3. Алгоритм метода заключается в последовательном улучшении приближений решения, пока не будет достигнута заданная точность.