Содержание

[Введение 3](#_Toc25572636)

[1. Анализ объекта 4](#_Toc25572637)

[1.1 Описание предметной области. 4](#_Toc25572638)

[1.2 Построение концептуальной модели предметной области. 7](#_Toc25572639)

[2 Постановка задачи 8](#_Toc25572640)

[2.1 Определение требований к программной системе. 8](#_Toc25572641)

[2.2 Описание аналогов системы. 9](#_Toc25572642)

[2.3 Обзор и обоснование выбора средств реализации 10](#_Toc25572643)

[3 Проектирование 11](#_Toc25572644)

[3.1 Разработка архитектуры программного продукта 11](#_Toc25572645)

[3.2 Проектирование структур хранения данных. 12](#_Toc25572646)

[4 Реализация 13](#_Toc25572647)

[4.1 Разработка архитектуры программного продукта 13](#_Toc25572648)

[4.2 Разработка интерфейса программного продукта 14](#_Toc25572649)

[Разработка алгоритмов реализации вариантов использования. 18](#_Toc25572650)

[5. Тестирование 20](#_Toc25572651)

[Литература 23](#_Toc25572652)

[Приложение А 24](#_Toc25572653)

# Введение

Линейное программирование - это раздел математики ориентируемый на нахождении экстремума в задачах, которые описываются линейными уравнениями.

Для упрощения процесса решения исходные данные задачи линейного программирования при решении ее симплекс методом записываются в специальные симплекс-таблицы. Поэтому одна из модификаций симплекс метода получила название табличный симплекс метод.

Мой проект представляет собой приложение для решения линейных оптимизационных задач табличным симплекс методом. Данная курсовая работа выполнена на языке программирования высокого уровня Java с использованием IDE Eclipse.

# Анализ объекта

## Описание предметной области.

Среди универсальных методов решения задач линейного программирования наиболее распространен симплексный метод (метод последовательного улучшения плана). Суть этого метода в том, что вначале получают допустимый вариант, удовлетворяющий всем ограничениям, но необязательно оптимальный (начальное опорное решение); оптимальность достигается последовательным улучшением исходного варианта за определенное число этапов (итераций). При применении этого метода исходная ЗЛП формулируется в канонической форме (1.3).

Симплекс-метод основан на следующих свойствах ЗЛП:

1. Не существует локального экстремума, отличного от глобального. Другими словами: если экстремум есть, то он единственный.
2. Множество всех планов задачи линейного программирования выпукло.
3. Целевая функция ЗЛП достигает своего максимального (минимального) значения в угловой точке многогранника решений (в его вершине). Если целевая функция принимает свое оптимальное значение более чем в одной угловой точке, то она достигает того же значения в любой точке, являющейся выпуклой линейной комбинацией этих точек.
4. Каждой угловой точке многогранника решений отвечает опорный план ЗЛП.

Существует две разновидности симплексного метода:

* симплекс-метод с естественным базисом;
* симплекс-метод с искусственным базисом (М-метод);
* Симплекс-метод с естественным базисом.

Для применения этого метода ЗЛП должна быть сформулирована в канонической форме (1.4), причем матрица системы уравнений должна содержать единичную подматрицу размерностью *m×m* . В этом случае очевиден начальный опорный план (неотрицательное базисное решение).

***Алгоритм симплекс – метода***

I. Если задача не приведена к каноническому виду, то сделать это так, как в описано в пункте №1. Получить расширенную систему вида:



II. Исходную расширенную систему заносим в первую симплексную таблицу. Последняя строка таблицы, в которой приведено уравнение для линейной функции цели, называется оценочной. В ней указываются коэффициенты функции цели с противоположным знаком:

Δ*j =*.

В левом столбце таблицы записываем основные переменные (базис), в первой строке таблицы — все переменные (отмечая при этом основные), во втором столбце – коэффициенты при основных переменных системы *сi*, в третьем столбце – свободные члены расширенной системы *b1, b2, ..., bm*. Последний столбец подготовлен для оценочных отношений, необходимых при расчете наибольшего возможного значения переменной. В рабочую часть таблицы (начиная с четвертого столбца и второй строки) занесены коэффициенты *aij* при переменных из расширенной системы. Далее таблица преобразуется по определенным правилам.

