Содержание

[Введение 3](#_Toc533207633)

[1. Анализ объекта 4](#_Toc533207634)

[1.1 Описание предметной области. 4](#_Toc533207635)

[1.2 Построение концептуальной модели предметной области. 6](#_Toc533207636)

[2 Постановка задачи 7](#_Toc533207637)

[2.1 Определение требований к программной системе. 7](#_Toc533207638)

[2.2 Описание аналогов системы. 8](#_Toc533207639)

[2.3 Обзор и обоснование выбора средств реализации 9](#_Toc533207640)

[3 Проектирование 10](#_Toc533207641)

[3.1 Разработка архитектуры программного продукта 10](#_Toc533207642)

[3.2 Проектирование структур хранения данных. 10](#_Toc533207643)

[4 Реализация 11](#_Toc533207644)

[4.1 Разработка архитектуры программного продукта 11](#_Toc533207645)

[4.2 Разработка интерфейса программного продукта 13](#_Toc533207646)

[4.3 Разработка алгоритмов реализации вариантов использования. 15](#_Toc533207647)

[5 Тестирование 18](#_Toc533207648)

[Литература 20](#_Toc533207649)

[Приложение А 21](#_Toc533207650)

# Введение

Транспортная задача - это математическая задача линейного программирования специального вида о поиске оптимального распределения однородных объектов с минимизацией затрат на перемещение. Существует несколько методов решения транспортной задачи:

1. решение транспортной задачи методом потенциалов
2. решение транспортной задачи с использованием симплекс метода.

Мой проект представляет собой программу для решение транспортных задач методом потенциалом с нахождением опорного плана методом минимального элемента. Данная курсовая работа выполнена на языке программирования высокого уровня Java с использованием IDE Eclipse.

# Анализ объекта

## Описание предметной области.

Метод минимальных стоимостей получения опорного плана

Суть метода состоим в том, чтобы в первую очередь направлять груз в те пункты, где "расценки" в матрице стоимостей минимальны. Если клеток с наименьшими тарифами несколько, то заполняется любая из них.

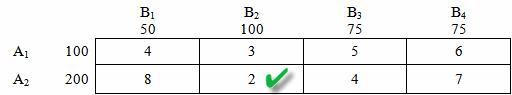


Рисунок 1.1 – Описание создания опорного плана № 1

Направляем 100 единиц груза из склада А2 в магазин В2.

Остатки на складе А2 — 100 единиц. Потребности магазина В2 выполнены.

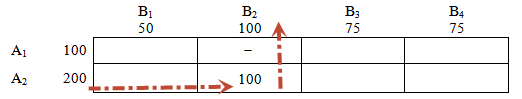


Рисунок 1.2 – Описание создания опорного плана № 2

Груз со склада А2 отправим в магазин, у которого стоимость перевозки ниже — магазин В3, так как  мин(4;7)=4

Размер поставки равен потребности магазина — 75. Остатки со склада 200-100-75=25 перенесем  в магазин В4.

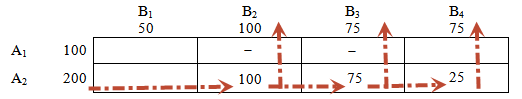


Рисунок 1.3 – Описание создания опорного плана № 3

Остается только раскидать груз со склада А1 по магазинам: В1 — 50 единиц, В4 —  75-25=50 единиц.

