ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Факультет компьютерных наук
Департамент программной инженерии

СОГЛАСОВАНО
Доцент, старший научный сотрудник
научно-учебной лаборатории ПОИС,
к.ф.-м.н.

_____ А.А. Каленкова
«___» _____ 2018 г.

УТВЕРЖДАЮ
Академический руководитель
образовательной программы
«Программная инженерия»

_____ В.В. Шилов
«___» _____ 2018 г.

Инв. № подл	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата
RU.17701729.507140-01 51 01-1				

ПРОГРАММА ПОСТРОЕНИЯ РЕГУЛЯРНОГО ВЫРАЖЕНИЯ ПО КОНЕЧНОМУ АВТОМАТУ

Программа и методика испытаний

ЛИСТ УТВЕРЖДЕНИЯ

RU.17701729.04.13-01 51 01-1-ЛУ

Исполнитель:
студент группы БПИ173
_____ /Таранцова Полина Дмитриевна /
«___»_____ 2018 г.

2018

УТВЕРЖДЕНО
RU.17701729.04.13-01 51 01-1-ЛУ

ПРОГРАММА ПОСТРОЕНИЯ РЕГУЛЯРНОГО ВЫРАЖЕНИЯ ПО КОНЕЧНОМУ АВТОМАТУ

Программа и методика испытаний

RU.17701729.04.13-01 51 01-1

Листов 26

Инв. № подл	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата
RU.17701729.507140-01 51 01-1				

2018

Оглавление

Аннотация 3
1. Объект испытаний 5
2. Цель испытаний 5
3. Требования к программе 5
3.1.Основные функции: 5
3.2.Дополнительные функции: 5
4. Требования к программной документации 6
5. Средства и порядок испытаний 6
5.1. Технические средства 6
5.2. Программные средства 6
5.3 Порядок проведения испытаний 7
6. Методы испытаний 7
6.1.Испытание выполнений требований, связанных с работой программы с автоматом, загруженным в программу в формате .tsml. 8
6.2.Испытание выполнений требований, связанных с построением автомата в программе и работы с построенным автоматом. 13
ПРИЛОЖЕНИЕ 1. ТЕРМИНОЛОГИЯ. 22
ПРИЛОЖЕНИЕ 2. СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ 24
ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ 26

Аннотация

Программа и методика испытаний – это документ, в котором содержится информация о программном продукте, а также полное описание приемочных испытаний для данного программного продукта.

Настоящая Программа и методика испытаний для «Программы построения регулярного выражения по конечному автомату» содержит следующие разделы: «Объект испытаний», «Цель испытаний», «Требования к программе», «Требования к программной документации», «Средства и порядок испытаний», «Приложения» [7].

В разделе «Объект испытаний» указано наименование и область применения программы.

В разделе «Цель испытаний» указана цель проведения испытаний. Раздел «Требования к программе» содержит основные требования к программе, которые подлежат проверке во время испытаний.

Раздел «Требования к программным документам» содержит состав программной документации, которая представляется на испытания.

Раздел «Средства и порядок испытаний» содержит информацию о технических и программных средствах, которые следует использовать во время испытаний, а также порядок этих испытаний.

Раздел «Методы испытаний» содержит информацию об используемых методах испытаний.

Настоящий документ разработан в соответствии с требованиями:

1) ГОСТ 19.101-77 Виды программ и программных документов [1];

2) ГОСТ 19.102-77 Стадии разработки [2];

3) ГОСТ 19.103-77 Обозначения программ и программных документов [3];

4) ГОСТ 19.104-78 Основные надписи [4];

5) ГОСТ 19.105-78 Общие требования к программным документам [5];

6) ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом [6];

7) ГОСТ 19.301-79 Программа и методика испытаний. Требования к содержанию и оформлению [7].

Изменения к данному документу оформляются согласно ГОСТ 19.603-78 [8], ГОСТ 19.604-78 [9].

Перед прочтением данного документа рекомендуется ознакомиться с терминологией, приведенной в Приложении 1 настоящей программы и методики испытаний.

1. Объект испытаний

Наименование: «Программа построения регулярного выражения по конечному автомату».

Название исполняемого файла: «Project.exe»

Краткая характеристика и область назначения: «Программа построения регулярного выражения по конечному автомату» несет научно-образовательный характер и предназначена для анализа работы алгоритмов.

2. Цель испытаний

Испытания, описанные ниже, проводятся с целью проверки разработанной программы на соответствие требованиям, изложенным в документе «Программа построения регулярного выражения по конечному автомату. Технической задание».

3. Требования к программе

Программа должна соответствовать следующим функциональным требованиям, изложенным в документе «Программа построения регулярного выражения по конечному автомату. Технической задание»:

3.1. Основные функции:

1) Создание конечного автомата;
2) Загрузка конечного автомата в программу в tsml-формате;
3) Приведение конечного автомата к регулярному выражению с использованием анимации, демонстрирующей метод удаления состояний;

3.2. Дополнительные функции:

1) Удаление и добавление состояний и переходов конечного автомата в процессе его создания;
2) Выбор начального и конечных состояний конечного автомата в процессе его создания;
3) Преобразование загруженного из внешнего источника конечного автомата из формата .tsml в формат .png;
4) Возможность выбрать построение регулярного выражения как с анимацией, так и без нее;
5) Возможность продвинуться к последующему и вернуться назад к предыдущему шагу алгоритма последовательного удаления состояний конечного автомата;
6) Возможность сохранить созданный конечный автомат в формате .tsml;
7) Возможность сохранить каждый шаг алгоритма последовательного удаления состояний в формате .png;

8) Предоставление регулярного выражения, полученное с применением алгоритма построения регулярного выражения с заменой подпроцессов и без него, а также анализ эффективности данного алгоритма, возможность сохранить полученные данные в формате .txt.

4. Требования к программной документации

На испытание должна быть представлена документация к программе в следующем составе:

1. «Программа построения регулярного выражения по конечному автомату». Техническое задание.
2. «Программа построения регулярного выражения по конечному автомату». Руководство оператора.
3. «Программа построения регулярного выражения по конечному автомату». Программа и методика испытаний.
4. «Программа построения регулярного выражения по конечному автомату». Текст программы.
5. «Программа построения регулярного выражения по конечному автомату». Пояснительная записка.

5. Средства и порядок испытаний

5.1. 5.1 Технические средства

Во время испытаний должны быть использованы следующие технические средства:

- процессор с частотой 2,2 Ггц или выше;
- 0.5 гигабайт (ГБ) (для 32-разрядной системы) или 1 ГБ (для 64-разрядной системы) оперативной памяти (ОЗУ);
- 0.5 гигабайт (ГБ) (для 32-разрядной системы) или 1 ГБ (для 64-разрядной системы) пространства на жестком диске;
- монитор с разрешением 1024×640 или более высоким;
- клавиатура и мышь.
- CD-ROM привод;

5.2. 5.2 Программные средства

Во время испытаний должны быть использованы следующие программные средства:

- операционная система Windows 7 или более поздняя версия;
- установленный Microsoft .NET Framework версии не ниже 4.0.

5.3 Порядок проведения испытаний

Испытания должны проводиться в следующем порядке:

1. Выполнить начальную загрузку операционной системы.
2. Вставить в компьютер информационно-накопительное устройство, содержащее программный продукт.
3. Открыть папку «Expression Miner».
4. Запустить программу двойным щелчком левой кнопки мыши по файлу «`Project.exe`».
5. Провести необходимые испытания, описанные ниже в разделе «Методы испытаний».
6. Завершить работу программы.

6. Методы испытаний

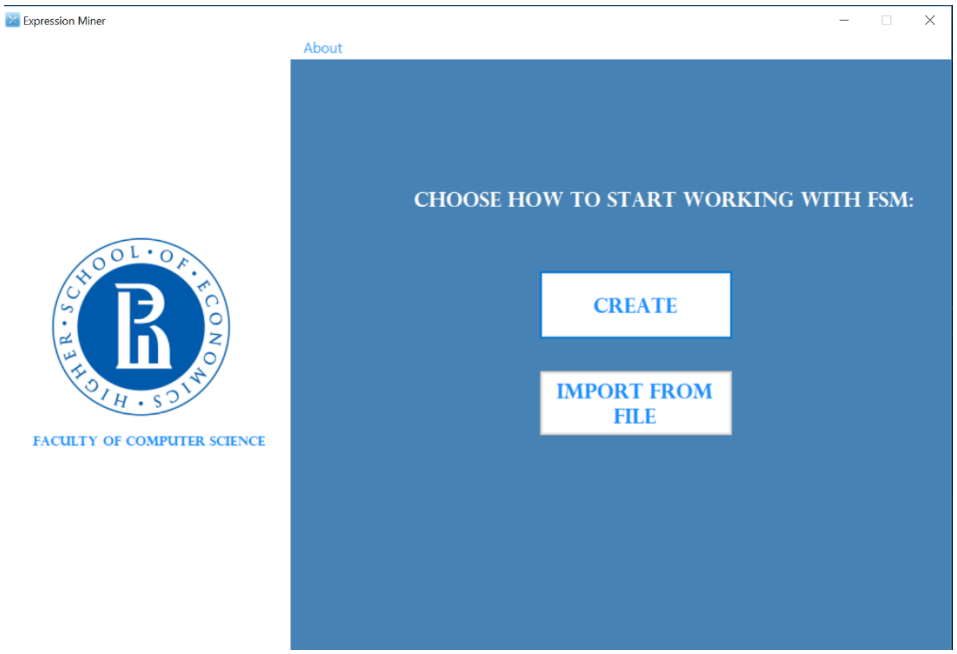


Рис. 1. Стартовое окно приложения.

