|  |  |  |
| --- | --- | --- |
| ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ Федеральное государственное автономное образовательное  учреждение высшего профессионального образования Национальный исследовательский университет  "Высшая школа экономики"  Пермский филиал  Факультет бизнес-информатики | | |
| Кафедра информационных технологий в бизнесе | | |
| УДК 004.02 | | |
| **Проект**  «Формализация алгоритмов: машина Тьюринга» | | |
|  |  | Работу выполнили студенты группы БИ-15-1 1 курса факультета бизнес-информатики  Шерстобитова П.Д.  Самсонова А.О  Хафизов А.С.  Корзухин. А.А.  Принял:  Доцент кафедры ИТБ ,к.т.н  Лядова.Л.Н  “\_8\_\_\_\_”апреля 2016\_ г. |
| Пермь 20*16* | | |

Оглавление

[Глава.1 Постановка задачи 3](#_Toc447998034)

[Глава.2 Журнал тестов 4](#_Toc447998035)

[Версия 1.0 4](#_Toc447998036)

[Тесты 4](#_Toc447998037)

[Ошибки 7](#_Toc447998038)

[Версия 1.1 7](#_Toc447998039)

[Исправления 7](#_Toc447998040)

[Глава.3 Описание алгоритма 8](#_Toc447998041)

[1. Текстовое описание алгоритма 8](#_Toc447998042)

[2. Графическое описание алгоритма 13](#_Toc447998043)

[Глава.4 Описание порядка использования программы 21](#_Toc447998044)

[Руководство пользователя 21](#_Toc447998045)

[Глава.5 Исходный текст программы 24](#_Toc447998046)

[Заключение 38](#_Toc447998047)

[Библиографический список 39](#_Toc447998048)

[Приложения 40](#_Toc447998049)

# ****Постановка задачи****

В 1936 Алан Тьюринг предложил идею абстрактной вычислительной машины, которую назвал «Машиной Тьюринга». Машина Тьюринга (МТ)- это простое вычислительное устройство, которое состоит из бесконечно длинной, в обе стороны, ленты и автомата.

Доказано, что Машина Тьюринга по своим возможностям эквивалентна Машинам Поста и Нормальным алгорифмам Маркова. Эти интерпретаторы направлены на уточнение понятия алгоритма.

С каждой МТ связан алфавит символов *A* и набор внутренних состояний *Q* (всего *N* cостояний, обозначаемых *q*0, *q*1, ... *qn*− 1). В каждой ячейке ленты записан один символ из *A* (считается, что *A* содержит «пустой» символ λ «лямбда», а отсутствие записи в ячейке интерпретируется как запись символа λ). УУ может находиться в одном из состояний *q*0 ... *qn*− 1 (в начале работы УУ находится в состоянии *q*0). В каждый момент времени головка обозревает одну из ячеек ленты.

Задача: создание интерпретатора для решения различных прикладных задач на абстрактной вычислительной машине Тьюринга.

Первоначальный этап работы с проектом - распределение ролей. В ходе обсуждения наиболее эффективной схемы работы было решено распределить обязанности следующим образом:

* Корзухин А.А. – разработчик, менеджер;
* Шерстобитова П.Д. – аналитик, менеджер;
* Самсонова А.О. – технический писатель;
* Хафизов А.С. – тестировщик.

**Анализ входных данных:**

Таблица 1.1 (Исходные данные)

|  |  |  |
| --- | --- | --- |
| **Исходные данные** | **Классы** | **Ожидаемый результат** |
| Процесс создания алфавита | Ввести строку самому  Выгрузить из файла  Выбрать из предложенных | Алфавит |
| Задание значений на ленту | Выбранный из списка элемент алфавита | Пустая лента |
| Задание значений на таблицу состояний | Заранее заготовленная строка в формате: Элемент алфавита, направление («<», «>», «.»)и номер столбца перехода | Строка необходимого типа[[1]](#endnote-1) |
| Вещественное  Строка | Ошибка |
| Считывание данных из файла | Файл в подготовленном формате (заранее созданный в программе) | Продолжение работы внутри программы |
|  | | |

При создании интерфейса интерпретатора мы во много основывались на уже готовых программах. В первую очередь, хотелось бы указать интерпретатор с данного сайта <http://kpolyakov.spb.ru/prog/turing.htm>, обозначим её как «МТ2». Хочется отметить, что в данной программе, чтобы добавить элемент алфавита на ленту, нужно сделать 3 клика, в своей программе мы обошлись двумя. Также наша программа превосходит чужую процедурой создания алфавита (в МТ2 пользователь имеет возможность ввести алфавит только сам в строку), которая позволяет выбрать алфавит из заготовленных, ввести самому или задать из файла, безусловно, каждый шаг пользователя контролируется, и нет возможности задать неверные с точки зрения смысла значения. Однако в МТ2 визуализация пошагового выполнения алгоритма намного более наглядна, и существует процесс замедленного запуска программы, к сожалению, на реализацию данных функций нам не хватило времени, но несмотря на эти допущения в нашей программе можно подготавливать программы самим, посредством формирования их в файлах txt по определенному формату, который можно увидеть, открыв пример. Таким образом в программу можно завести данные более быстрым способом, нежели чем постепенно вводить всё в ячейки таблицы.

# Журнал тестов

## Версия 1.0

### Тесты

1. Создание алфавита: выбор варианта {0;1}

* Результат – вывод на экран таблицы для записи программы со строками 0, 1 и Е

1. Создание алфавита: выбор варианта {-1;0;1}

* Результат – вывод на экран таблицы для записи программы со строками -1, 0, 1 и Е

1. Создание алфавита: выбор варианта {0;1;2;3;4;5;6;7;8;9; +; -}

* Результат – вывод на экран таблицы для записи программы со строками 0-9, +, - и Е

1. Создание алфавита: ввод своего алфавита АБВГД

* Результат – вывод на экран таблицы для записи программы со строками А, Б, В, Г, Д и Е (ожидаемый результат –строки АБВГД и Е)

1. Создание алфавита: ввод своего алфавита АБ В ГД

* Результат – вывод на экран таблицы для записи программы со строками А, Б, В, Г, Д и Е (ожидаемый результат – строки АБ, В, ГД и Е)
* Примечание: результат оказался выше ожиданий – программа самостоятельно разделяет все символы, введенные пользователем без пробелов. Тем не менее, стоит написать уведомление о том, что в качестве элемента алфавита может использоваться только один символ.

1. Создание алфавита: ввод своего алфавита ЕЕЕЕЕ

* Результат – вывод на экран таблицы для записи программы со строками Е и Е (ожидаемый результат – вывод сообщения о недопустимости 2-х одинаковых элементов алфавита)

1. Создание алфавита: ввод своего алфавита 000111

* Результат – вывод на экран таблицы для записи программы со строками 0, 1 и Е

1. Создание алфавита: выбор варианта «Из файла» (сформированный самостоятельно алфавит «ABCD», хранящийся в файле«Alphabet.txt»)

* Результат – вывод на экран таблицы для записи программы со строками A, B, C, D и Е

1. Создание алфавита: выбор варианта «Назад»

* Результат – закрытие окна с выбором алфавита

1. Работа с таблицей: нажатие на кнопку «Добавление столбца»

* Результат – появление в таблице столбца q3

1. Работа с таблицей: нажатие на кнопку «Удаление столбца»

* Результат – удаление из таблицы последнего столбца q3

1. Работа с таблицей: попытка удаления единственного столбца q1 (предварительно при помощи кнопки «Удаление столбца» достигнут вид таблицы, в которой один столбец)

* Результат – сохранение таблицы в прежнем виде

1. Работа с таблицей: нажатие кнопки «Пошагово» при незаполненной таблице

* Результат – вывод сообщения «Вы не ввели необходимые команды»

1. Работа с таблицей: нажатие кнопки «Выполнить» при незаполненной таблице

* Результат – вывод сообщения «Вы не ввели необходимые команды»

1. Работа с лентой: нажатие кнопки «Left»

* Результат – смещение ленты на 1 символ влево

1. Работа с лентой: нажатие кнопки «Right»

* Результат – смещение ленты на 1 символ вправо

1. Работа с верхним меню «Файл»: открытие заранее сформированного автором файла «Test.txt»

* Результат – вывод в поле «Условие» сообщения «МОЕ УСЛОВИЕ», в поле «Комментарий к состояниям» сообщения «МОЙ КОММЕНТАРИЙ», создание таблицы для записи программы со строками, соответствующими символам алфавита, в соответствующие ячейки вписаны предписания (чтение из файла выполнено верно)
* Примечание: тем не менее, программа не работает – не введены все команды. Стоит либо доработать этот файл, либо удалить, так как имеется полностью работающий файл «Uchebnoe\_zadanie\_2.txt»

1. Работа с верхним меню «Файл»: открытие заранее сформированного автором файла «Uchebnoe\_zadanie\_2.txt»

* Результат – вывод в полях «Условие» и «Комментарий к состояниям» текста из файла, создание таблицы из строк – символов алфавита и столбцов – состояний q1 и q2. Программа выполняется корректно при нажатиях кнопок «Пошагово» и «Выполнить»

1. Работа с верхним меню «Файл»: открытие заранее сформированного самостоятельно файла «Двоичное число +2.txt»

* Примечание: файл оформлен в соответствии с рекомендациями:

Условие:

Число, записанное в двоичной системе счисления (двоичное число), должно быть увеличено на 2. Пример: 1111 -> 10001

Комментарий:

q1 - движение к концу

q2 - движение до первого разряда справа

q3 - движение до второго разряда справа

q4 - движение вправо до начала

q5 - завершение алгоритма

Алфавит:

01E

Лента:

1111

Программа(i строка - предписания для i состояния):

0>1 1>1 E<2

0<3 1<3 E<3

1<4 0<3 1.5

0<4 1<4 E>5

* Результат – сообщение об ошибке «Данные введены неверно» (ожидаемый результат – верно введенная таблица для записи программы с 3 строками (0, 1 и Е) и 5 столбцами (5 состояний))
* Примечание: при запуске данного теста в таблицу не добавились 3 недостающих столбца. Повторный запуск теста, но уже с предварительно самостоятельно добавленными столбцами также не решил проблему. Возможно, при создании теста стоит также указывать кол-во состояний, либо автоматически распознавать количество состояний по количеству строк в программе)

1. Ввод данных: попытка ввести программу из файла Двоичное число +2.txt в интерпретатор

* Результат – сообщение об ошибке «Данные введены неверно» при попытке ввести предписание в ячейку, соответствующую 3-й строке и 3-му столбцу (ожидаемый результат – корректный ввод программы и ее исполнение)
* Примечание: в 3-й строке и 3-м столбце должно было быть предписание 1.5, где «.» - это знак «stand», соответствующий команде «остановиться на текущей ячейке. Возможно, интерпретатор не распознает этот символ как «stand» и есть другой символ для его обозначения, либо же интерпретатор вообще не распознает «stand» (не задан символ, который будет обозначать это направление). В любом случае, необходимо указать об этом в руководстве пользователя, чтобы избежать недоразумений.

1. Работа с верхним меню «Файл»: сохранение программы с помощью кнопки «Сохранить как…»

* Результат – появление диалогового окна с возможностью выбрать папку для сохранения и ввода имени файла (ожидаемый рез-т – возможность выбора формата сохраняемого файла (как минимум один - .txt))
* Примечание: в сохраненном файле обнаружена другая кодировка символов, вследствие чего прочитать сохраненную информацию человеку невозможно. А при открытии этого файла программой возникает ошибка (интерпретатор «зависает»).

1. Работа с верхним меню «Файл»: сохранение программы с помощью кнопки «Сохранить»

* Примечание: был использован файл «Test.txt», в котором в поле «Условие» было дописано слово «условие», после чего и производилось сохранение
* Результат – файл сохранен, но информация в нем так же нечитаема, как и в тесте №22

### Ошибки

* Допускается ввод символа Е;
* Самостоятельно созданный файл с программой, в которой больше 2-х состояний, не считывается интерпретатором так, как должен;
* Нет возможности ввода направления «stand»;
* Файлы сохраняются в неизвестной кодировке, прочитать которую невозможно.

## Версия 1.1

### Исправления

* Не допускается ввод символа E;
* Файлы с программами с любым количеством состояний, считываются должным образом;
* Добавлена возможность ввода направления «stand» (по символу «.»);
* Проблема с кодировкой исправлена.

*В данной версии не было выявлено новых ошибок, были повторно проведены все тесты из предыдущей версии интерпретатора, которые оказались неудачными. Весь заявленный функционал интерпретатора машины Тьюринга работает.*

# Описание алгоритма

## Текстовое описание алгоритма

***Запись элемента на ленту***

Начало

Создание алфавита

Начало

Если алфавит создан, то

Считывание элементов ленты

Начало

Если элемент = E, то

Шрифт остается стандартным

Конец

Иначе

Начало

Выделить элемент жирным шрифтом

Конец

Конец

Иначе

Начало

Вывести "Вы не вывели алфавит"

Конец

Конец

**Выбор алфавита и создание таблицы для команд пользователя**

Начало

Открыть окно выбора файла с алфавитом

Начало

Если есть алфавит, то

Начало

Создание заголовков таблицы

Конец

Начало

Разблокировать кнопки работы приложения

Конец

Начало

Вывести алфавит

Конец

Начало

Заполнить пустые поля символом E

Конец

Начало

Очистить поля "Задание" и "Комментарий"

Конец

Конец

Конец

**Добавление столбца в таблицу**

Начало

Начало

Увеличить столбцы на 1

Конец

Начало

Присвоить элементу таблицы "q" и номер нового столбца

Конец

Начало

Создать двумерный массив размера таблицы и добавить 1 к количеству столбцов

Конец

Начало

От 0 до количества размера строк делать с шагом +1

Начало

От 0 до количества размера столбцов делать с шагом +1

Начало

Переписать элементы таблицы в новый массив

Конец

Конец

Конец

Начало

Присвоить массиву со значениями таблицы новый больший массив

Конец

Конец

**Удаление столбца таблицы**

Начало

Начало

Создать строку S

Присвоить S значение выделенного элемента таблицы

Конец

Логическая переменная flagco = false

Счетчик j = -1

Начало

Пока

j< длина алфавита

flagco = true

Начало

Если в S есть элемент алфавита номера j

Начало

Присвоить flagco = true

Конец

J + 1

Конец

Конец

Начало

Если flagco = true

Начало

Запись элемента

Конец

Конец

Конец

**Считывание элемента в двумерный массив**

Начало

Начало

Если позиция головки меньше числа столбцов ленты - 1, то

Счетчик i = 0

Начало

Если

Значение ленты пусто, то

Начало

Присвоить I количество столбцов матрицы -1

Конец

Иначе

Начало

Делать пока I > длины алфавита

Значение ленты не равно элементам алфавита

I + 1

Конец

Конец

Начало

Изменить значение в ленте на значение таблицы i + 1

Конец

Начало

Если направление по ленте "<"

Начало

Сдвиг по ленте влево

Конец

Конец

Иначе

Начало

Сдвиг по ленте вправо

Конец

Конец

Конец

**Открытие файла**

Начало

Открыть диалоговое окно

Начало

Выбрать файл

Конец

Начало

Заполнить ленту значениями E

Конец

Начало

Переписать информацию из файла в "Комментарии" и "Задания"

Конец

Начало

Считать алфавит из файла

Конец

Начало

От 0 до длины заданной файлом

Начало

Делать пока не конец таблицы

Начало

Считывание и заполнение элементов таблицы из файла

Конец

Конец

Начало

Генерация элемента ленты

Конец

Конец

Начало

Вывести таблицу на экран алфавита

Конец

Конец

**Сохранение файла**

Начало

Начало

Открыть поток для записи файлов

Конец

Начало

Записать в файл "Комментарии", "Задания" и алфавит

Конец

Начало

От 0 до количества элементов ленты делать с шагом -1

Начало

Элемент записываем в файл

Конец

Начало

Закрываем поток

Конец

Начало

Файл успешно сохранен

Конец

Конец

## Графическое описание алгоритма

Начало

Алфавит создан?

Считывание элемента ленты

Элемент равен Е?

Шрифт остается стандартным

Элемент выделяется жирным шрифтом

Вы не ввели алфавит!

Конец

+

+

-

-

Рис. 3.2.Блок-схема (Считывание элемента ленты)

Начало

Окно выбора файла с алфавитом

Есть алфавит?

Создаем заголовок таблицы

Разблокировка кнопок работы приложения

Вывод алфавита

Заполнение пустых полей символом Е

Очистка полей «Задание» и «Комментарии»

Конец

+

-

рис. 3.2 Блок-схема (Выбор алфавита и создание таблицы для команд пользователя)

Начало

Кол-во столбцов увеличиваем на 1

Элементу таблицы присваиваем «q» + номер нового столбца

От 0 до кол-ва строк

От 0 до кол-ва столбцов+1

Создаем дв. массив размера таблицы (+1 к кол-ву столбцов)

Переписываем элементы таблицы в новый массив

Присваиваем массиву со значениями таблицы новый больший массив

Конец

рис. 3.3 Блок-схема (Добавление столбца в таблицу)

Начало

Присваиваем массиву со значениями таблицы новый меньший массив

Конец

Кол-во столбцов больше 2?

Уменьшаем кол-во столбцов таблицы на 1

Создаем дв. массив размера таблицы (-1 к кол-ву столбцов)

От 0 до кол-ва строк

От 0 до кол-ва столбцов-1

Переписываем элементы таблицы в новый массив

-

+

рис. 3.4 Блок-схема (Удаление столбца таблицы)

рис. 3.5 Блок-схема (Считывание элемента в двумерный массив)

Начало

Создаем строку S и присваиваем ей значение выделенного элемента таблицы

Логическая переменная flagсо значением false

Счетчик j равный -1

Пока j меньше длины алфавита и flagне равен true

В S есть элемент алфавита номера j?

Flag присвоить true

J+1

Flag=true?

Запись элемента

Конец

-

-

+

+

Начало

Позиция головки меньше числа столбцов ленты -1?

Значение ленты пусто?

Счетчик i=0

присвоить кол-во столбцов таблицы -1

Пока Iменьше длинны алфавита и значение ленты не равно элем. алфавита

I+1

Изменяем значение в ленте на значение таблицы i+1

Направление «<»или «>»?