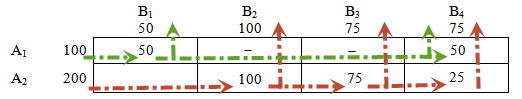


Рисунок 1.4 – Описание создания опорного плана № 4

Опорный план по методу минимальных стоимостей

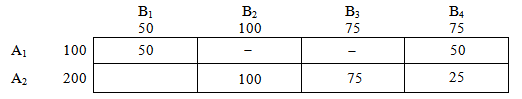


Рисунок 1.5 – Описание создания опорного плана № 5

## Построение концептуальной модели предметной области.

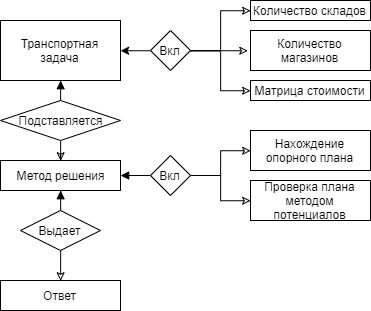


Рисунок 1.6 – модель предметной области

Концептуальная модель состоит транспортной задачи, данные которой пользователю необходимо ввести (склады, магазины, стоимость). После ввода пользователь нажимает кнопку «Решить», и данные задачи подставляются в метод решения. Сперва находится опорный план, затем используется метод потенциалов. Как только задача решена, метод выдает пользователю ответ.

# Постановка задачи

## Определение требований к программной системе.

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

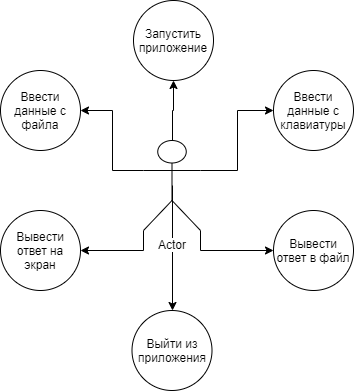


Рисунок 2.1 - диаграмма вариантов использования

## 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» и «matworld.ru».

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

### 2.3 Обзор и обоснование выбора средств реализации

Eclipse —интегрированная среда разработки ПО. Написана на языке Java в 2003 году. Также является кроссплатформенной. За счёт присоединяемых к этой среде дополнений — имеется возможность создавать программные продукты более чем на пяти языках программного кода.

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 «Количество потребностей»
* Поле «Количество запасов»
* Таблицу, созданную по заданным размерам для хранения стоимостей
* Кнопку «Создать таблицу»
* Кнопку «Рассчитать» - решение задачи
* Кнопку «Открыть файл» - для ввода данных из файла

## Проектирование структур хранения данных.

Для хранения данных извне будет использован текстовый файл, данные в нем будут храниться построчно

# Реализация

## Разработка архитектуры программного продукта



Рисунок 4.1 – Диаграмма классов

Главным классом является класс Main, у которого есть переменные для ввода количества магазинов и потребителей, а также матрицы суммы

При нажатии кнопки «Рассчитать», данные таблицы передаются в класс TestSymply2. Если пользователь ввел не все данные, некорректные или же не ввел их, ему выдастся соответствующее уведомление. В классе решения находится опорный план методом минимального элемента, находятся его потенциалы и план проверяется методом потенциалов. Как только с помощью метода будет получено решение, откроется новое окно Answer с ответом.