6.1. Испытание выполнений требований, связанных с работой программы с автоматом, загруженным в программу в формате .tsml.

Нажимаем на кнопку «Import from file», чтобы импортировать тестовый файл формата .tsml. Выбрав необходимый файл, нажимаем «Открыть».

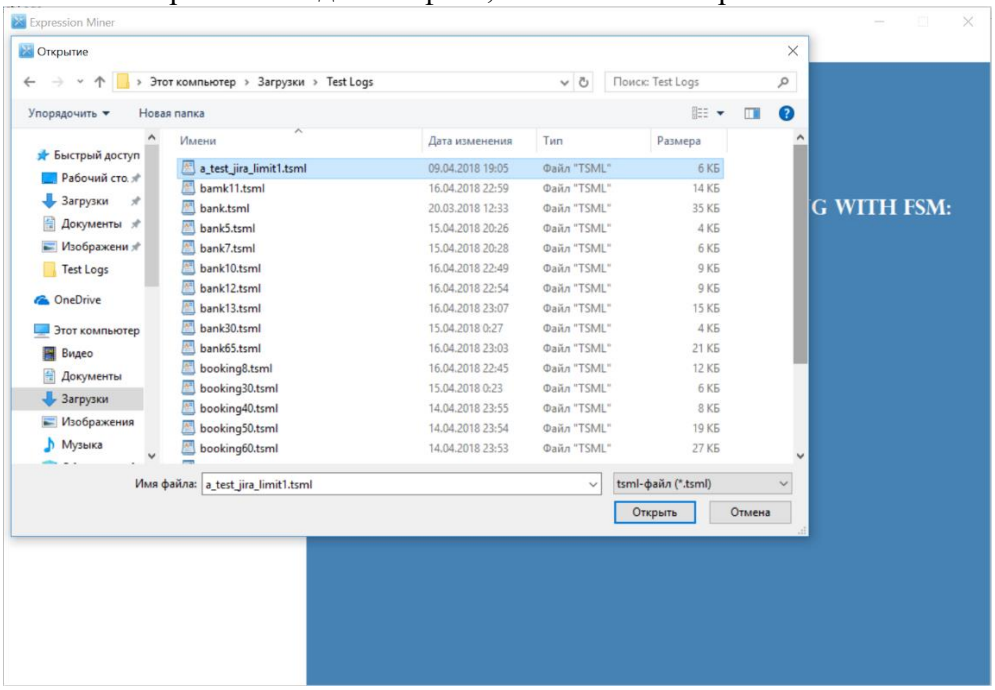


Рис. 2. Нажатие на кнопку «Import from file». Импорт в программу файла формата .tsml.

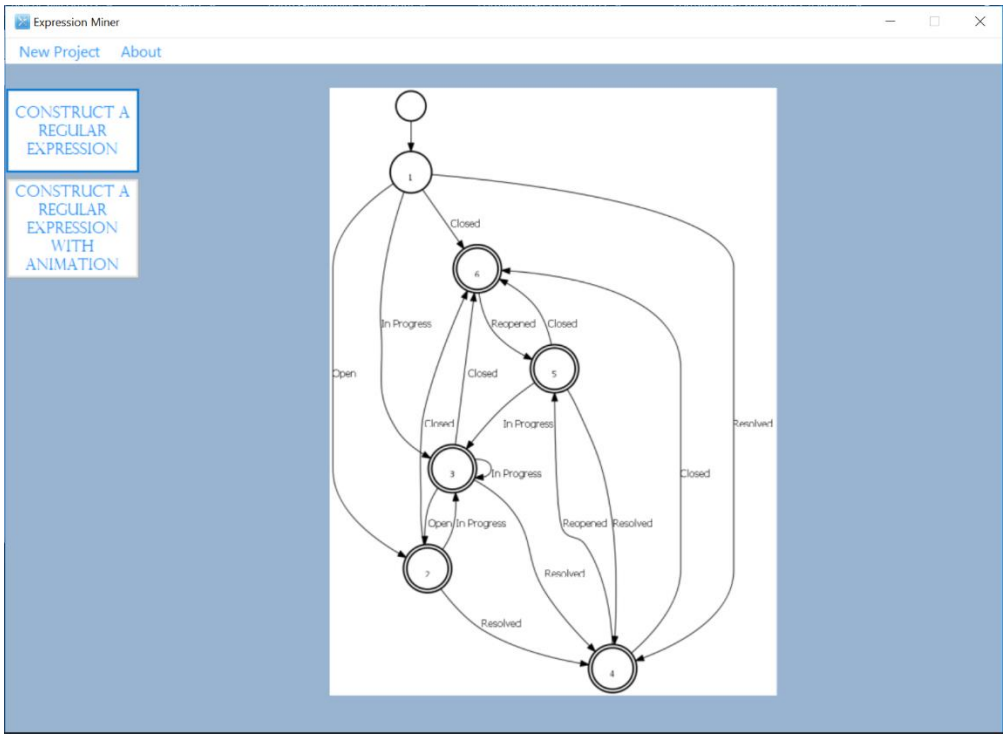


Рис. 3. Открыт файл «a_test_jira_limit1.tsml». В программу импортирован конечный автомат.

Нажимаем на кнопку «Construct a regular expression», чтобы запустить построение регулярное выражение по конечному автомату без показа анимации.

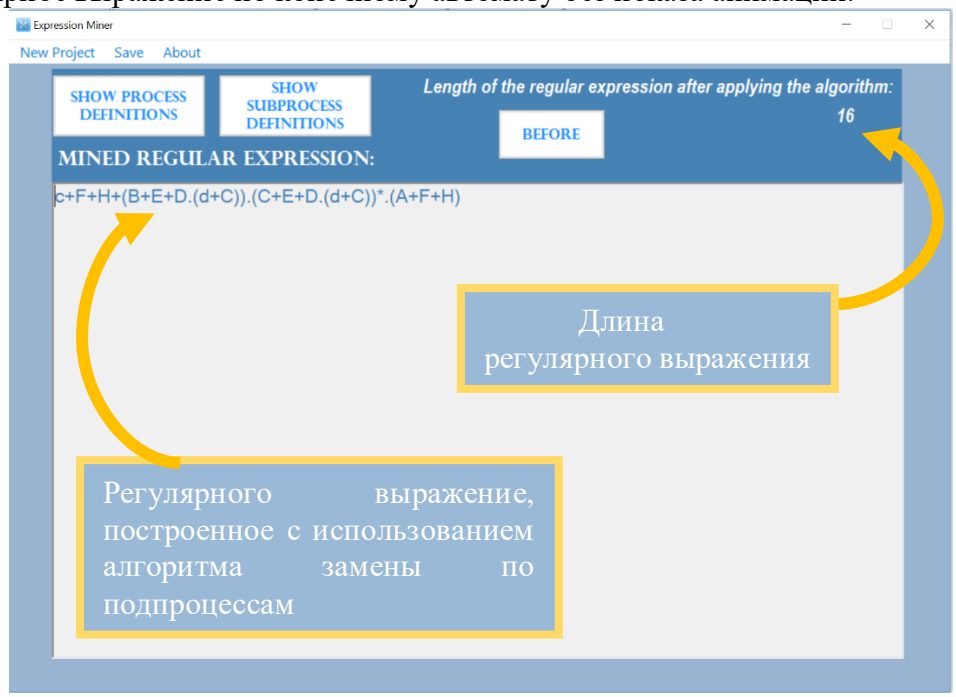


Рис. 4. Построенное по конечному автомату регулярное выражение.

Нажимаем на кнопку «Show process definitions», чтобы открыть окно со словарем процессов.

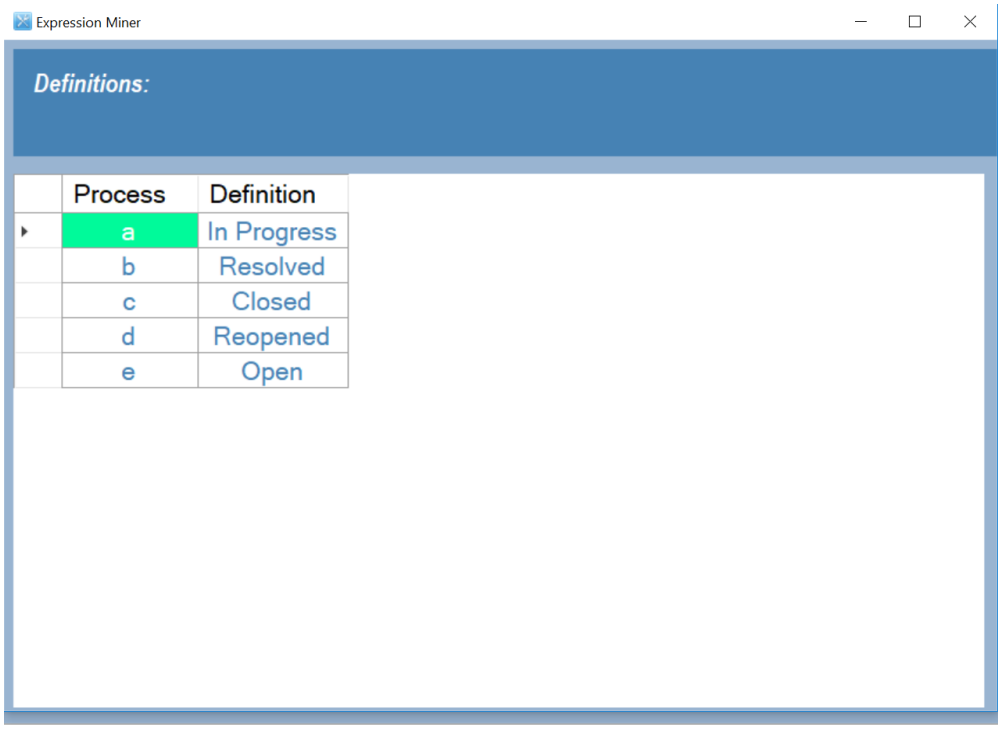


Рис. 5. Открыт словарь с определениями процессов.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.502610-01 51				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

Закрываем окно с определениями процессов. Нажимаем на кнопку «Show subprocess definitions» (окно на Рис. 4), чтобы открыть окно со словарем подпроцессов.