Сдвиг по ленте вправо

Сдвиг по ленте влево

<

>

-

+

+

-

Конец

Рис.3.6Блок - схема (Основной алгоритм)

Начало

Диалоговое окно

Выбор файла

Заполняем ленту значениями Е

Переписываем информацию из файла в «Комментарии» и «Задания»

Считываем из файла алфавит

От 0 до длины ленты, заданной в файле

Генерация элемента ленты

Считывание и заполнение элементов таблицы из файла

Пока не конец таблицы

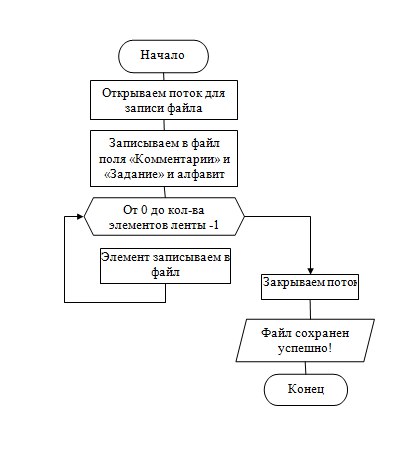
Вывод таблицы и алфавита на экран

Конец

рис. 3.7 Блок-схема (Открытие файла)

3.8 Блок-схема (Сохранение файла)

7 Блок-схема (сохранение файла)



* 1. Описание тестов

# Описание порядка использования программы

## Руководство пользователя

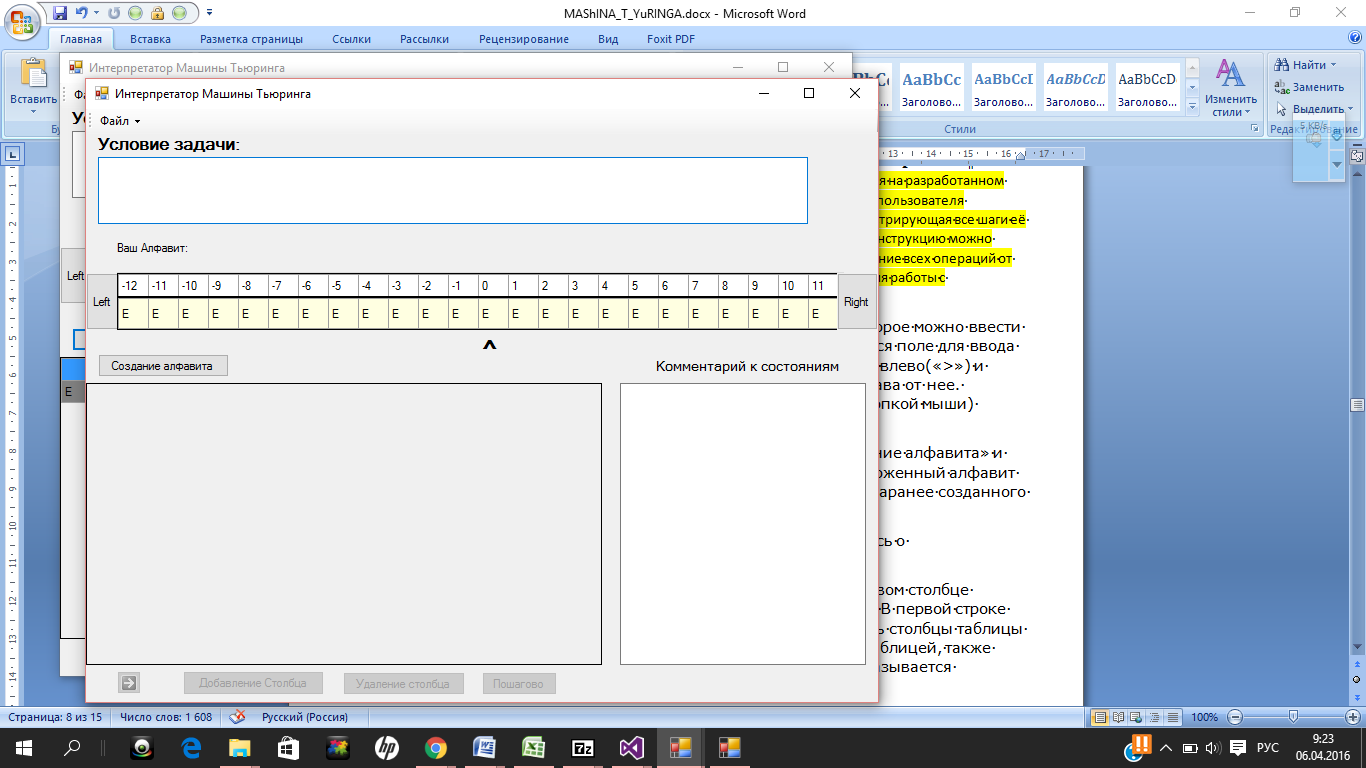
В верхней части программы находится поле редактора, в которое можно ввести условие задачи в свободной форме. В правой части- находится поле для ввода комментариев к состояниям программы. Лента перемещается влево(«>») и вправо(«<») с помощью кнопок, расположенных слева и справа от нее («Left» и «Right»). Двойным щелчком по ячейке ленты (или щелчком правой кнопкой мыши) можно изменить ее содержимое.

Рисунок 4.1 Интерфейс интерпретатора

Чтобы задать алфавит необходимо нажать на кнопку «Создание алфавита» и откроется новая форма, в которой можно выбрать уже предложенный алфавит или создать алфавит самому, а также выгрузить алфавит из заранее созданного файла.

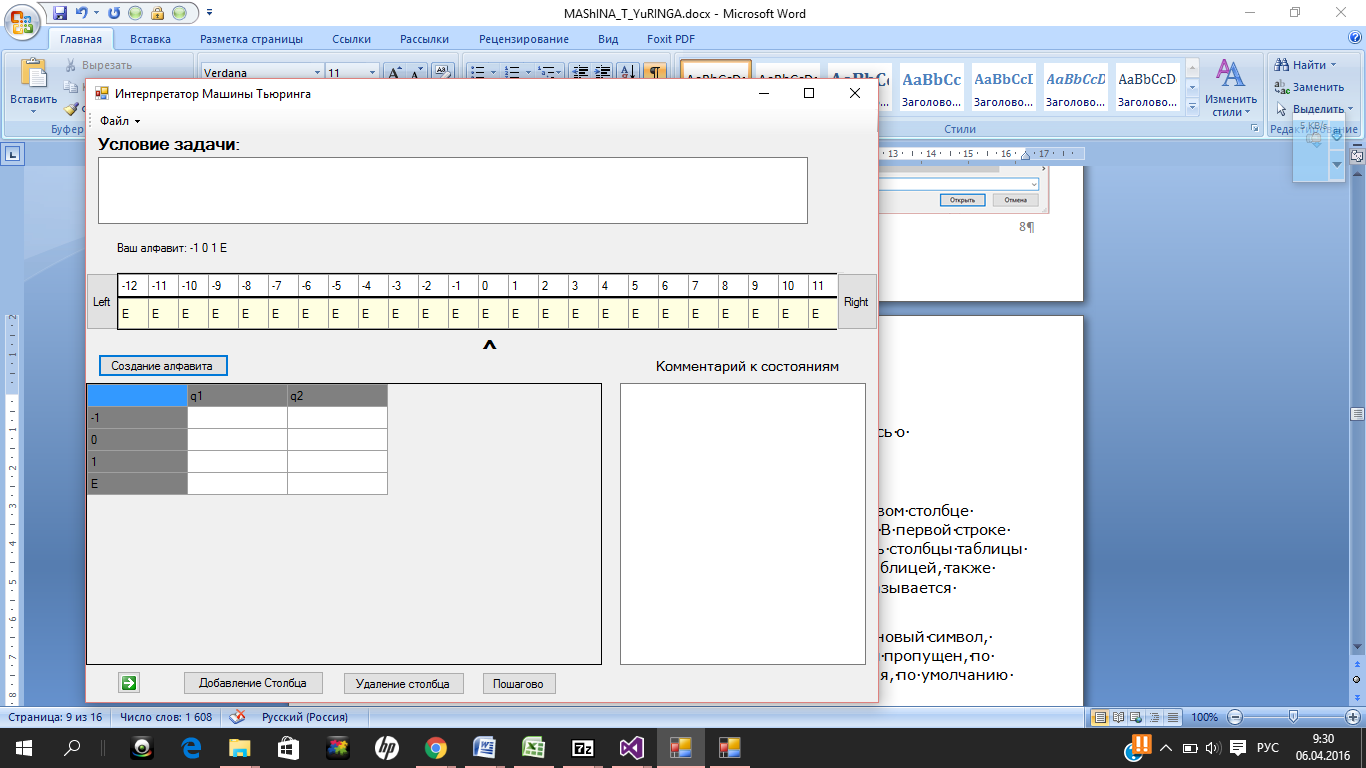
После задания алфавита под полем редактора появится запись о сформированном алфавите «Ваш алфавит…».