## Разработка интерфейса программного продукта

Главное окно

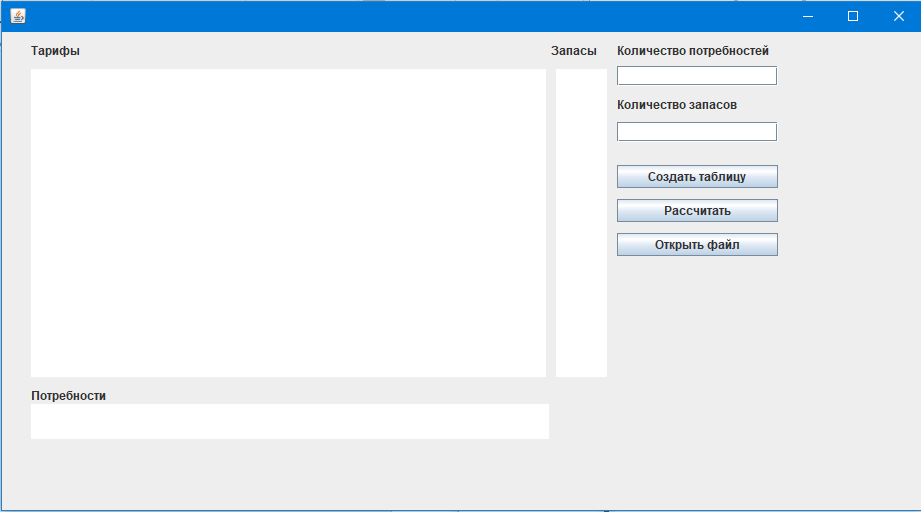


Рисунок 4.2.1 – Вид главного окна

Введем данные и создадим таблицу

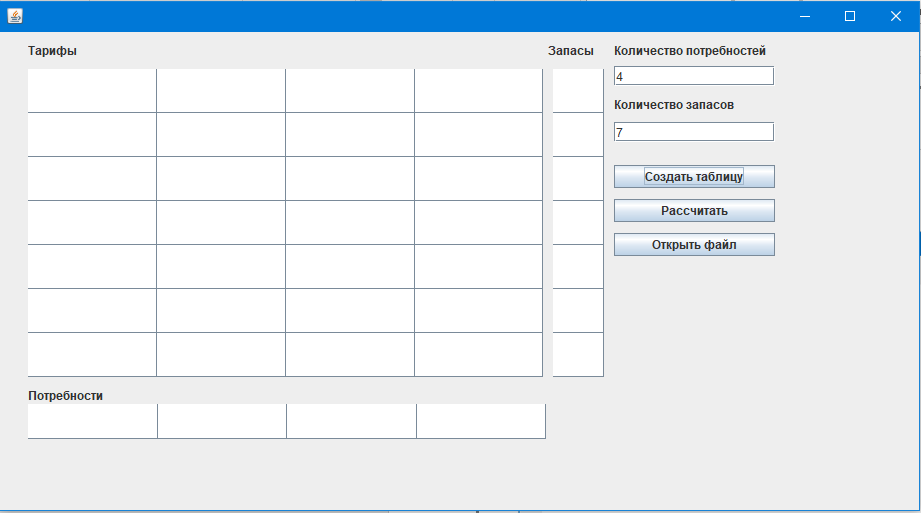


Рисунок 4.2.2 – Создание таблицы

Попробуем загрузить данные из файла

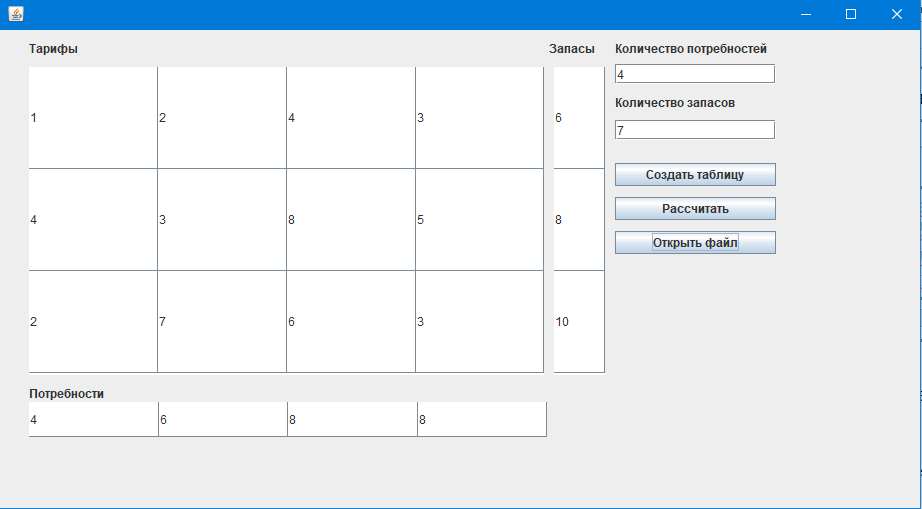


Рисунок 4.2.3 – Заполнение данных из файла

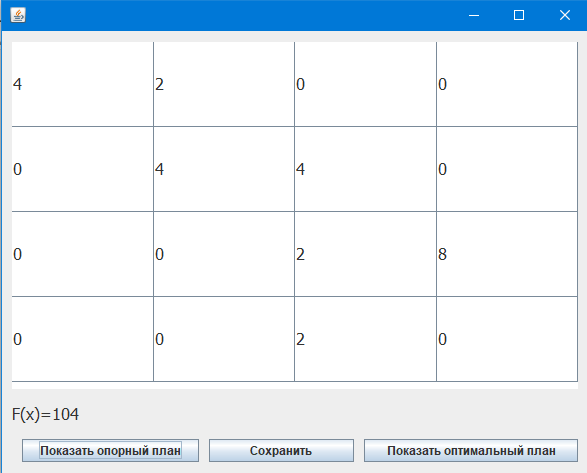


Рисунок 4.2.4 – Окно ответа. Опроный план

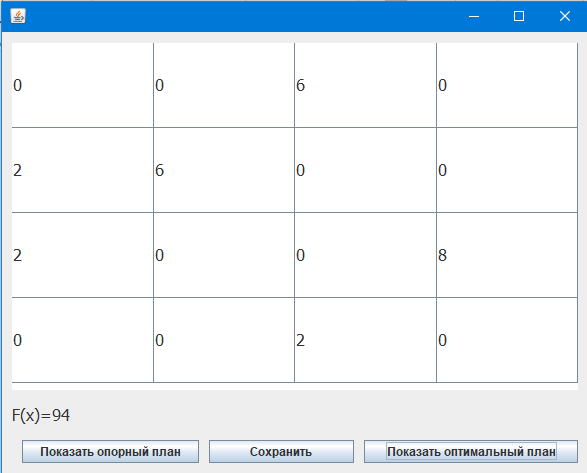


Рисунок 4.2.5 – Окно ответа. Оптимальный план

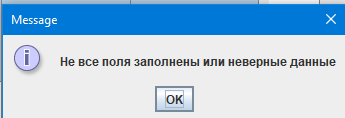


Рисунок 4.2.6 – Попытка решить без введенных данных

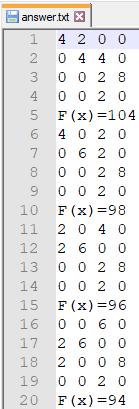


Рисунок 4.2.7 – Вид сохраненного файла

## Разработка алгоритмов реализации вариантов использования.

Общий алгоритм работы приложения:

1. Открывается главное меню, переход к 2 или 9
2. Вводит количество потребностей, если еще не введена, переход к 3
3. Вводит количество запасов от, если еще не введен, переход к 4
4. Кнопка создать таблицу создает таблицу, переход к 5
5. Ввод матрицы сумм, если еще не введена, переход к 6
6. Если нажата кнопка из фала, вывод окна выбора файла, заполнение матрицы, переход к 7
7. Если нажата кнопка решение, решается задача и переход к 8
8. Открывается окно ответа, переход к 2
9. Завершение работы приложения
10. Создание таблицы

void createTable(int rows, int collumns) {

table.removeAll();

table\_1.removeAll();

table\_2.removeAll();

if(rows<0 && collumns<0) {

rows = Integer.parseInt(textField.getText());

collumns = Integer.parseInt(textField\_1.getText());

}

table.setModel(model);

model.setColumnCount(rows);

model.setRowCount(collumns);

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

table.setRowHeight(308/collumns);

table\_1.setModel(model\_1);

model\_1.setColumnCount(rows);

model\_1.setRowCount(1);

table\_1.setAutoResizeMode(JTable.AUTO\_RESIZE\_ALL\_COLUMNS);

table\_1.setRowHeight(35);

table\_2.setModel(model\_2);

model\_2.setRowCount(collumns);

model\_2.setColumnCount(1);

table\_2.setAutoResizeMode(JTable.AUTO\_RESIZE\_ALL\_COLUMNS);

table\_2.setRowHeight(308/collumns);