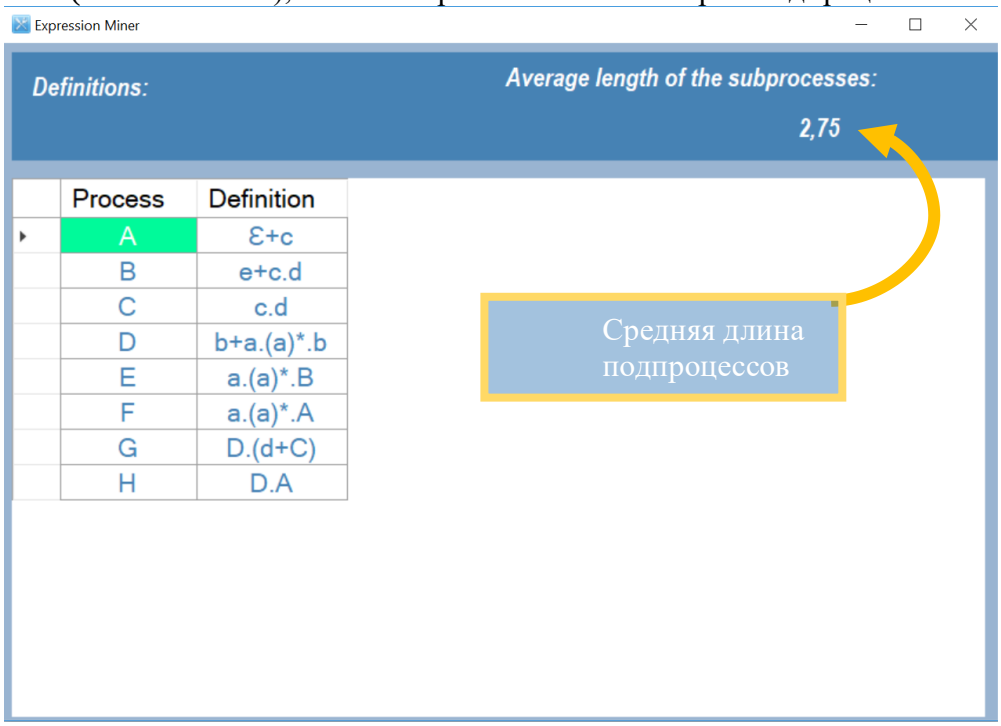


Рис. 6. Открыт словарь с определениями подпроцессов.

Закрываем окно с определениями подпроцессов. Нажимаем на кнопку «Before» (окно на Рис. 4), чтобы открыть окно с регулярным выражением, полученным без использования алгоритма замены по подпроцессам.

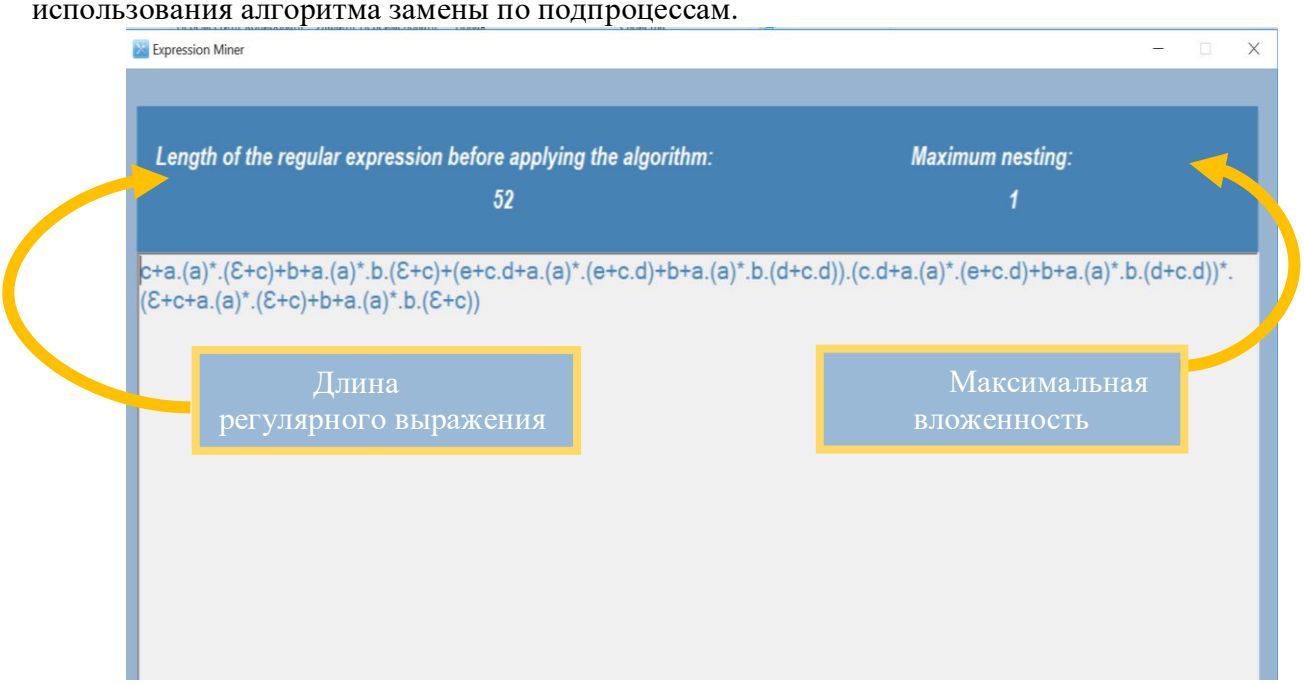


Рис. 7. Открыто окно с регулярным выражением, полученным без использования алгоритма замены по подпроцессам.

Закрываем окно с регулярным выражением, полученным без использования алгоритма замены по подпроцессам. Вызовем сохранение данных, полученных в результате работы программы. Нажмем кнопку «Save» в верхнем меню. Открылось окно с параметрами сохранения.



Рис. 8. Открыто окно с параметрами сохранения.

Сохраним все данные, полученные в результате работы программы. Нажмем кнопку «Save all».

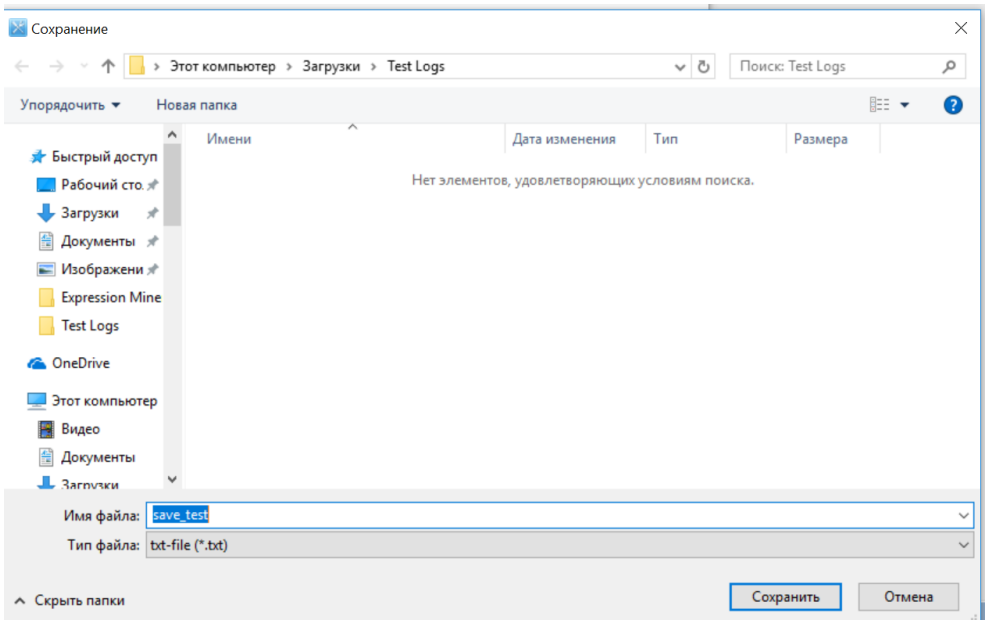


Рис. 9. Сохранение файла.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.502610-01 51				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