Рисунок 4.2 Поле, показывающее алфавит

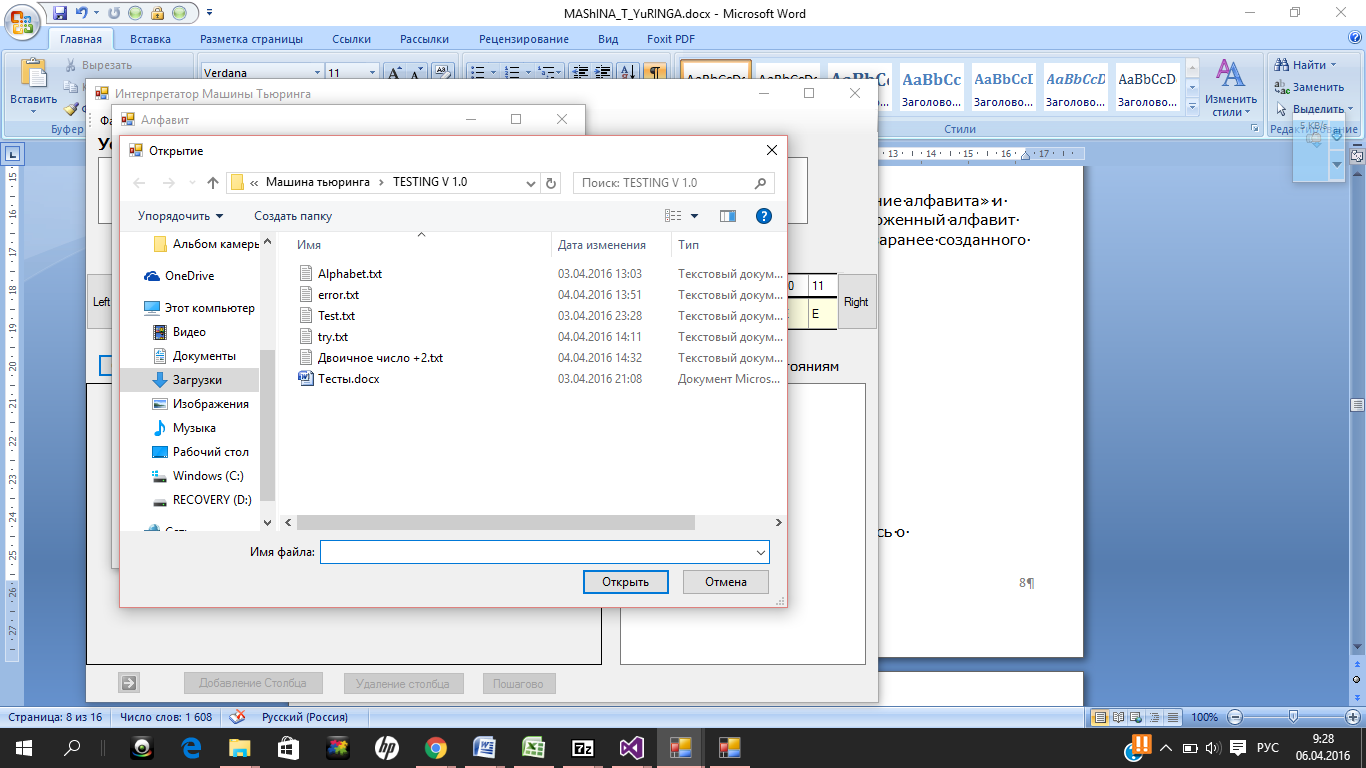
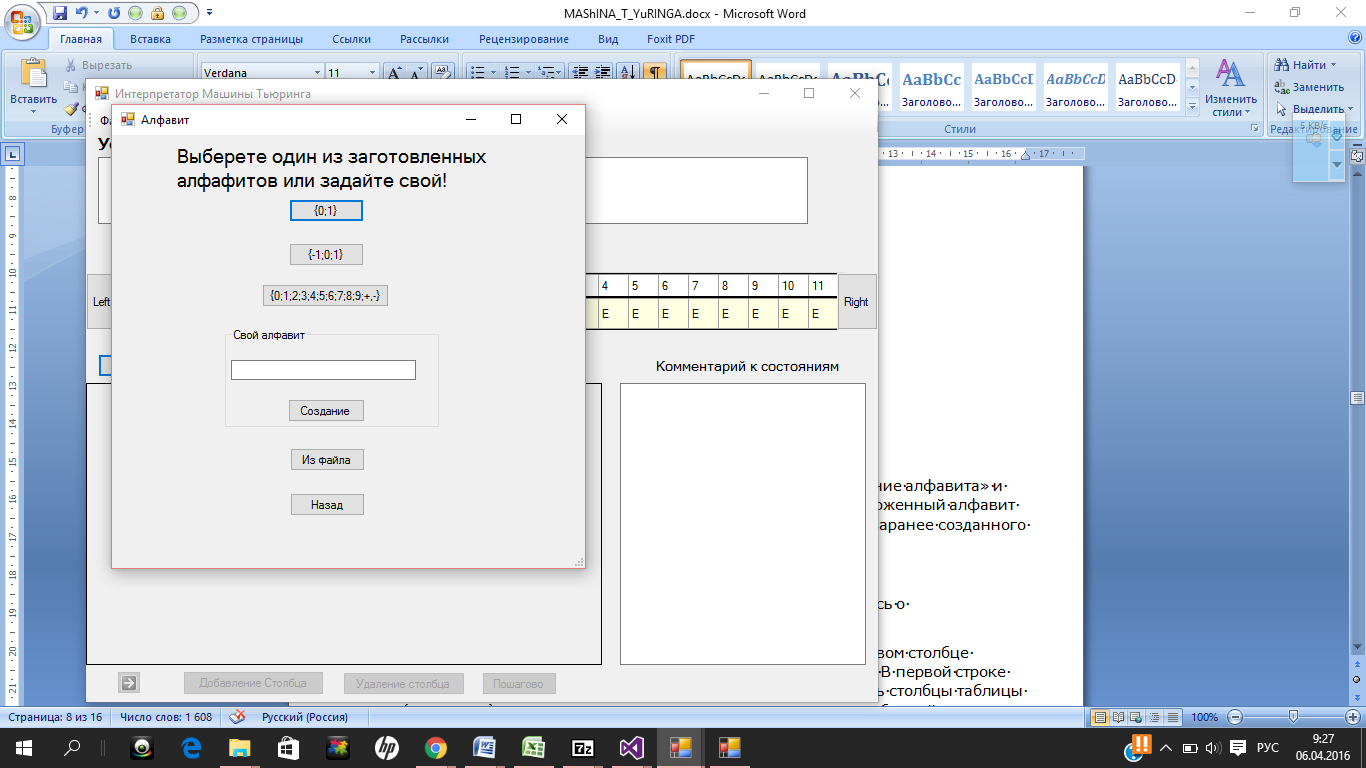
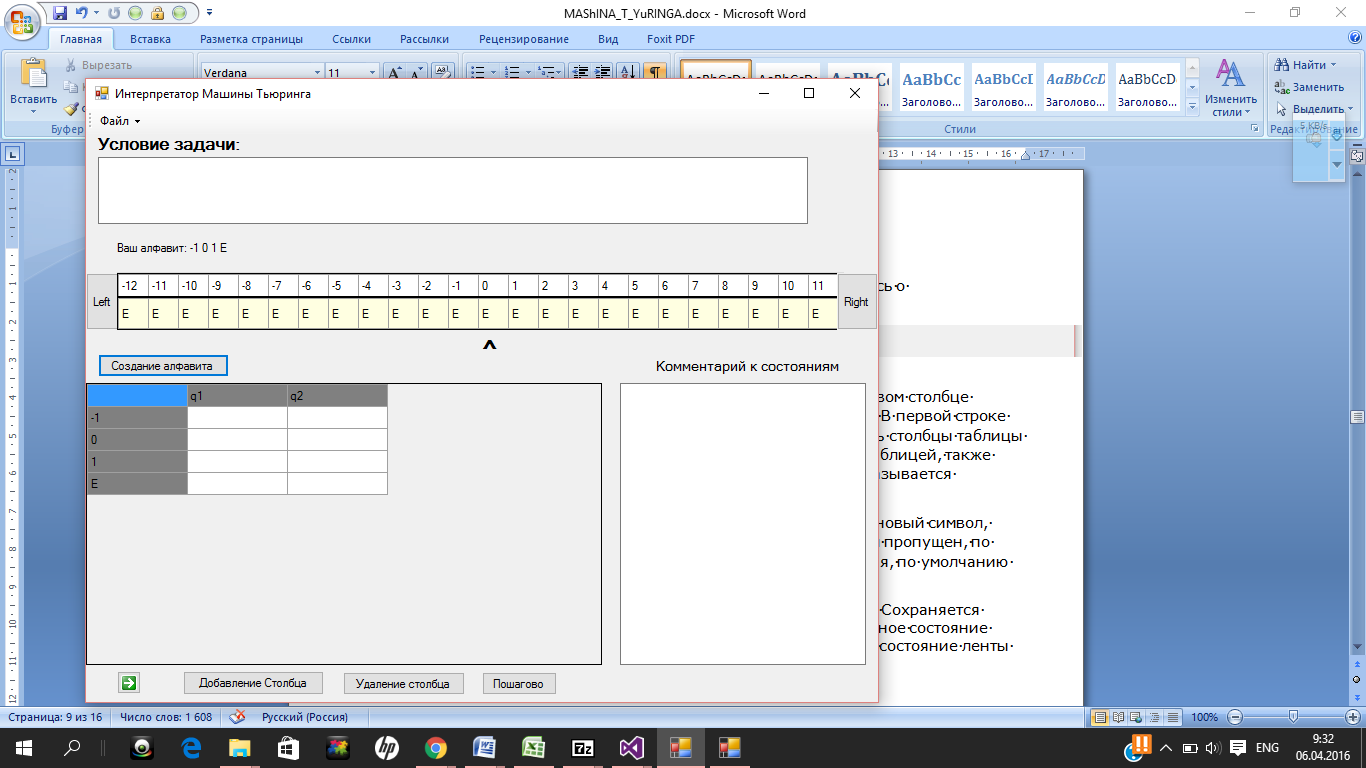
В таблице в нижней части окна набирается программа. В первом столбце записаны символы алфавита, он заполняется автоматически. В первой строке перечисляются все возможные состояния. Добавить и удалить столбцы таблицы (состояния) можно с помощью кнопок, расположенных под таблицей, также есть кнопка «пошагово», при нажатии которой детально показывается заполнение ленты.