}

1. Нахождение минимального опорного плана

void getMin(){

try {

JFileChooser fileopen = new JFileChooser();

int ret = fileopen.showDialog(null, "Открыть файл");

if (ret == JFileChooser.APPROVE\_OPTION) {

File file = fileopen.getSelectedFile();

sc= new Scanner(file);

int row = sc.nextInt();

int col = sc.nextInt();

createTable(col-1, row-1);

arr = new int[row][col];

for(int i=0; i<row;i++) {

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

arr[i][j]=sc.nextInt(); } }

TestSimply2.setArr(arr);

TestSimply2.initial();

TestSimply2.makeSbalansirowSadach();

for(int i=0; i<row-1;i++) {

model\_2.setValueAt(TestSimply2.sapasi[i], i, 0);

for(int j=0; j<col-1;j++) {

model.setValueAt(arr[i][j], i, j); } }

for(int i=0; i<col-1;i++) {

model\_1.setValueAt(TestSimply2.potrebit[i], 0, i);

}

}

}catch (Exception e) {

System.out.println(e);

JOptionPane.showMessageDialog(frame, "Ошибка при работе с файлом");

}

}

1. Поиск опорных точек для сдвига базисов методом

static int[] findRightUp(int i, int j) {

System.out.println("find right up");

try {

for(int collRight=j+1; collRight<tempCros[0].length;collRight++) {

if(tempCros[i][collRight]>0) {

//find up

for(int rowUp=i-1; rowUp>=0;rowUp--) {

if(tempCros[rowUp][collRight]>0) {

//find left

for(int collLeft=collRight-1; collLeft>=0;collLeft--) {

if(tempCros[rowUp][collLeft]>0) {

//find up finnaly

for(int rowDown=rowUp+1;rowDown<tempCros.length;rowDown++) {

if(tempCros[rowDown][collLeft]==-1 && i==rowDown && j==collLeft) {

System.out.println("i:"+i +",j:"+j +"; "

+ "i:"+i +",k:"+collRight +"; "

+ "k:"+rowUp +",l:"+collRight +"; "

+ "l:"+rowUp +",o:"+collLeft);

return new int[] {

i,j,

i,collRight,

rowUp, collRight,

rowUp,collLeft};

} }

} }

} }

}

}}catch(Exception ex) {}

return new int[] {-1};

}

# 5. Тестирование

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

Попробуем найти решение не вводя значениЙ.