III. Проверяем выполнение критерия оптимальности при решении задачи на максимум — наличие в последней строке отрицательных коэффициентов Δj < 0 (*ci* > 0). Если таких нет, то решение оптимально, достигнут *max F = * (в левом нижнем углу таблицы), основные переменные принимают значения *bi* (третий столбец), неосновные переменные равны 0, т.е. получаем оптимальное базисное решение.

IV. Если критерий оптимальности не выполнен, то наибольший по модулю отрицательный коэффициент Δj < 0 в последней строке определяет разрешающий столбец s.

Составляем оценочные ограничения каждой строки по следующим правилам:

1. ∞, если *bi* и *ais* имеют разные знаки;
2. ∞, если *bi* = 0 и *ais* < 0;
3. ∞, если *ais* = 0;
4. 0, если *bi* = 0 и *ais* > 0;
5. , если  *ai*0 и *ais* имеют одинаковые знаки.

Определяем . Если конечного минимума нет, то задача не имеет конечного оптимума (*Fmax = ∞*). Если минимум конечен, то выбираем строку *q*, на которой он достигается (любую, если их несколько), и называем ее разрешающей строкой. На пересечении разрешающих строки и столбца находится разрешающий элемент *aqs*.

V. Переходим к следующей таблице по правилам:

а) в левом столбце записываем новый базис: вместо основной переменной *xq* - переменную *xs*,

б) в столбцах, соответствующих основным переменным, проставляем нули и единицы: 1 — против "своей" основной переменной, 0 — против "чужой" основной переменной, 0 — в последней строке для всех основных переменных;

в) новую строку с номером *q* получаем из старой делением на разрешающий элемент *aqs*;

г) все остальные элементы *a′qs* вычисляем по правилу прямоугольника:



Далее переходимк п.III алгоритма.

## Построение концептуальной модели предметной области.

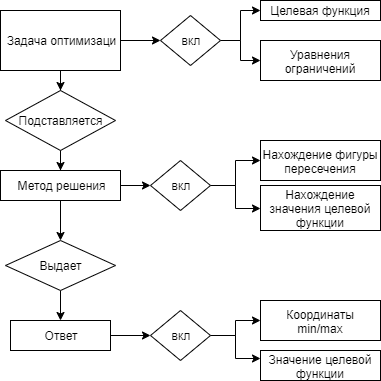


Рисунок 1.6 – модель предметной области

Концептуальная модель состоит задачи оптимизации, данные которой пользователю необходимо ввести (целевая функция, ограничения). После ввода пользователь нажимает кнопку «Решить», и данные задачи подставляются в метод решения. После решения, все неизвестные и результат целевой функции выводиться как ответ.

# Постановка задачи

## Определение требований к программной системе.

1. Приложение, позволяющее решить уравнение табличным симплекс методом
2. Поддержка ввода данных как с клавиатуры, так и с файла
3. Поддержка вывода данных как на экран, так и в файл
4. Дружелюбный интерфейс
5. Поддержка Java версии 9 и выше

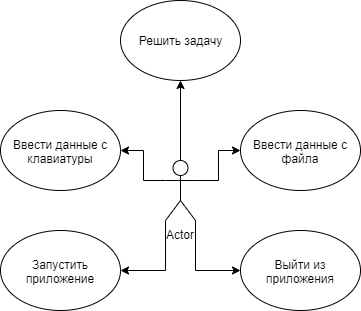


Рисунок 2.1 - диаграмма вариантов использования

## Описание аналогов системы.

В сети Интернет можно найти большое количество работ, которые реализуют решение данных задач. Практически все они представляют собой сайты, так что для решения уравнений необходимо подключение к интернету. Примером таким сайтов являются «math.semestr.ru»

Мой курсовой проект представляет собой приложение для ПК. Программа не требует подключения к интернету и имеет функции загрузки данных в файл.

### 2.3 Обзор и обоснование выбора средств реализации

Eclipse — свободная интегрированная среда разработки модульных кроссплатформенных приложений. Является бесплатной программной платформой с открытым исходным кодом, контролируется организацией Eclipse Foundation. Написана на языке программирования Java и основной целью ее создания является повышение продуктивности процесса разработки программного обеспечения. Претендует на статус наиболее популярной Java IDE и является единственным конкурентом такой мощной платформы как NetBeans. Но в отличие от NetBeans который для создания элементов пользовательского интерфейса использует платформо-независимую библиотеку Swing, в Eclipse используется платформо-зависимая библиотека Standard Widget Toolkit.