Рисунок 4.5 Таблица состояний

Рисунок 4.4 Открытие файла

Рисунок 4.3 Алфавит

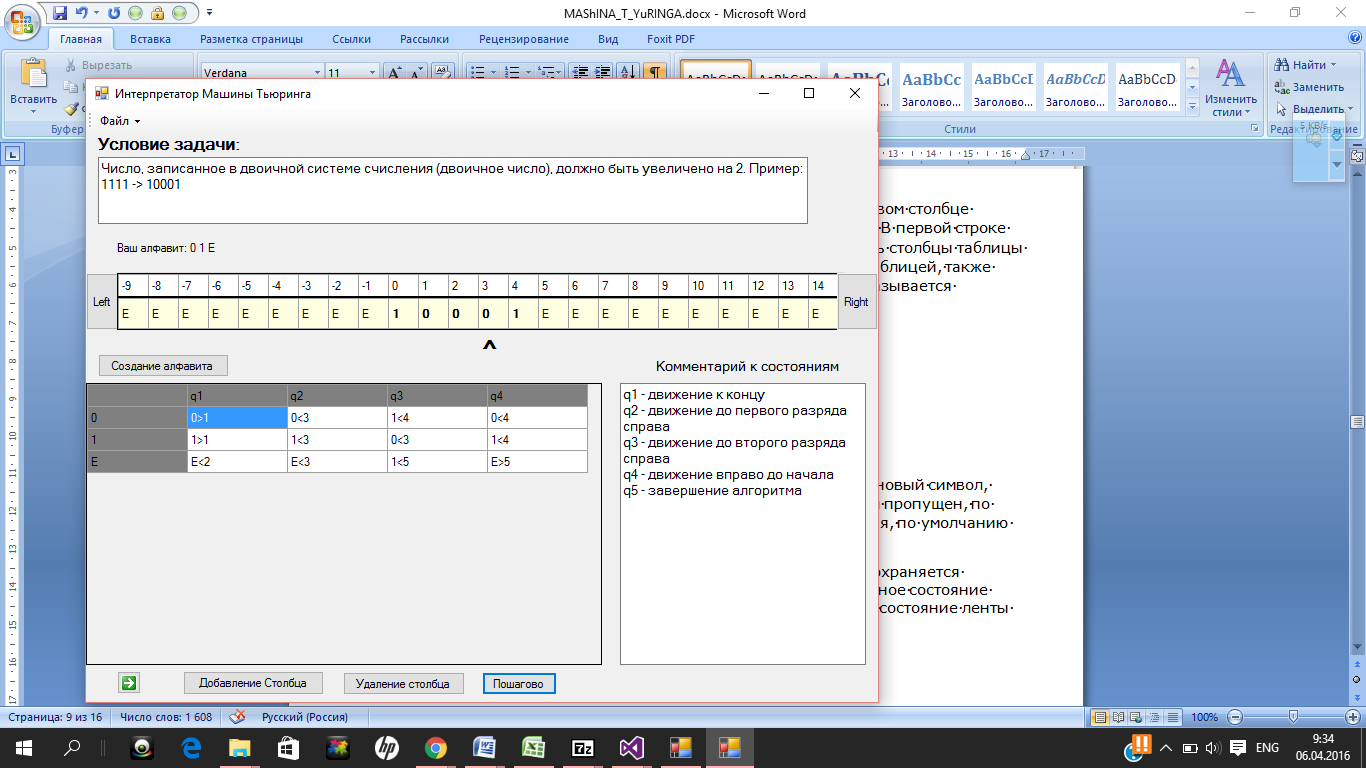
При вводе команды в ячейку таблицы сначала нужно ввести символ – элемент алфавита, затем направление перехода (“>”- сдвиг вправо, “<” – сдвиг влево, “.” - остановка) и номер состояния. Если символ пропущен, по умолчанию он не изменяется. Если пропущен номер состояния, по умолчанию состояние автомата не изменяется.

Рисунок 4.6 Заполненная таблица состояний

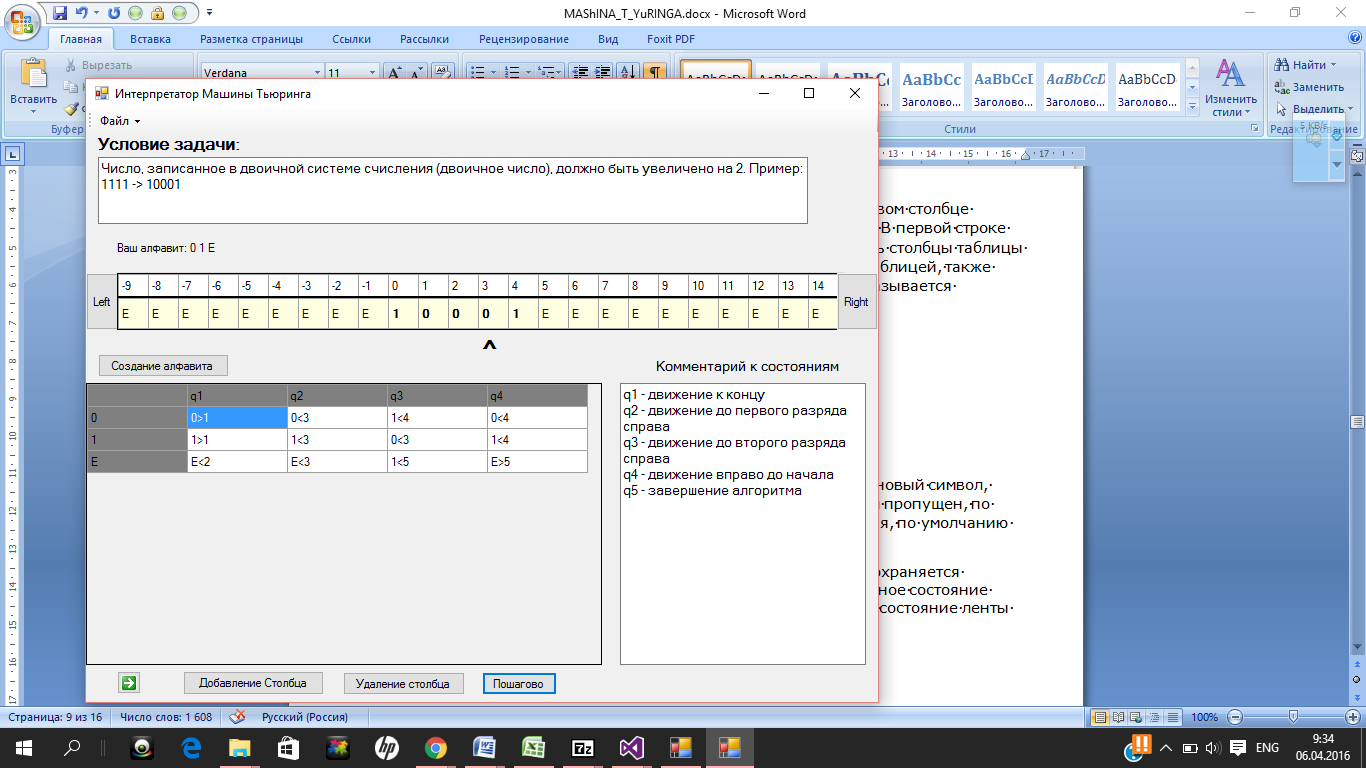
Задачи для машины Тьюринга можно сохранять в файлах формата \*.txt. Сохраняется условие задачи, алфавит, программа, комментарии и начальное состояние ленты. В дальнейшем эти файлы можно открыть через соответствующий пункт в меню файл (левый верхний угол) При загрузке задачи из файла и сохранении в файле состояние ленты автоматически записывается в буфер.