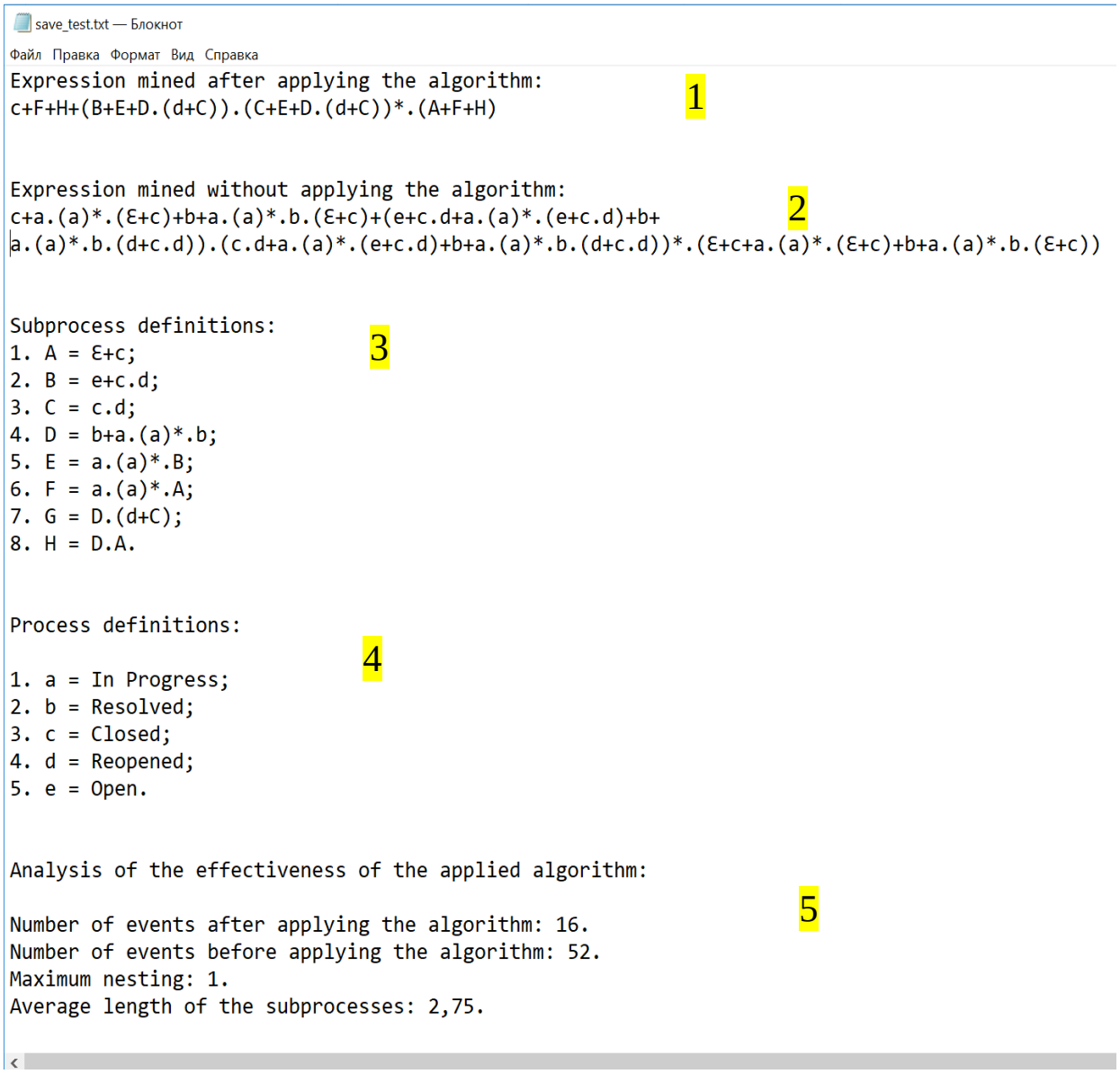


Рис. 10. Сохраненный файл.

1 – Регулярное выражение, построенное с использованием алгоритма замены по подпроцессам.

2 – Регулярное выражение, построенное без использования алгоритма замены по подпроцессам.

3 – Словарь подпроцессов.

4 – Словарь процессов.

5 – Аналитические данные.

6.2. Испытание выполнений требований, связанных с построением автомата в программе и работы с построенным автоматом.

Нажмем на кнопку «New Project» (в окне на Рис. 5), чтобы начать новый проект.

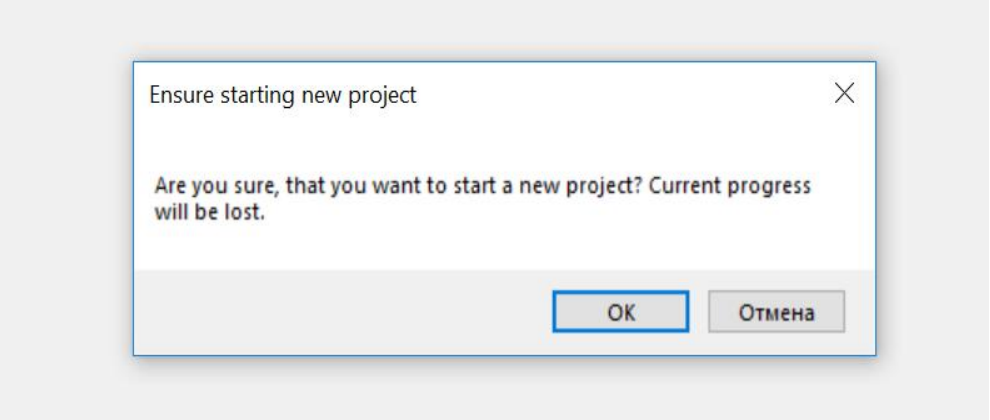


Рис. 11. Предупреждение о возможной потере данных.

Нажмем «OK». На открывшемся стартовом окне нажмем на кнопку «Create», чтобы создать в программе конечный автомат.

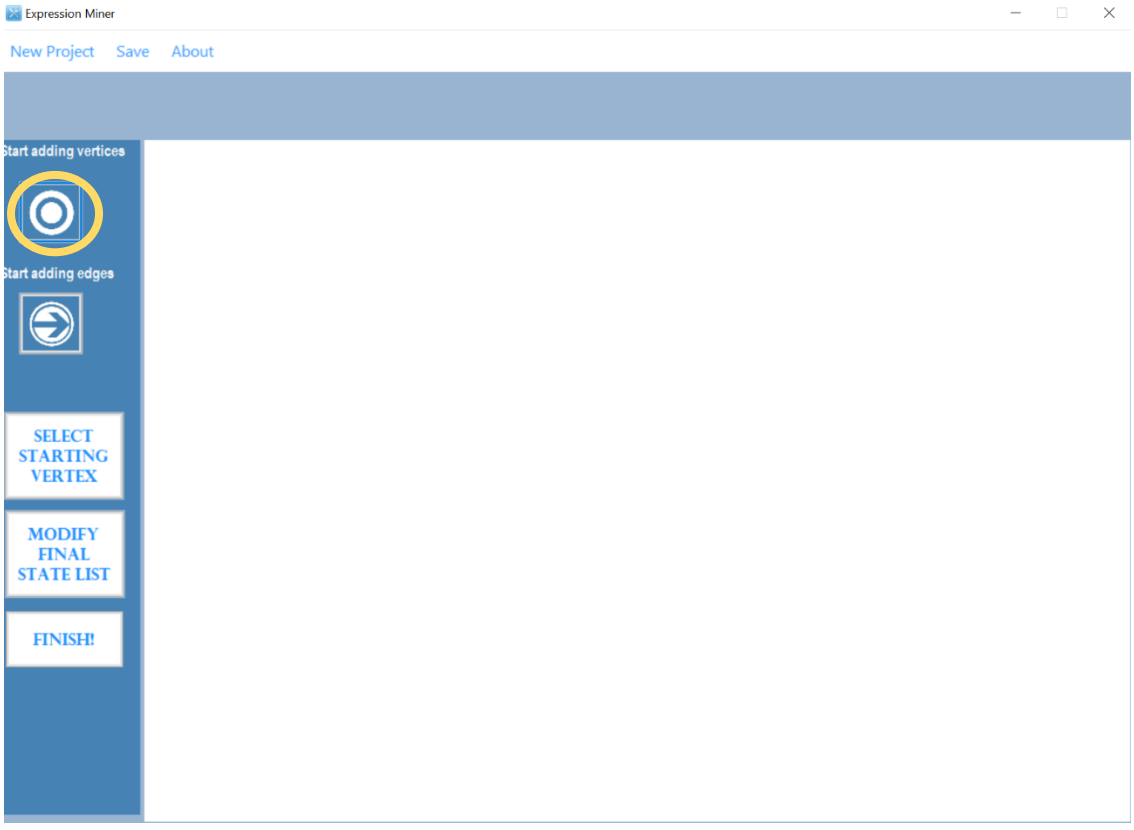


Рис. 12. Окно с инструментами для создания конечного автомата.

Нажмем на кнопку, выделенную на Рис. 12, чтобы добавить состояния автомата.