Достоинства: Постоянное обновление версий среды разработки, поддержка многих языков (в том числе и русского), является бесплатной, поддержка многих языков программирования, среда имеет промышленный уровень, является гибкой — то есть легко настраивается как под любую платформу, так и под любого пользователя.

Для создания интерфейса было применено:

JFrame – класс, реализующий отображение окон приложения.

JLabel – элементы отображения надписей

JButton – элементы отображения кнопок

JTextField – элементы отображение полей ввода

JTable – элемент отображения таблиц

# Проектирование

## Разработка архитектуры программного продукта

Поскольку приложение не сетевое, для лучшей скорости сделано однопоточным.

Главное окно будет содержать следующие элементы:

* Поле «Введите количество уравнений» - количество ограничений
* Поле «Введите количество неизвесных» - количество переменных (х1, х2…)
* Поле «Введите вашу функцию» - принимает коэффициенты функции
* Таблицу, созданную по заданным размерам для хранения строк ограничений
* Кнопку «Построить таблицу для заполнения»
* Кнопку «Загрузить данные из файла"» - для ввода данных из файла
* Кнопку «Решить» - для решение задачи и выдачи ответа
* Кнопку «Сохранить результаты в файл» - для сохранения ответа в файл

## Проектирование структур хранения данных.

Хранение данных извне не используется. Для хранения данных внутри приложения используются следующие переменные:

**private** String answerField = "";

**private** JTable table;

**private** JTextField fieldRowNumber;

**private** JTextField fieldColNumber;

**private** JTextField fieldFunction;

# Реализация

## Разработка архитектуры программного продукта

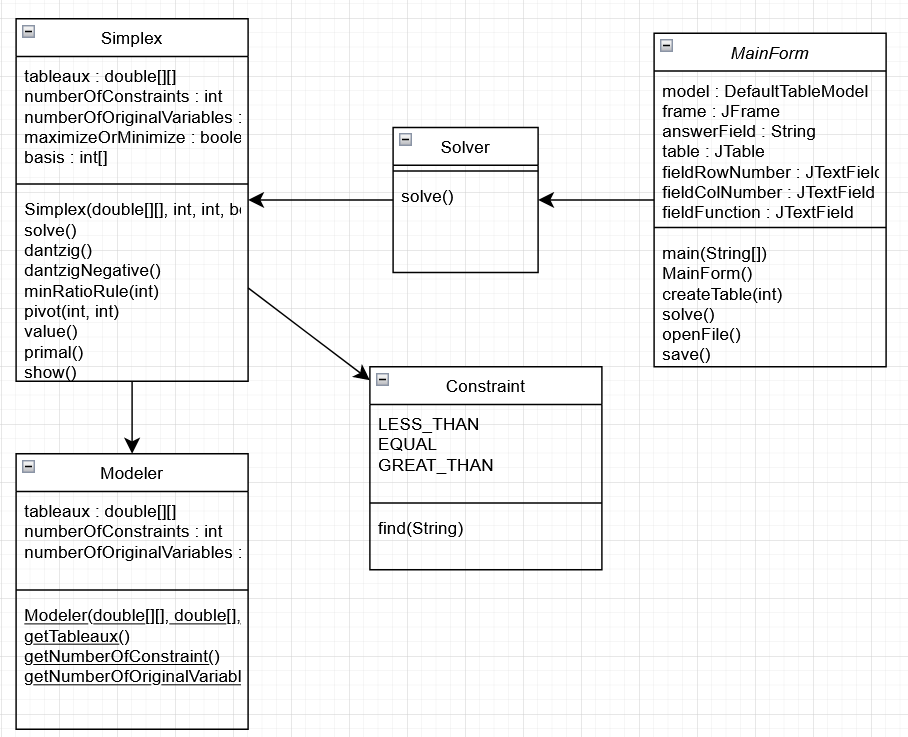


Рисунок 4.1 – Диаграмма классов

Главным классом является класс MainForm, у которого есть переменные для ввода данных таблицы, уравнений ограничений и целевой функции. После того как все данные введены, этот класс использует Solver и передает ему все необходимые данные для функции. После того как данные переданы – класс Simplex использует Modeler и Constraint для решения уравнения и возврата результата.