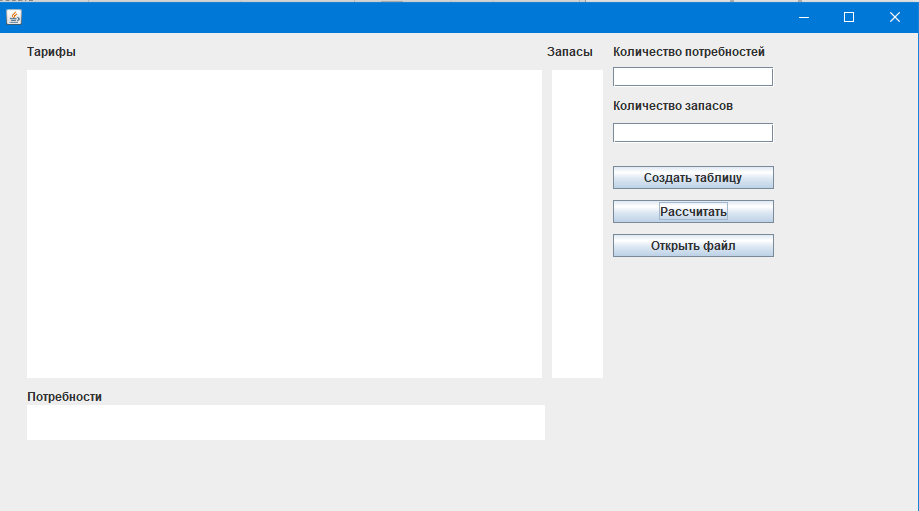


Рисунок 5.1 – Запуск главного окна

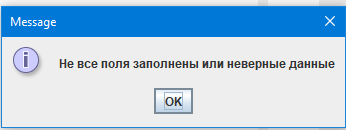


Рисунок 5.2 – Попытка запуска игры без введенных значений

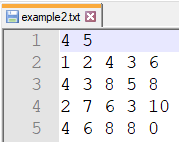
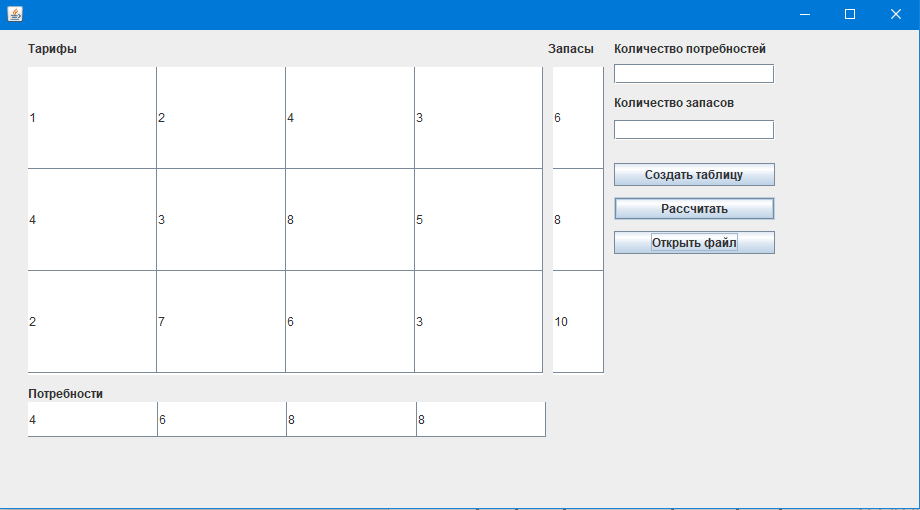
 

Рисунок 5.2 – Загрузка данных из файла

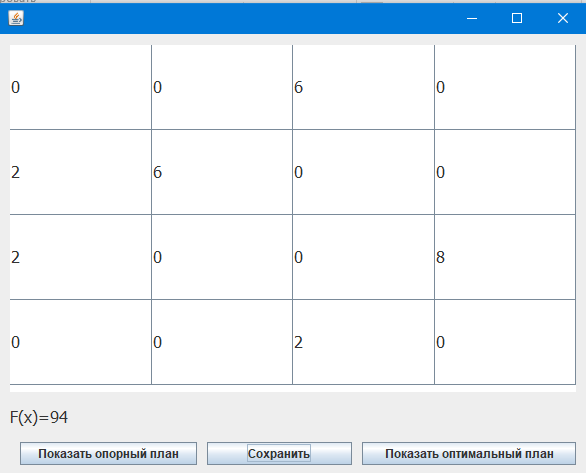


Рисунок 5.4 – Вывод ответа

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: 20.11.2019.
6. Делюсь знаними [Электронный ресурс]. – Режим доступа: http://matecos.ru/. – Дата доступа: 20.11.2019.
7. Сайт преподователя экономики [Электронный ресурс]. – Режим доступа: http://galyautdinov.ru/. – Дата доступа: 20.11.2019.
8. Мир математики [Электронный ресурс]. – Режим доступа: https://matworld.ru/. – Дата доступа: 20.11.2019.

## Приложение А

Исходный код, откомпилированная программа и записка находятся на диске.