Рисунок 4.7 Пример задачи

# Исходный код программы

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.IO;

namespaceМашина\_тьюринга

{

publicpartialclassMain : Form

{

internalCode[,] tableCode = newCode[1,1]; //таблица с командами пользователя

CreateAlphabet ca = newCreateAlphabet();

public Main()

{

InitializeComponent();

}

//регион, отвечающий за интерфейс программы

#region Interface

privateintNumLenta = 201; // количествоэлементовналенте

privateint PosLenta = 88; // Крайняя левая позиция на ленте, для сдвига

privatevoid Main\_Load(object sender, EventArgs e)

{

EventTable.Refresh();

Lenta.RowCount = 1;

LentaNum.RowCount = 1;

Lenta.ColumnCount = NumLenta;

LentaNum.ColumnCount = NumLenta;

for (int i = 0; i <= NumLenta - 1; i++) //генерацияленты

{

LentaNum[i, 0].Value = i - 100;

LentaNum.Columns[i].Width = Lenta.Columns[i].Width = 30;

Lenta.Rows[0].Cells[i].Value = "E";

}

Lenta.FirstDisplayedScrollingColumnIndex = PosLenta;

LentaNum.FirstDisplayedScrollingColumnIndex = PosLenta;

buttonStart.Enabled = false;

buttonAddCol.Enabled = false;

buttonDelCol.Enabled = false;

buttonStep.Enabled = false;

СохранитьToolStripMenuItem.Enabled = false;

}

privatevoid Main\_Resize(object sender, EventArgs e)

{

//if (Lenta.FirstDisplayedScrollingColumnIndex > 12)

//{

Lenta.FirstDisplayedScrollingColumnIndex = midLenta - 12;

LentaNum.FirstDisplayedScrollingColumnIndex = midLenta - 12;

//}

}

//регион, относящийсякленте

#region Lenta\_code

privatevoid Right\_Click(object sender, EventArgs e) // сдвиглентывправо

{

if (Lenta.FirstDisplayedScrollingColumnIndex < NumLenta - 1)

{

Lenta.FirstDisplayedScrollingColumnIndex++;

LentaNum.FirstDisplayedScrollingColumnIndex++;

PosLenta = Lenta.FirstDisplayedScrollingColumnIndex;

midLenta++;

}

}

privatevoid Left\_Click(object sender, EventArgs e) //cдвиглентывлево

{

if (Lenta.FirstDisplayedScrollingColumnIndex > 0)

{

Lenta.FirstDisplayedScrollingColumnIndex--;

LentaNum.FirstDisplayedScrollingColumnIndex--;

PosLenta = Lenta.FirstDisplayedScrollingColumnIndex;

midLenta--;

}

}

privatevoid Lenta\_CellMouseClick(object sender, DataGridViewCellMouseEventArgs e) //считываниеэлементаленты

{

Lenta.ClearSelection();

if (ca.mas != null)

{

ChoseLentaElement cle = newChoseLentaElement(this);

cle.ShowDialog();

Lenta.Rows[0].Cells[e.ColumnIndex].Value = cle.LentaVal;

if (cle.LentaVal != "E")

Lenta.Rows[0].Cells[e.ColumnIndex].Style.Font = newFont(this.Font, FontStyle.Bold);

else

Lenta.Rows[0].Cells[e.ColumnIndex].Style.Font = newFont(this.Font, FontStyle.Regular);

}

else

{

MessageBox.Show("Выневвелиалфавит");

}

}

#endregion

//регион создания и форматирования таблицы с командами

#region CreatingFormatting\_MainTable

privatevoid CreateTableButton\_Click(object sender, EventArgs e)

{

ca.ShowDialog();

if (ca.mas != null)

{

EventTable.RowCount = 1;

EventTable.ColumnCount = 1;

EventTable.RowCount = ca.mas.GetLength(0) + 1;

EventTable.ColumnCount = 3;

for (int i = 0; i < ca.mas.Length; i++) // делаемподобиезаголовка

{

EventTable.Rows[i + 1].Cells[0].Style.BackColor = System.Drawing.Color.Gray;

EventTable.Rows[i + 1].Cells[0].Value = ca.mas[i];

EventTable.Rows[i + 1].Cells[0].ReadOnly = true;

}

EventTable.Rows[0].Cells[1].Style.BackColor = System.Drawing.Color.Gray;

EventTable.Rows[0].Cells[2].Style.BackColor = System.Drawing.Color.Gray;

EventTable.Rows[0].Cells[1].Value = "q1";

EventTable.Rows[0].Cells[2].Value = "q2";

EventTable.Rows[0].Cells[1].ReadOnly = true;

EventTable.Rows[0].Cells[2].ReadOnly = true;

EventTable.Rows[0].Cells[0].Style.BackColor = System.Drawing.Color.Gray;

EventTable.Rows[0].Cells[0].ReadOnly = true;

buttonStart.Enabled = true;

buttonAddCol.Enabled = true;

buttonDelCol.Enabled = true;

labelYourAlphabet.Text = "Вашалфавит: ";

for (int i = 0; i < ca.mas.Length; i++)

{

labelYourAlphabet.Text += ca.mas[i] + " ";

}

tableCode = newCode[ca.mas.GetLength(0) + 1, 4];

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

{

Lenta.Rows[0].Cells[i].Value = "E";

}

LastCodeColumnStep = 1;

buttonStep.Enabled = true;

textBoxTask.Clear();

textBoxComment.Clear();

}

} //выбор алфавита и создание таблицы для команд пользователя

privatevoid buttonAddCol\_Click(object sender, EventArgs e)

{

EventTable.ColumnCount++;

EventTable.Rows[0].Cells[EventTable.ColumnCount - 1].Value = "q" + Convert.ToString(EventTable.ColumnCount - 1);

EventTable.Rows[0].Cells[EventTable.ColumnCount - 1].Style.BackColor = System.Drawing.Color.Gray;

EventTable.Rows[0].Cells[EventTable.ColumnCount - 1].ReadOnly = true;

Code[,] bufTableCode = newCode[tableCode.GetLength(0), tableCode.GetLength(1) + 1];

for (int i = 0; i < tableCode.GetLength(0); i++)

{

for (int j = 0; j < tableCode.GetLength(1); j++)

{

bufTableCode[i, j] = tableCode[i, j];

}

}

tableCode = bufTableCode;

}

privatevoid buttonDelCol\_Click(object sender, EventArgs e)

{

if (EventTable.ColumnCount > 2)

{

EventTable.ColumnCount--;

Code[,] bufTableCode = newCode[tableCode.GetLength(0), tableCode.GetLength(1) - 1];

for (int i = 0; i < bufTableCode.GetLength(0); i++)

{

for (int j = 0; j < bufTableCode.GetLength(1); j++)

{

bufTableCode[i, j] = tableCode[i, j];

}

}

tableCode = bufTableCode;

}

}

privatevoid EventTable\_CellEndEdit(object sender, DataGridViewCellEventArgs e) //считываниеэлементатаблицывдвумерныймассив

{

try

{

string s = Convert.ToString(EventTable.Rows[e.RowIndex].Cells[e.ColumnIndex].Value);

bool flag = false;

int j = -1;

while ((j < ca.mas.Length) && (!flag))

{

j++;

flag = s.StartsWith(ca.mas[j]);

}

if (flag)

{

tableCode[e.RowIndex, e.ColumnIndex] = newCode();

tableCode[e.RowIndex, e.ColumnIndex].alph = ca.mas[j];

s = s.Remove(0, ca.mas[j].Length);

if (s[0] == '<' || s[0] == '>')

{

tableCode[e.RowIndex, e.ColumnIndex].direction = s[0];

s = s.Remove(0, 1);

try

{

if ((Convert.ToInt32(s) <= tableCode.GetLength(1)+1) && (Convert.ToInt32(s) >= 1))

{

tableCode[e.RowIndex, e.ColumnIndex].q = Convert.ToInt32(s);

}

else

{

MessageBox.Show("Данные введены неверно");

EventTable.Rows[e.RowIndex].Cells[e.ColumnIndex].Value = "";

}

}

catch

{

MessageBox.Show("Данныевведеныневерно");

EventTable.Rows[e.RowIndex].Cells[e.ColumnIndex].Value = "";

}

}

else

{

MessageBox.Show("Данные введены неверно");

EventTable.Rows[e.RowIndex].Cells[e.ColumnIndex].Value = "";

}

}

else

{

MessageBox.Show("Данные введены неверно");

EventTable.Rows[e.RowIndex].Cells[e.ColumnIndex].Value = "";

}

}

catch

{

MessageBox.Show("Данныевведеныневерно");

EventTable.Rows[e.RowIndex].Cells[e.ColumnIndex].Value = "";

}

}

#endregion

#endregion

//регион работы с алгоритмами интерпретатора

#region Code\_MainTable

privateint midLenta = 100;

privateint LastCodeColumnStep = 1;

privatevoid buttonStart\_Click(object sender, EventArgs e)

{

try

{

int LastCodeColumn = 1;

while (LastCodeColumn < tableCode.GetLength(1) - 1)

{

int i = 0;

if (Lenta.Rows[0].Cells[midLenta].Value == null)

{

i = EventTable.RowCount - 1;

}

else

{

while ((i < ca.mas.GetLength(0)) && (Convert.ToString(Lenta.Rows[0].Cells[midLenta].Value) != Convert.ToString(ca.mas[i])))

{

i++;

}

}

Lenta.Rows[0].Cells[midLenta].Value = tableCode[i + 1, LastCodeColumn].alph; //изменениезначенияналенте

if (tableCode[i + 1, LastCodeColumn].alph != "E")

{

Lenta.Rows[0].Cells[midLenta].Style.Font = newFont(this.Font, FontStyle.Bold);

}

switch (tableCode[i + 1, LastCodeColumn].direction) //сдвигполенте

{

case'<':

if (Lenta.FirstDisplayedScrollingColumnIndex > 0)

{

Lenta.FirstDisplayedScrollingColumnIndex--;

LentaNum.FirstDisplayedScrollingColumnIndex--;

PosLenta = Lenta.FirstDisplayedScrollingColumnIndex;

midLenta--;

}

break;

case'>':

if (Lenta.FirstDisplayedScrollingColumnIndex < NumLenta - 1)

