МИНОБРНАУКИ РОССИИ
федеральное государственное автономное образовательное учреждение высшего образования «Московский государственный технологический университет «СТАНКИН» (ФГАОУ ВО «МГТУ «СТАНКИН»)

Институт информационных технологий

Кафедра информационных технологий и вычислительных систем

ОТЧЕТ О ВЫПОЛНЕНИИ
ЛАБОРАТОРНОЙ РАБОТЫ ПО ДИСЦИПЛИНЕ
«Вычислительная математика»

СТУДЕНТА 2 КУРСА баклавариата ГРУППЫ ИДБ-23-03

Долгополов Александр Витальевич

НА ТЕМУ
«Метод Зейделя»

Направление: 09.03.01 Информатика и вычислительная техника

Профиль подготовки: «Разработка программных комплексов в рамках цифровой трансформации деятельности предприятий»

Отчёт сдан: «__» апреля 2025 г.
Оценка: _____
Преподаватель _____

МОСКВА 2025

Исходный код

```
#define MAX_ITER 1000

// Условие окончания
bool MainWindow::converge(
const std::vector<double>& X,
const std::vector<double>& Xp
) {
    double norm = 0;
    for (int i = 0; i < X.size(); i++)
        norm += (X[i] - Xp[i]) * (X[i] - Xp[i]);
    return (sqrt(norm) < ui->doubleSpinBox_epsilon->value());
}

// Округление
double MainWindow::round(double x) {
    int i = 0;
    double eps = ui->doubleSpinBox_epsilon->value();
    while (eps < 1) {
        i++;
        eps *= 10;
    }
    int n = pow(10, i);
    x = int(x * n + 0.5) / double(n);
    return x;
}


// Поиск решения
void MainWindow::updateResults() {
    using boost::numeric::ublas::matrix;
    QTableWidget* input_table = ui->tableWidget_input;
    QTableWidget* output_table = ui->tableWidget_output;
    QTableWidgetItem* cell;
    QString text;

    bool valid;
    double d;

    size_t row_count = input_table->rowCount();
    matrix<double> A(row_count, row_count);
    std::vector<double> B(row_count);

    // Проверка ввода
    #pragma region Input check
    for (size_t i = 0; i < row_count; i++) {
        for (size_t j = 0; j < row_count; j++) {
            cell = input_table->item(i, j);
            if (!cell || cell->text().isEmpty()) {
                ui->label_Error->setText(
                    tr("Ошибка: ячейка %1:%2 пустая")
                    .arg(i+1).arg(j+1)
                );
                return;
            }
            d = cell->text().toDouble(&valid);
            if (!valid) {
                ui->label_Error->setText(
                    tr("Ошибка: в ячейке %1:%2 указано не число")
                    .arg(i+1).arg(j+1)
                );
                return;
```

```
            }
            A(i, j) = d;
        }
        cell = input_table->item(i, row_count);
        if (!cell || cell->text().isEmpty()) {
            ui->label_Error->setText(
                tr("Ошибка: ячейка %1:%2 пустая")
                .arg(i+1).arg(row_count)
            );
            return;
        }
        d = cell->text().toDouble(&valid);
        if (!valid) {
            ui->label_Error->setText(
                tr("Ошибка: в ячейке %1:%2 указано не число")
                .arg(i+1).arg(row_count)
            );
            return;
        }
        B[i] = d;
    }
    ui->label_Error->setText("");
    #pragma endregion

    // Решение
    #pragma region Solution
    // X - текущая итерация
    // Xp - предыдущая итерация
    std::vector<double> X(B);
    std::vector<double> Xp;
    size_t k = 0;

    // Итерируем, пока не достигнем необходимой точности
    do {
        // Поиск решения
        Xp = X;
        for (size_t i = 0; i < row_count; i++) {
            double s = 0;
            for (size_t j = 0; j < row_count; j++)
                if (i != j) s += A(i, j) * X[j];
            X[i] = (B[i] - s) / A(i, i);
        }
        // Проверка количества итераций
        // (выход из бесконечного цикла, если сходимость не выполняется)
        if (++k > MAX_ITER) {
            ui->label_Error->setText(
                tr("Превышено ограничение по количеству итераций")
            );
            return;
        }
    } while (!converge(X, Xp));
    // Вывод результата
    ui->label_Error->setText(
        tr("Решение найдено за %n итераций", "", k)
    );
    for (size_t i = 0; i < row_count; i++) {
        cell = new QTableWidgetItem(locale().toString(this->round(X[i])));
        output_table->setItem(i, 0, cell);
    }
    #pragma endregion
}
```