## Разработка интерфейса программного продукта

Главное окно

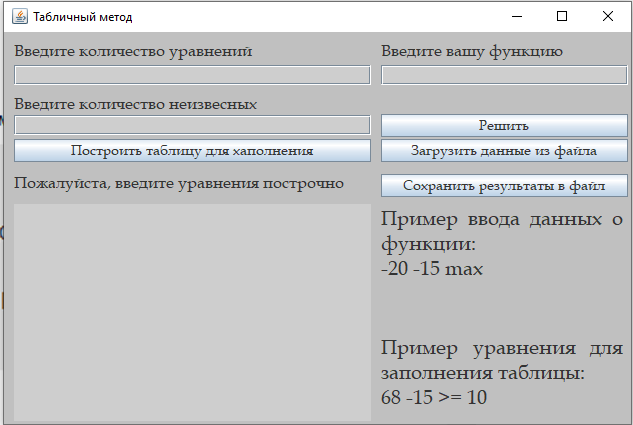


Рисунок 4.2.1 – Вид главного окна

Введем данные и создадим таблицу

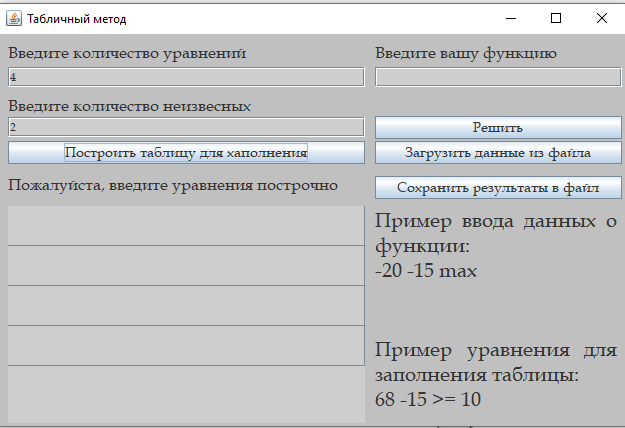


Рисунок 4.2.2 – Создание таблицы

Попробуем загрузить данные из файла

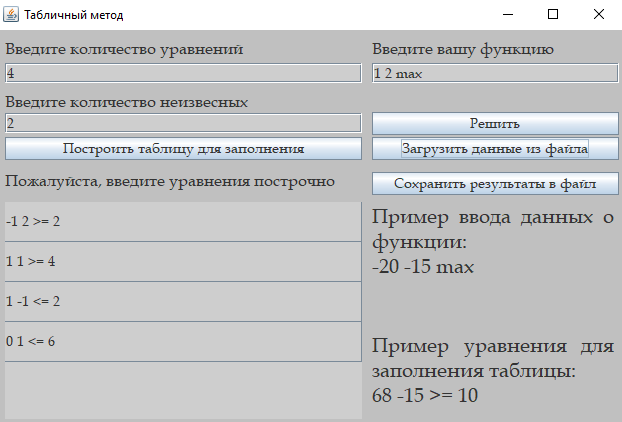


Рисунок 4.2.3 – Заполнение данных из файла

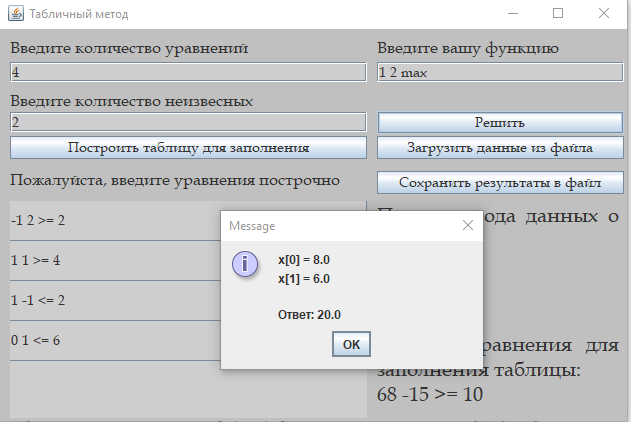


Рисунок 4.2.4 – Окно ответа

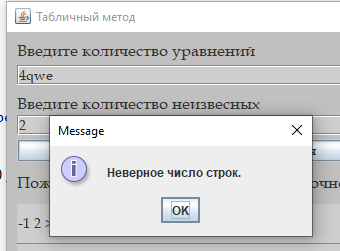


Рисунок 4.2.5 – Попытка решить с неверными данными

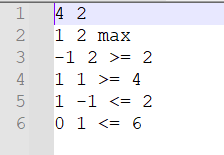


Рисунок 4.2.6 – Вид загружаемого файла

## Разработка алгоритмов реализации вариантов использования.

Общий алгоритм работы приложения:

1. Открывается форма приложения, переход к 2 или 8
2. Вводит количество ограничений, если еще не введена, переход к 3
3. Кнопка создать таблицу создает таблицу, переход к 4
4. Ввод ограничений, если еще не введены, переход к 5
5. Ввод целевой функции, переход к 6
6. Если нажата кнопка решение, решается задача и переход к 7
7. Открывается окно ответа, переход к 2
8. Завершение работы приложения
9. Создание таблицы

**void** createTable(**int** row) {

**try** {

table.removeAll();

table.setModel(model);

model.setRowCount(row);

model.setColumnCount(1);

table.setAutoResizeMode(JTable.***AUTO\_RESIZE\_ALL\_COLUMNS***);

table.setRowHeight(160 / row);

} **catch** (Exception ex) {

JOptionPane.*showMessageDialog*(**null**, "Ошибка при создании таблицы. \n" + "Проверьте правильность ввода");

}

}

1. Метод получения значений целевой функции и вывода на экран

**void** solve() {

**try** {

// Read function

String[] functionArray = fieldFunction.getText().split(" ");

**double**[] function = **new** **double**[functionArray.length - 1];

**for** (**int** i = 0; i < functionArray.length - 1; i++) {

String s = functionArray[i];

function[i] = Double.*parseDouble*(s);

}

**boolean** isFunctionMax = functionArray[functionArray.length - 1].equalsIgnoreCase("max");

// Read constraintLeftSide

**int** row = Integer.*parseInt*(fieldRowNumber.getText());

**int** col = Integer.*parseInt*(fieldColNumber.getText());

**double**[][] constraintLeftSide = **new** **double**[row][col];

Constraint[] constraintOperator = **new** Constraint[row];

**double**[] constraintRightSide = **new** **double**[row];

**for** (**int** i = 0; i < model.getRowCount(); i++) {

String[] rowData = table.getValueAt(i, 0).toString().split(" ");

**for** (**int** j = 0; j < col; j++) {

constraintLeftSide[i][j] = Integer.*parseInt*(rowData[j]);

}

constraintOperator[i] = Constraint.*find*(rowData[col]);

constraintRightSide[i] = Integer.*parseInt*(rowData[col + 1]);

}

answerField = Solver.*solve*(function, constraintLeftSide, constraintOperator, constraintRightSide,

isFunctionMax);

JOptionPane.*showMessageDialog*(**null**, answerField);

} **catch** (Exception ex) {

JOptionPane.*showMessageDialog*(**null**, "Некоректные данные. Проверьте введёные вами условия.");

}

}

# 5. Тестирование

Проведем тестирование программы, пройдя все варианты взаимодействия. Запустим само приложение. Запуск происходит без проблем.

Попробуем найти решение не вводя значений.