{

Lenta.FirstDisplayedScrollingColumnIndex++;

LentaNum.FirstDisplayedScrollingColumnIndex++;

PosLenta = Lenta.FirstDisplayedScrollingColumnIndex;

midLenta++;

}

break;

case'.':

break;

}

LastCodeColumn = tableCode[i + 1, LastCodeColumn].q;

if (LastCodeColumn <= EventTable.ColumnCount - 1)

{

EventTable.Rows[i + 1].Cells[LastCodeColumn].Selected = true;

}

}

MessageBox.Show("Работаалгоритмазавершена!", "Успех!", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

catch

{

MessageBox.Show("Выневвелинеобходимыекоманды", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

privatevoid buttonStep\_Click(object sender, EventArgs e)

{

if (LastCodeColumnStep <= EventTable.ColumnCount - 1)

{

try

{

int i = 0;

if (Lenta.Rows[0].Cells[midLenta].Value == null)

{

i = EventTable.RowCount - 1;

}

else

{

while ((i < ca.mas.GetLength(0)) && (Convert.ToString(Lenta.Rows[0].Cells[midLenta].Value) != Convert.ToString(ca.mas[i])))

{

i++;

}

}

Lenta.Rows[0].Cells[midLenta].Value = tableCode[i + 1, LastCodeColumnStep].alph; //изменениезначенияналенте

if (tableCode[i + 1, LastCodeColumnStep].alph != "E")

{

Lenta.Rows[0].Cells[midLenta].Style.Font = newFont(this.Font, FontStyle.Bold);

}

switch (tableCode[i + 1, LastCodeColumnStep].direction) //сдвигполенте

{

case'<':

if (Lenta.FirstDisplayedScrollingColumnIndex > 0)

{

Lenta.FirstDisplayedScrollingColumnIndex--;

LentaNum.FirstDisplayedScrollingColumnIndex--;

PosLenta = Lenta.FirstDisplayedScrollingColumnIndex;

midLenta--;

}

break;

case'>':

if (Lenta.FirstDisplayedScrollingColumnIndex < NumLenta - 1)

{

Lenta.FirstDisplayedScrollingColumnIndex++;

LentaNum.FirstDisplayedScrollingColumnIndex++;

PosLenta = Lenta.FirstDisplayedScrollingColumnIndex;

midLenta++;

}

break;

}

LastCodeColumnStep = tableCode[i + 1, LastCodeColumnStep].q;

int iselect = 0;

if (Lenta.Rows[0].Cells[midLenta].Value == null)

{

iselect = EventTable.RowCount - 1;

}

else

{

while ((iselect < ca.mas.GetLength(0)) && (Convert.ToString(Lenta.Rows[0].Cells[midLenta].Value) != Convert.ToString(ca.mas[iselect])))

{

iselect++;

}

}

if (LastCodeColumnStep <= EventTable.ColumnCount - 1)

EventTable.Rows[iselect + 1].Cells[LastCodeColumnStep].Selected = true;

}

catch

{

MessageBox.Show("Выневвелинеобходимыекоманды", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

else

{

MessageBox.Show("Работаалгоритмазавершена!", "Успех!", MessageBoxButtons.OK, MessageBoxIcon.Information);

LastCodeColumnStep = 1;

}

}

#endregion

//регионработысфайлами

#region File\_Menu

string path;

privatevoidвыходToolStripMenuItem\_Click(object sender, EventArgs e)

{

Close();

}

privatevoidоткрытьToolStripMenuItem\_Click(object sender, EventArgs e) //открываемзаготовленныйобразец (+ формируем eventTable + формируемалфавит+ записываемвтекстбоксысмысл)

{

OpenFileDialog ofd = newOpenFileDialog();

ofd.Filter = "TXT files (\*.txt)|\*.txt";

if (ofd.ShowDialog() == DialogResult.OK)

{

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

{

Lenta.Rows[0].Cells[i].Value = "E";

}

StreamReader sr = newStreamReader(ofd.FileName, Encoding.Default);

path = ofd.FileName;

sr.ReadLine();

string s = sr.ReadLine();

textBoxTask.Text = "";

while (s != "Комментарий:")

{

textBoxTask.Text += s;

s = sr.ReadLine(); ;

}

s = sr.ReadLine();

textBoxComment.Text = "";

while (s!= "Алфавит:")

{

if (s!="")

textBoxComment.Text += s+ Environment.NewLine;

s = sr.ReadLine();

}

s = sr.ReadLine();

ca.mas = newstring[s.Length];

for (int i = 0; i < ca.mas.Length; i++)

{

ca.mas[i] = Convert.ToString(s[i]);

}

s = sr.ReadLine();

s = sr.ReadLine();

s = sr.ReadLine();

for (int i = 0; i < s.Length; i++)//генерациязначенийналенте

{

Lenta.Rows[0].Cells[midLenta + i].Value = s[i];

if (s[i] != 'E')

Lenta.Rows[0].Cells[midLenta + i].Style.Font = newFont(this.Font, FontStyle.Bold);

}

sr.ReadLine();

sr.ReadLine();

int col = 1;

tableCode = newCode[ca.mas.Length + 1,2];

string[] s1;

int colCount = 1;

do//считывание предписаний пользователя

{

s1 = sr.ReadLine().Split();

if (s1.Length == 1)

break;

for (int str = 1; str <= s1.Length; str++)

{

try

{

string stroka = s1[str - 1];

bool flag = false;

int j = -1;

while ((j < ca.mas.Length) && (!flag))

{

j++;

flag = stroka.StartsWith(ca.mas[j]);

}

if (flag)

{

tableCode[str, col] = newCode();

tableCode[str, col].alph = ca.mas[j];

stroka = stroka.Remove(0, ca.mas[j].Length);

if (stroka[0] == '<' || stroka[0] == '>' || stroka[0] == '.')

{

tableCode[str, col].direction = stroka[0];

stroka = stroka.Remove(0, 1);

try

{

if ((Convert.ToInt32(stroka) <= tableCode.GetLength(1) + 1) && (Convert.ToInt32(stroka) >= 1))

{

tableCode[str, col].q = Convert.ToInt32(stroka);

}

else

{

MessageBox.Show("Данныечастичновведеныневерно", "Ошибка!", MessageBoxButtons.OKCancel ,MessageBoxIcon.Warning);

tableCode[str, col] = null;

}

}

catch

{

MessageBox.Show("Данныевведеныневерно", "Ошибка!", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

}

else

{

MessageBox.Show("Данныевведеныневерно", "Ошибка!", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

}

else

{

MessageBox.Show("Данныевведеныневерно", "Ошибка!", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

}

catch

{

MessageBox.Show("Данныечастичновведеныневерно", "Ошибка!", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

}

col++;

colCount++;

tableCode = AddColumn(tableCode);

} while ((!sr.EndOfStream) && (s1.Length != 1));

sr.Close(); //начало генерации таблицы основной

EventTable.RowCount = 1;

EventTable.ColumnCount = 1;

EventTable.RowCount = ca.mas.GetLength(0) + 1;

EventTable.ColumnCount = colCount;

for (int i = 0; i < ca.mas.Length; i++) // делаемподобиезаголовка

{

EventTable.Rows[i + 1].Cells[0].Style.BackColor = System.Drawing.Color.Gray;

EventTable.Rows[i + 1].Cells[0].Value = ca.mas[i];

EventTable.Rows[i + 1].Cells[0].ReadOnly = true;

}

EventTable.Rows[0].Cells[0].Style.BackColor = System.Drawing.Color.Gray;

EventTable.Rows[0].Cells[0].ReadOnly = true;

for (int i = 1; i < colCount; i++)

{

EventTable.Rows[0].Cells[i].Style.BackColor = System.Drawing.Color.Gray;

EventTable.Rows[0].Cells[i].Value = "q" + i;

EventTable.Rows[0].Cells[i].ReadOnly = true;

}

buttonStart.Enabled = true;

buttonAddCol.Enabled = true;

buttonDelCol.Enabled = true;

labelYourAlphabet.Text = "Вашалфавит: ";

for (int i = 0; i < ca.mas.Length; i++)

{

labelYourAlphabet.Text += ca.mas[i] + " ";

}

LastCodeColumnStep = 1;

buttonStep.Enabled = true;

for (inti1=1; i1<EventTable.RowCount; i1++) //выводпредписанийизвнутреннегомассиванаэкран

{

for (int j1=1; j1<EventTable.ColumnCount;j1++)

{

try

{

EventTable.Rows[i1].Cells[j1].Value = Convert.ToString(tableCode[i1, j1].alph) + Convert.ToString(tableCode[i1, j1].direction) + Convert.ToString(tableCode[i1, j1].q);

}

catch

{ }

}

}

СохранитьToolStripMenuItem.Enabled = true;

}

}

privatevoidСохранитьToolStripMenuItem\_Click(object sender, EventArgs e) // перезаписьфайлаготового(сохранениеизменений)

{

StreamWriter sw = newStreamWriter(File.Open(path, FileMode.Open), Encoding.Default);

sw.WriteLine("Условие:");

sw.WriteLine(textBoxTask.Text);

sw.WriteLine();

sw.WriteLine("Комментарий:");

sw.WriteLine(textBoxComment.Text);

sw.WriteLine("Алфавит:");

string[] s = labelYourAlphabet.Text.Split(':');

s[1] = s[1].Replace(" ",""); //удаление всех пробелов в строке

sw.WriteLine(s[1]);

sw.WriteLine();

sw.WriteLine("Лента:");

for (int i=0; i<NumLenta; i++)