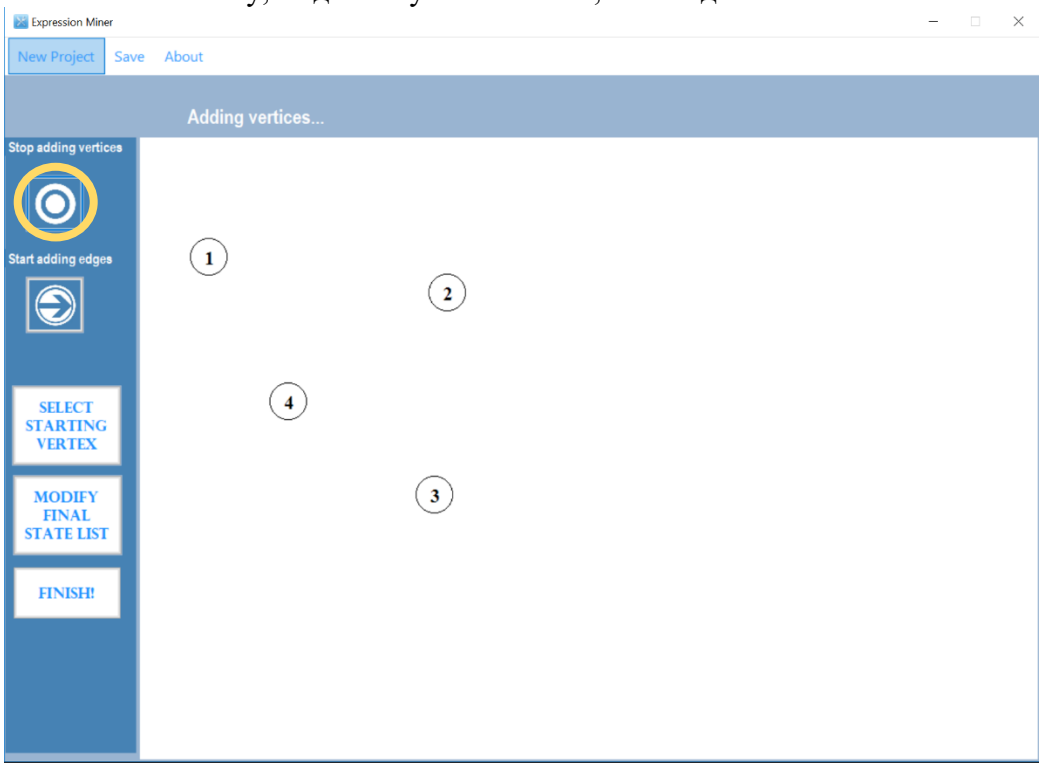


Рис. 13. Добавлены состояния автомата.

Нажмем на кнопку, выделенную на Рис. 13, чтобы завершить добавление состояний. Нажмем на состояние 4 правой кнопкой мыши, чтобы удалить состояние.

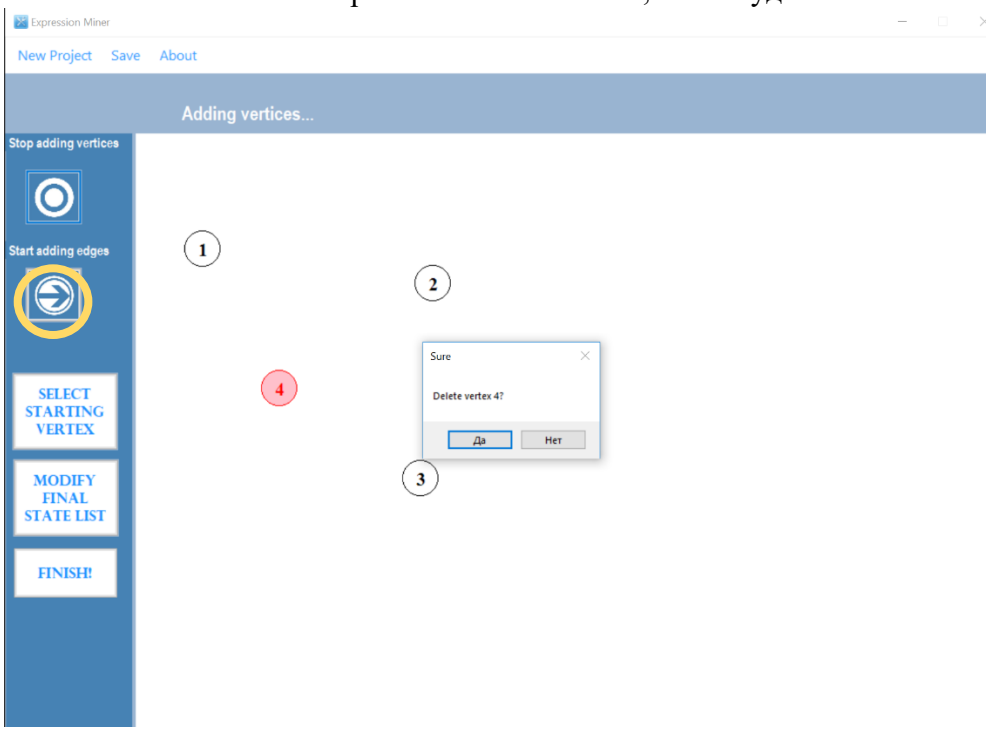


Рис. 14. Предупреждение об удалении состояния автомата.

Подтвердим удаление. Нажмем на кнопку, выделенную на Рис. 14, чтобы добавить переходы. Выберем вершину исхода (1), нажав на нее ЛКМ.

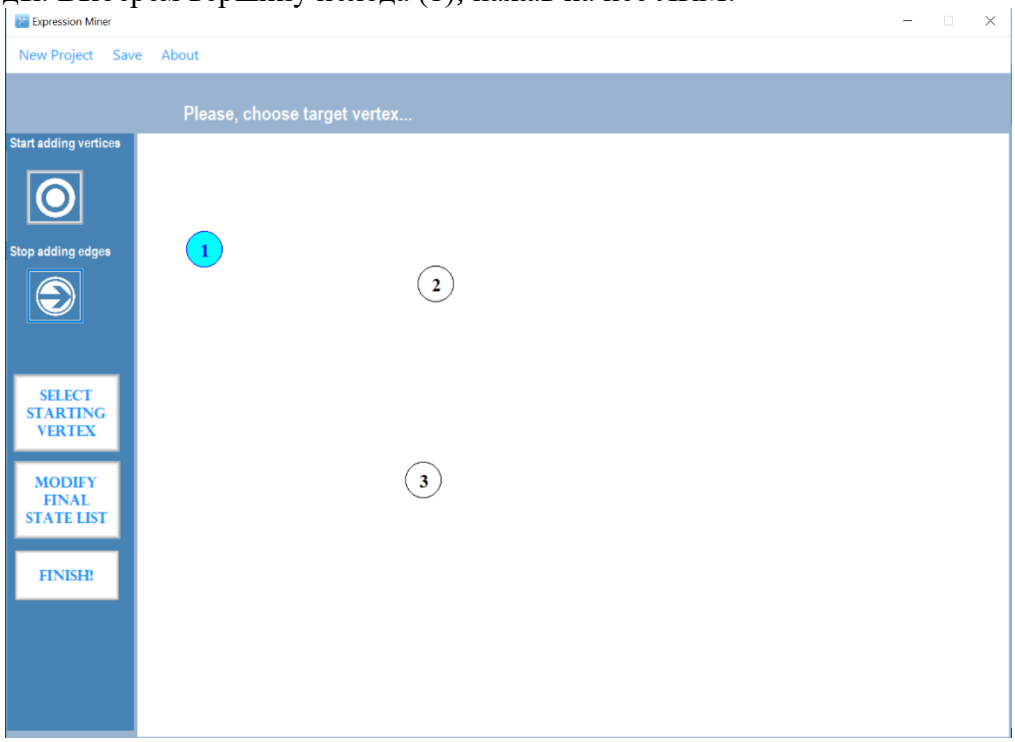


Рис. 15. Выбор вершины исхода перед добавлением перехода.

Выберем вершину входа (2) и введем регулярное выражение, задающее переход.

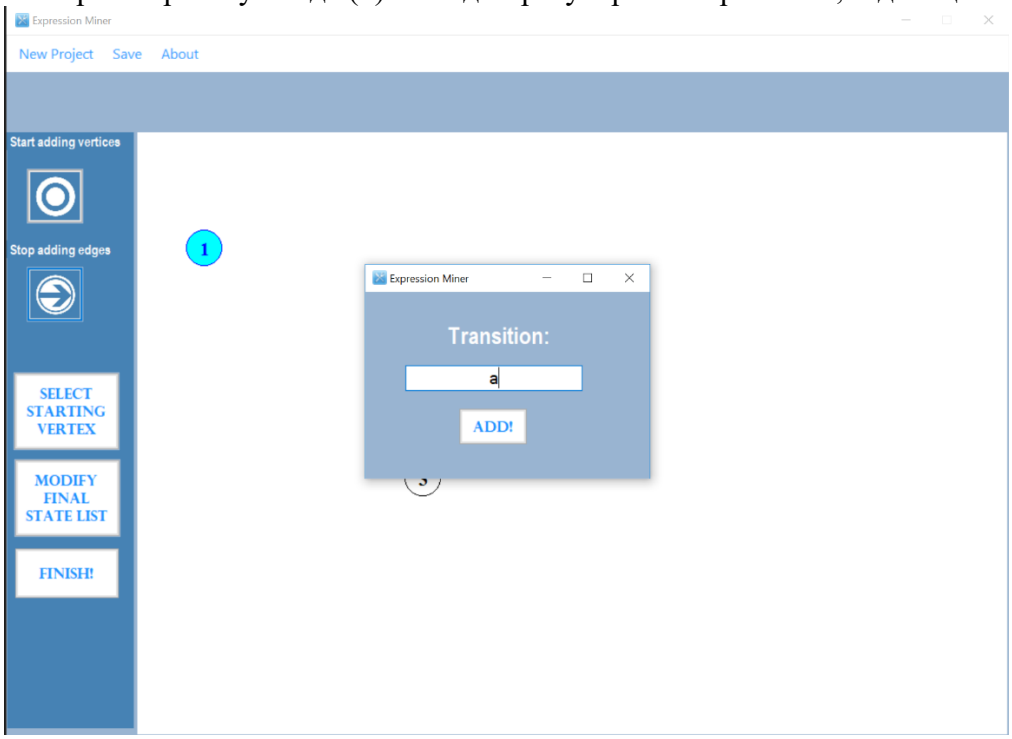


Рис. 16. Ввод регулярного выражения, задающего переход («a»).

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.502610-01 51				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

Нажмем кнопку «Add», чтобы завершить добавление перехода.