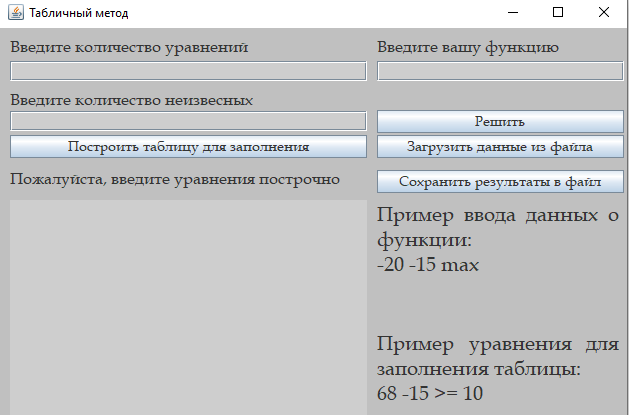


Рисунок 5.1 – Запуск главного окна

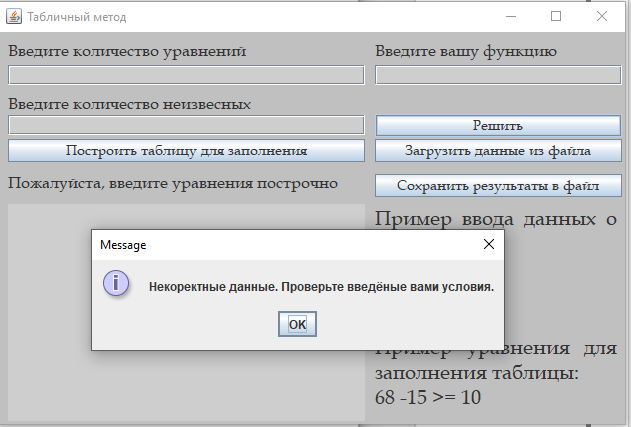


Рисунок 5.2 – Попытка решения задачи без введенных значений

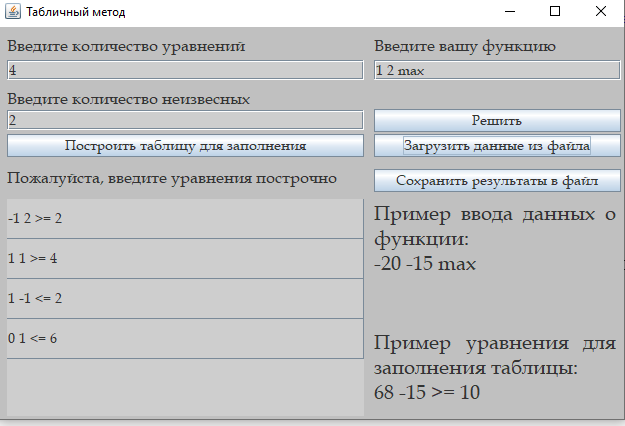


Рисунок 5.2 – Загрузка данных из файла

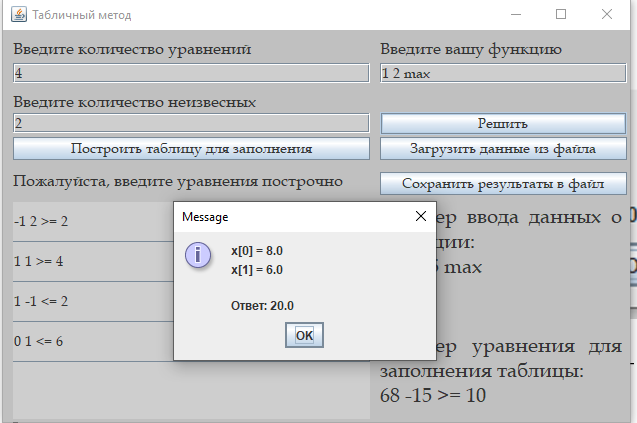


Рисунок 5.4 – Вывод ответа

Работа приложения происходит без ошибок

## Литература

1. Хорстманн, К.С. Java 2. Библиотека профессионала, том1. Основы. 8-е издание, : Пер. с англ. / К. С. Хорстманн, Г. Корнелл - М.: ООО «Вильямс», 2012. - 816 с.
2. Дирк, Л. Самоучитель Java 7: Пер. с нем. / Л. Дирк, П. Мюллер. – СПб: БХВ-Петербург, 2013. – 464 с.: илл.
3. Шилдт, Г. Java. Полное руководство. 8-е издание, : Пер. с англ. / Г. Шилдт. – М.:ООО «Вильямс», 2012. – 1104 с.
4. Блинов, И.Н. Java. Промышленное программирование./ И.Н. Блинов, В.С. Романчик –Минск: «Четыре четверти», 2013. – 896 с.
5. Онлайн калькулятор [Электронный ресурс]. – Режим доступа: math.semestr.ru. – Дата доступа: 01.12.2023.
6. Делюсь знаними [Электронный ресурс]. – Режим доступа: http://matecos.ru/. – Дата доступа: 01.12.2023.
7. Сайт преподователя экономики [Электронный ресурс]. – Режим доступа: http://galyautdinov.ru/. – Дата доступа: 01.12.2023.
8. Мир математики [Электронный ресурс]. – Режим доступа: https://matworld.ru/. – Дата доступа: 01.12.2023.

## Приложение А

Исходный код, откомпилированная программа и записка находятся на диске.