{

if (Lenta.Rows[0].Cells[i].Value.ToString() != "E")

sw.Write(Lenta.Rows[0].Cells[i].Value);

}

sw.WriteLine();

sw.WriteLine();

sw.WriteLine("Программа(i строка - предписаниядля i состояния):");

for (int col=1; col<tableCode.GetLength(1); col++)

{

for (int str=1; str<tableCode.GetLength(0);str++)

{

try

{

sw.Write(Convert.ToString(tableCode[str, col].alph) + Convert.ToString(tableCode[str, col].direction) + Convert.ToString(tableCode[str, col].q));

if (str != tableCode.GetLength(0) - 1)

sw.Write(" ");

}

catch

{ }

}

sw.WriteLine();

}

sw.Close();

MessageBox.Show("Файлуспешносохранён!", "Успешно", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

privatevoid сохранитьКакToolStripMenuItem\_Click(object sender, EventArgs e) //то же самое, что и сохранить, но с возможностью выбора файла

{

SaveFileDialog sfd = newSaveFileDialog();

sfd.Filter = "TXT files (\*.txt)|\*.txt";

if (sfd.ShowDialog() == DialogResult.OK)

{

StreamWriter sw = newStreamWriter(File.Create(sfd.FileName), Encoding.Default);

sw.WriteLine("Условие:"); //переписываниезаголовков

sw.WriteLine(textBoxTask.Text);

sw.WriteLine();

sw.WriteLine("Комментарий:");

sw.WriteLine(textBoxComment.Text);

sw.WriteLine();

sw.WriteLine("Алфавит:");

string[] s = labelYourAlphabet.Text.Split(':');

s[1] = s[1].Replace(" ", ""); //удаление всех пробелов в строке

sw.WriteLine(s[1]);

sw.WriteLine();

sw.WriteLine("Лента:");

for (int i = 0; i < NumLenta; i++)

{

if (Lenta.Rows[0].Cells[i].Value.ToString() != "E")

sw.Write(Lenta.Rows[0].Cells[i].Value);

}

sw.WriteLine();

sw.WriteLine();

sw.WriteLine("Программа(i строка - предписания для i состояния):");

for (int col = 1; col < tableCode.GetLength(1); col++) //считывание из таблицы с предписаниями информации в файл

{

for (int str = 1; str < tableCode.GetLength(0); str++)

{

try

{

sw.Write(Convert.ToString(tableCode[str, col].alph) + Convert.ToString(tableCode[str, col].direction) + Convert.ToString(tableCode[str, col].q));

if (str != tableCode.GetLength(0) - 1)

sw.Write(" ");

}

catch

{ }

}

sw.WriteLine();

}

sw.Close();

}

}

privateCode[,] AddColumn(Code[,] arr)

{

int numCol=1;

Random rand = newRandom(100);

int numStr = arr.GetLength(0);

if (numStr == 0)

numStr = 1;

Code[,] arrMain = newCode[numStr, arr.GetLength(1) + numCol];

for (int i = 0; i < numStr; i++)

{

for (int j = 0; j < arr.GetLength(1); j++)

{

arrMain[i, j] = arr[i, j];

}

}

return arrMain;

} //функция добавления 1 пустого столбца в таблицу

#endregion

}

publicpartialclassCreateAlphabet : Form

{

publicstring[] mas; // массив, в котором будет хранится алфавит

public CreateAlphabet()

{

InitializeComponent();

}

privatevoid alph1\_Click(object sender, EventArgs e) //формирование массива 4 разными способами (alph - объединение)

{

mas = newstring[3];

mas[0] = "0";

mas[1] = "1";

mas[2] = "E";

Hide();

}

privatevoid alph2\_Click(object sender, EventArgs e)

{

mas = newstring[4];

mas[0] = "A";

mas[1] = "B";

mas[2] = "C";

mas[3] = "E";

Hide();

}

privatevoid alph3\_Click(object sender, EventArgs e)

{

mas = newstring[13];

for (int i = 0; i < 10; i++)

mas[i] = Convert.ToString(i);

mas[10] = "+";

mas[11] = "-";

mas[12] = "E";

Hide();

}

privatevoid alph\_Click(object sender, EventArgs e)

{

try

{

string s = CreateMasText.Text;

for (int i = 0; i < s.Length; i++)

if (s[i] == ' ')

s = s.Remove(i, 1);

string [] bufMas = newstring[s.Length];

for (int i = 0; i < s.Length; i++)

{

bufMas[i] = Convert.ToString(s[i]);

if (bufMas[i]=="E" || bufMas[i] == "Е") // проверка на наличие зарезервированного символа

{

bufMas = null;

}

}

Array.Sort(bufMas); //сортировка для удобства

bufMas = bufMas.Distinct().ToArray(); //избавление от пробелов лишних

mas = newstring[bufMas.Length + 1];

for (int i = 0; i < bufMas.Length; i++)

mas[i] = bufMas[i];

mas[mas.Length - 1] = "E"; // добавление пустого символа в массив

Hide();

}

catch

{

MessageBox.Show("Нельзя использовать зарезервированное значение E !", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

mas = null;

CreateMasText.Text = "";

}

}

privatevoid buttonAlphCreate\_Click(object sender, EventArgs e)

{

try

{

OpenFileDialog ofd = newOpenFileDialog();

if (ofd.ShowDialog() == DialogResult.OK)

{

StreamReader sr = newStreamReader(ofd.FileName, Encoding.Default);

string s = sr.ReadToEnd();

for (int i = 0; i < s.Length; i++)

if (s[i] == ' ')

s = s.Remove(i, 1);

string[] bufMas = newstring[s.Length];

for (int i = 0; i < s.Length; i++)

bufMas[i] = Convert.ToString(s[i]);

Array.Sort(bufMas);

bufMas = bufMas.Distinct().ToArray();

mas = newstring[bufMas.Length + 1];

for (int i = 0; i < bufMas.Length; i++)

mas[i] = bufMas[i];

mas[mas.Length - 1] = "E";

Hide();

sr.Close();

}

}

catch

{

MessageBox.Show("Вашалфавитбылвведённеверно!");

mas = null;

}

}

privatevoid back\_Click(object sender, EventArgs e)

{

Hide();

}

}

publicpartialclassChoseLentaElement : Form

{

publicstring LentaVal="E"; //значение, которое передается в главную форму и отображается на ленте

publicMain main;

public ChoseLentaElement(Main main1)

{

InitializeComponent();

main = main1;

}

privatevoid ChoseLentaElement\_Load(object sender, EventArgs e) //менювыбораэлемента, которыйбудетзаписанналенту

{

if (main.EventTable.ColumnCount!=1)

{

ChoseElTable.RowCount = 1;

ChoseElTable.ColumnCount = main.EventTable.RowCount-1;

for (int i = 1; i <= ChoseElTable.ColumnCount; i++)

{

ChoseElTable.Rows[0].Cells[i-1].Value = main.EventTable.Rows[i].Cells[0].Value;

}

}

}

privatevoid ChoseElTable\_CellMouseClick(object sender, DataGridViewCellMouseEventArgs e) //непосредственнойзапоминаниеэтогоэлемента

{

LentaVal = Convert.ToString(ChoseElTable.Rows[0].Cells[e.ColumnIndex].Value);

Close();

}

}

classCode

{

publicstring alph;

publicchar direction;

publicint q;

public Code()

{

alph = "";

direction = ' ';

q = 0;

}

public Code(string Alph, char Direction, int Q)

{

alph = Alph;

direction = Direction;

q = Q;

}

publicoverridestring ToString()

{

return alph + direction + q;

}

}

}

# Заключение

За время работы над интерпретатором мы узнали о различных способах формализации алгоритмов, а также изучили на собственном опыте начальные этапы разработки программного обеспечения. Мы усовершенствовали свои навыки анализа задач, разработки алгоритмов и их решения, а также работу со структурой данных. Важным этапом работы было определение ролей в группе и последующее выполнение своих задач. Мы постарались добиться выполнения всех поставленных задач.

# Библиографический список

1. Методические рекомендации по подготовке и защите курсовых работ: Методические указания / НИУ ВШЭ – Пермь; Викентьева О.Л., Дерябин А.И., Лебедев В.В., Лядова Л.Н., Шестакова Л.В.,Корчагина В.П. – Пермь, 2013. – 18 с.
2. Лебедев В.В.Информационные технологии в офисной работе. Создание электронных документов с помощью текстового процессора MicrosoftWord 2007. НИУ ВШЭ – Пермь, Кафедра информационных технологий в бизнесе, 2010 г. – 218 с.
3. Т. А. Павловская "C# Программирование на языке высокого уровня" + cайт: [https://msdn.microsoft.com/ru-ru/library/67ef8sbd.aspx](https://vk.com/away.php?utf=1&to=https%3A%2F%2Fmsdn.microsoft.com%2Fru-ru%2Flibrary%2F67ef8sbd.aspx)

# Приложения





1. Строка необходимого типа – это строка,  в которой первый элемент: элемент алфавита, 2 элемент: направление, 3 элемент: номер столбца состояний. [↑](#endnote-ref-1)