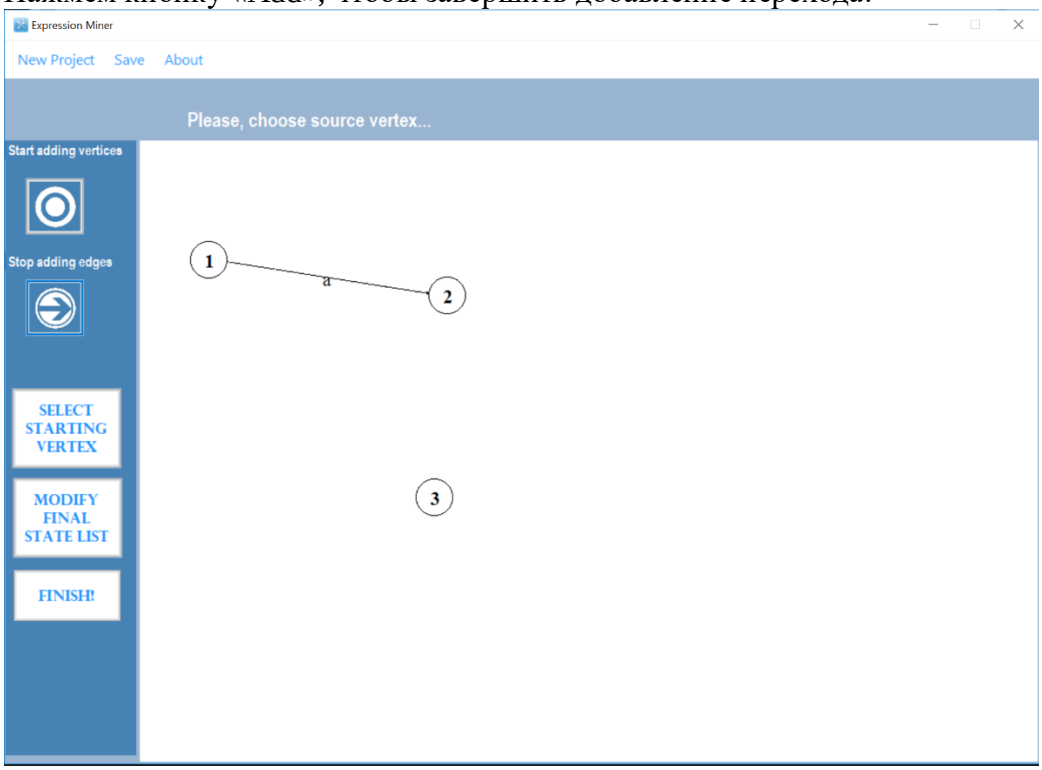


Рис. 17. Добавлен переход «a».

Таким же образом добавим еще несколько переходов.

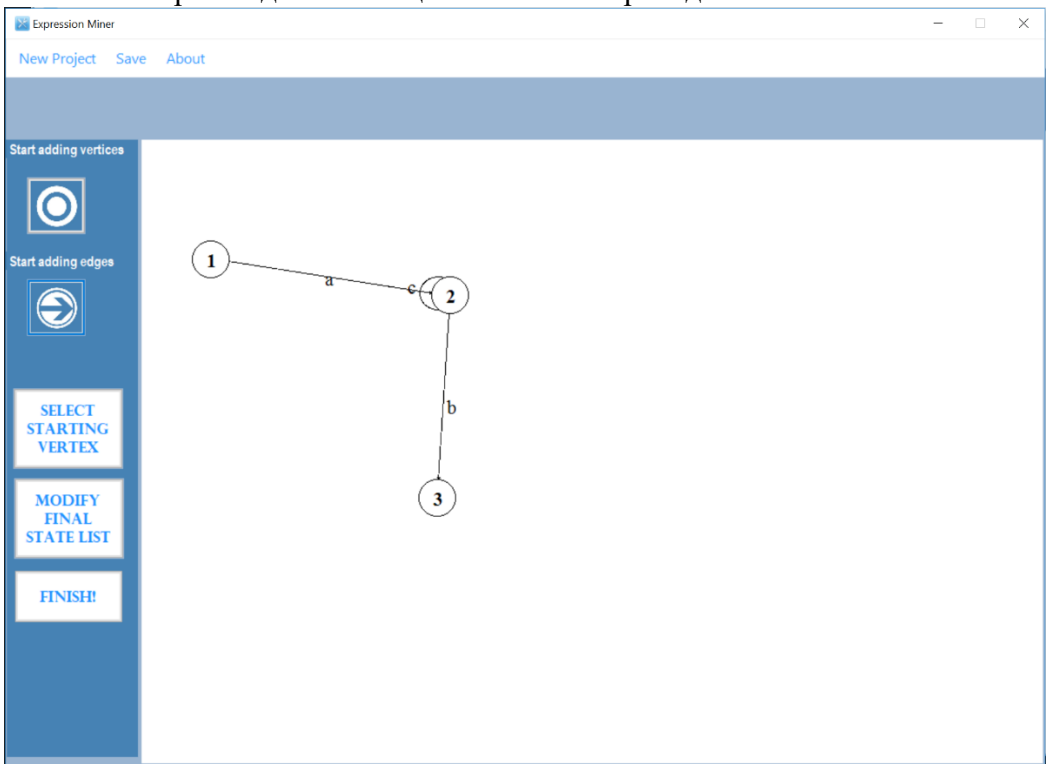


Рис. 18. Добавлены переходы «b» (2-3) и «c» (2-2).

Нажмем кнопку «Select starting vertex», чтобы добавить начальное состояние. Нажмем на состояние 1, чтобы выбрать его начальным.

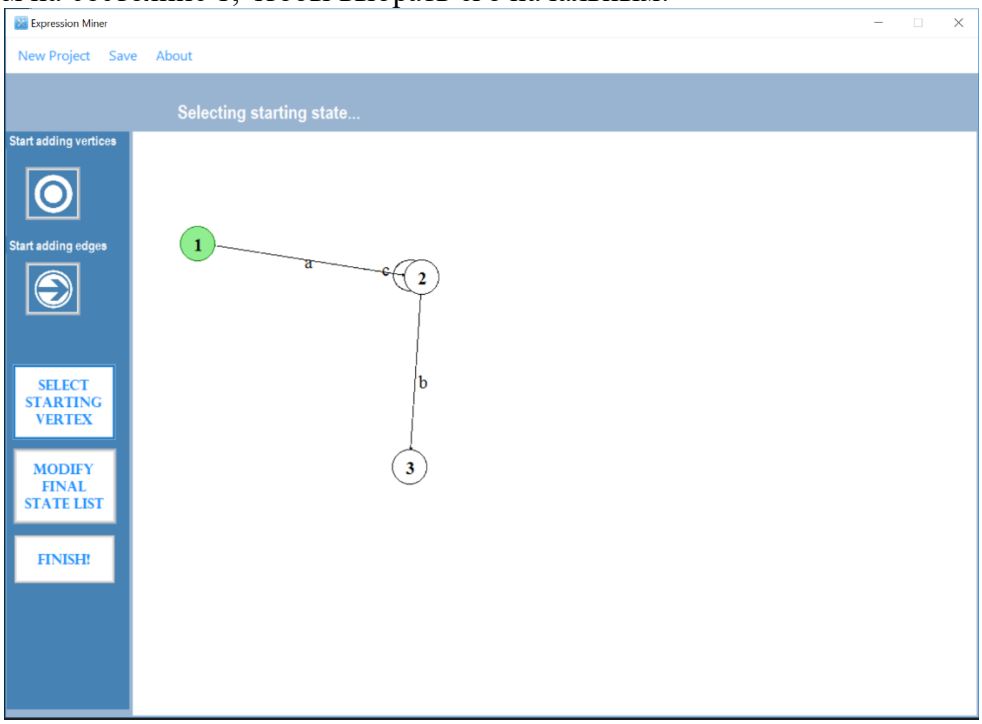


Рис. 19. Состояние 1 выбрано начальным.

Нажмем на кнопку «Modify final state list», чтобы изменить список допускающих состояний. Выберем состояние 3 допускающим.

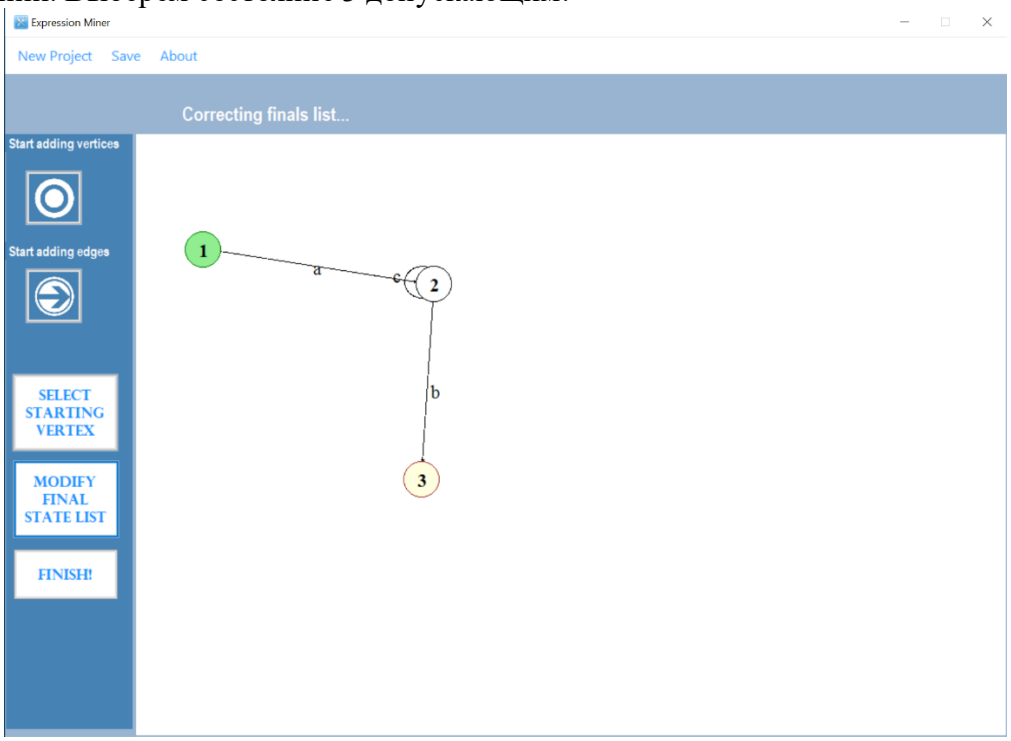


Рис. 20. Состояние 3 выбрано допускающим.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.502610-01 51				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

Сохраним построенный автомат в формате .tsml. Для этого нажмем на кнопку «Save» верхнего меню.

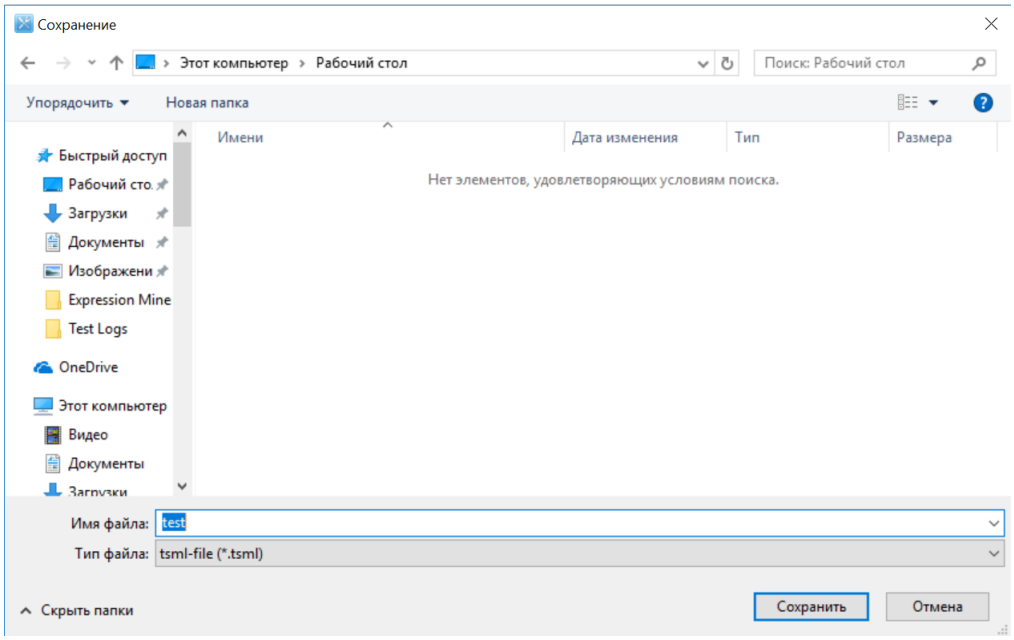


Рис. 21. Сохранение построенного в программе автомата.

Нажимаем «Сохранить».

```
<?xml version="1.0" encoding="utf-8"?>
<graph>
  <state id="state1">
    <name>2</name>
  </state>
  <state id="state2" accept="true">
    <name>3</name>
  </state>
  <state id="state3" start="true">
    <name>1</name>
  </state>
  <transition id="transition1" source="state3"
target="state1">
    <name>a</name>
  </transition>
  <transition id="transition2" source="state1"
target="state2">
    <name>b</name>
  </transition>
  <transition id="transition3" source="state1"
target="state1">
    <name>c</name>
  </transition>
</graph>
```

Рис. 22. Сохраненный файл «test.tsml».

Завершим построение автомата, нажав кнопку «Finish!».

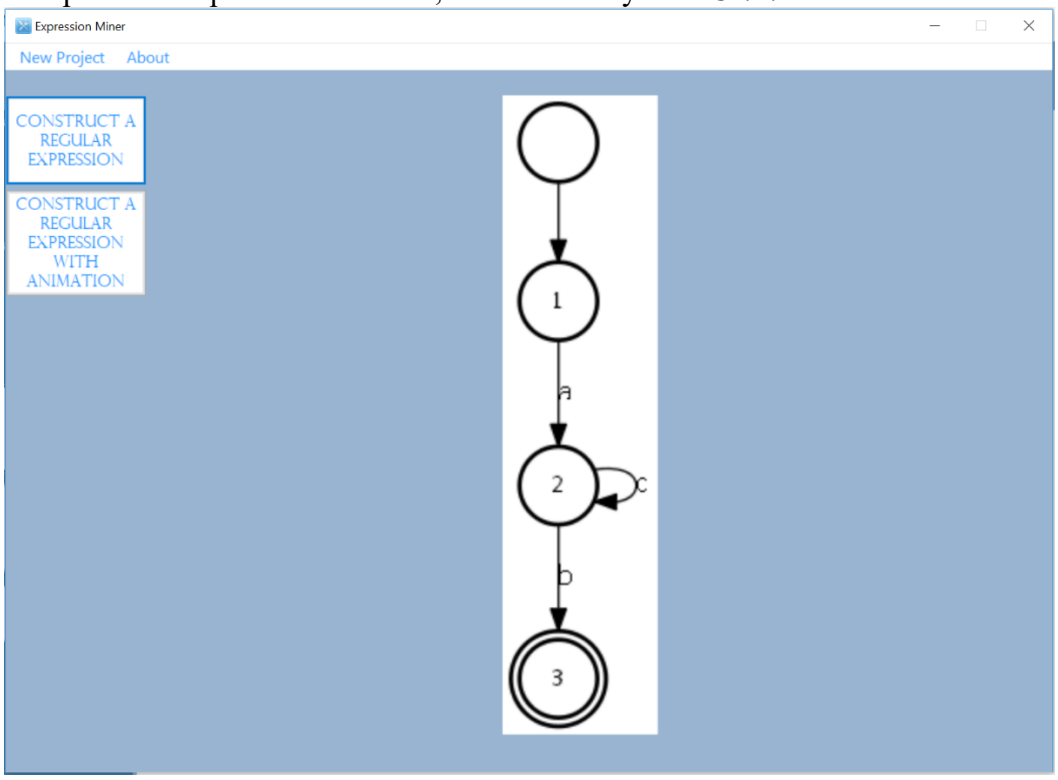


Рис. 23. Программа готова к работе с построенным в ней автоматом.

Запустим построение регулярного выражения из конечного автомата с показом анимации. Для этого нажмем кнопку «Construct a regular expression with animation».

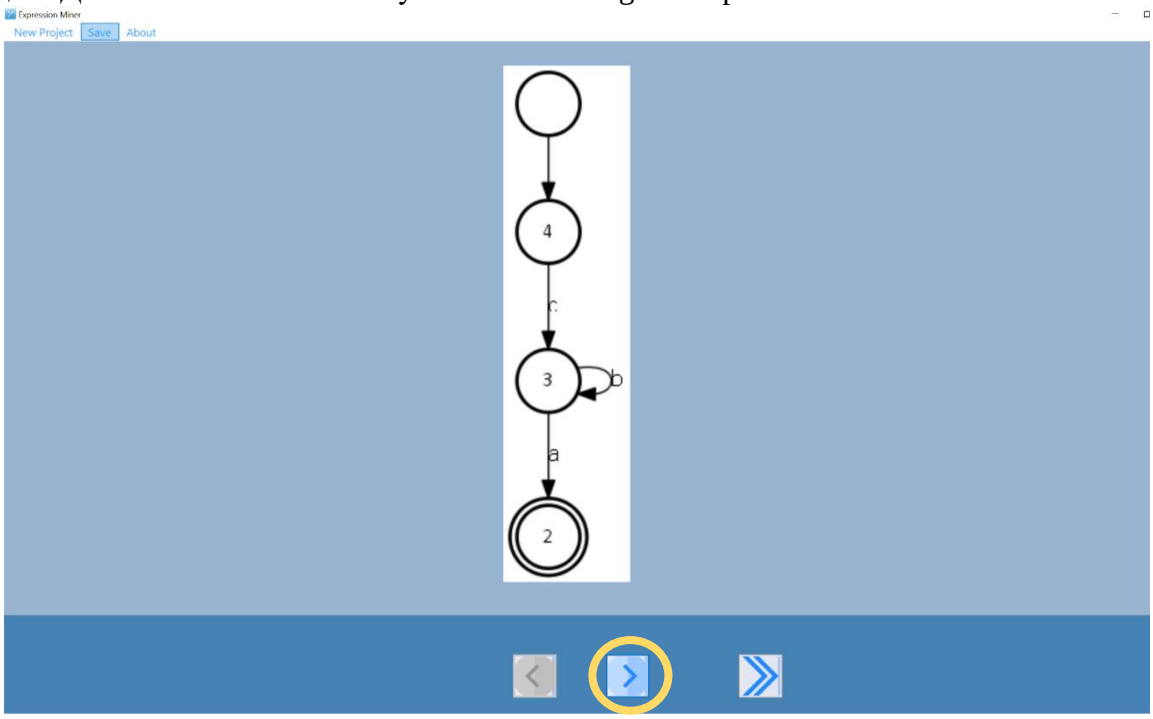


Рис. 24. Окно анимации.

Нажмем на кнопку, выделенную на Рис. 24, увидеть следующий шаг анимации.

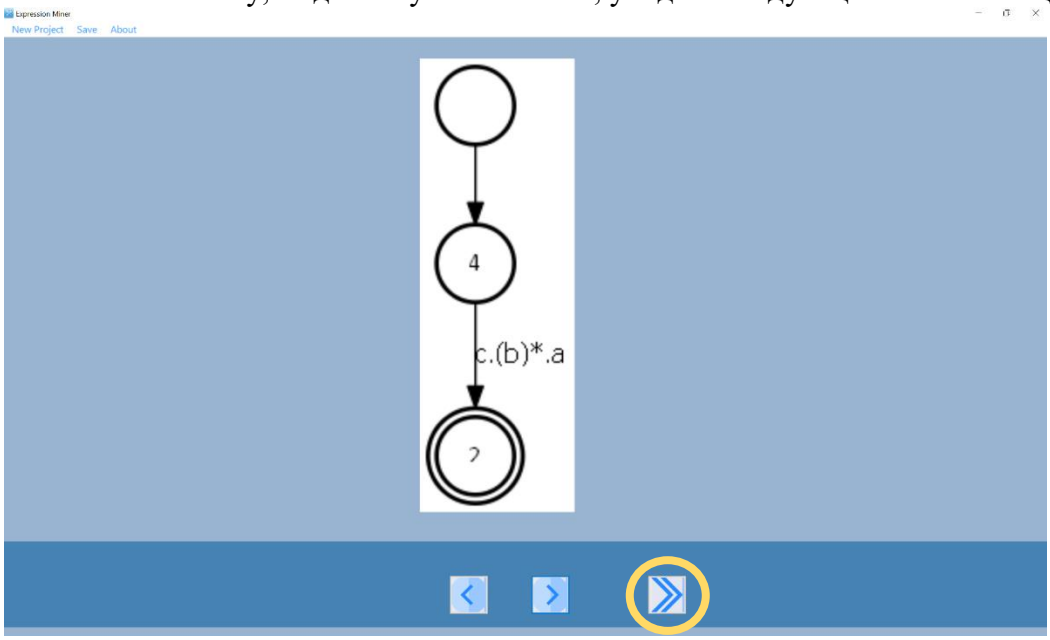


Рис. 25. Второй шаг анимации.

Нажмем на кнопку «Save» верхнего меню, чтобы сохранить текущий шаг анимации.

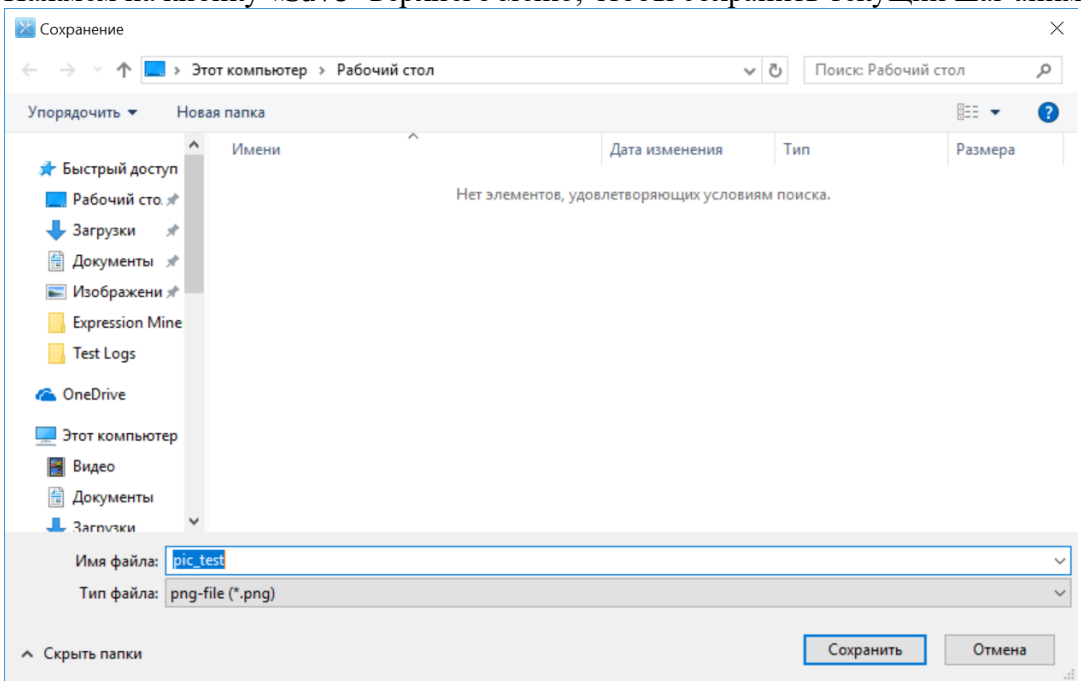


Рис. 26. Сохранение текущего шага анимации.

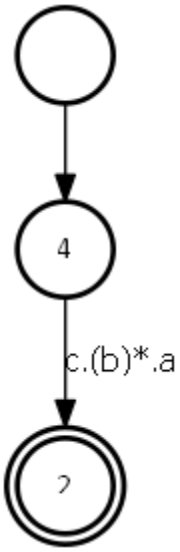


Рис. 27. Сохраненный файл «pic_test.png».

Нажмем на кнопку, выделенную на Рис. 25, чтобы пропустить все дальнейшие шаги анимации.

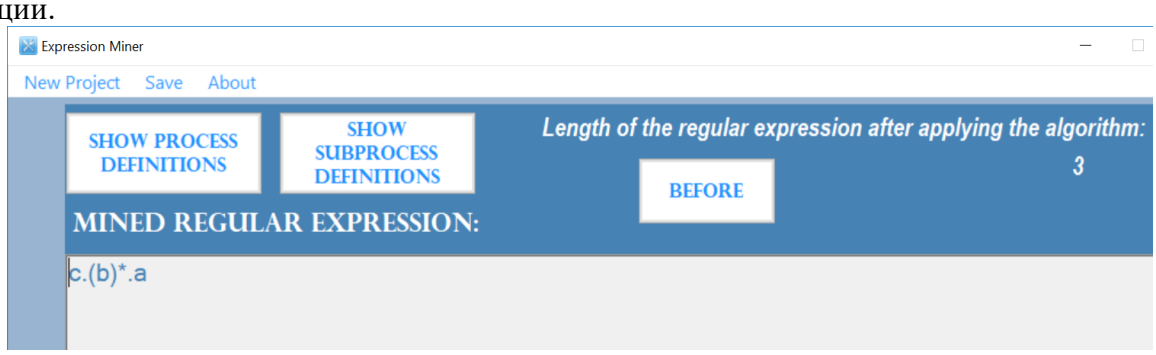


Рис. 28. Построенное регулярное выражение.

Нажмем на кнопку «About», чтобы увидеть информацию о разработчике.

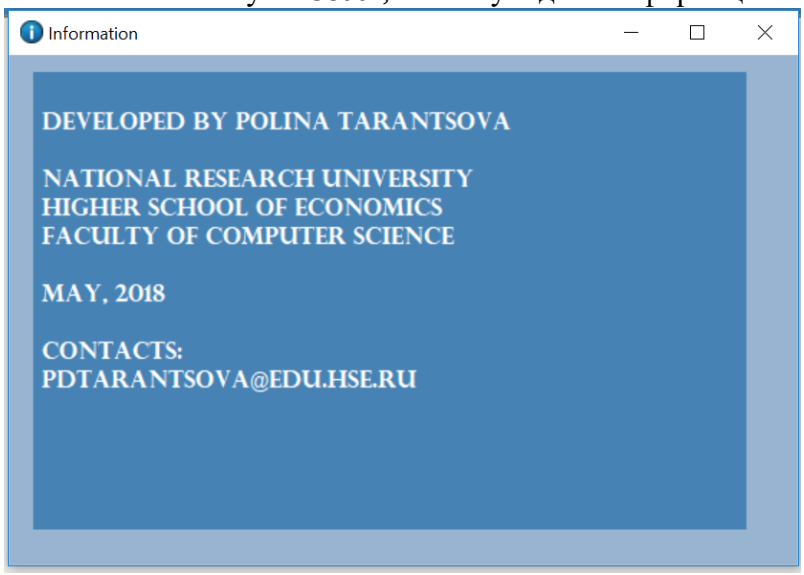


Рис. 29. Построенное регулярное выражение.

Проверка корректности работы программы завершена. Все испытания пройдены успешно.

ПРИЛОЖЕНИЕ 1

ТЕРМИНОЛОГИЯ

Таблица 1.1

Термин (рус)	Термин (англ)	Определение
Глубинный анализ процессов	**Process mining**	Область исследований, занимающаяся построением моделей процессов по логам событий.
Способ представления	**Representational bias**	Набор ограничений, налагаемых на модели целевых процессов, играет важную роль глубинном анализе процессов.
Регулярные выражения	**Regular expressions**	Способ моделирования последовательных процессов, использующий шаблоны выбора, повторения и итерации.
Метод удаления состояний	**State removal method**	Алгоритм построения регулярного выражения по конечному автомату, предполагающий последовательное удаление состояний автомата и одновременное построение регулярных выражений, соответствующих заменяемым системам переходов.
Язык	**Language**	Множество последовательностей действий.
Регулярный язык	**Regular language**	Регулярный язык – это язык на множестве активностей $\mathcal{A}$, если он может быть построен индуктивно, опираясь на следующие правила: 1. $\emptyset$ - это регулярный язык; 2. $\{\langle\rangle\}$ – это регулярный язык; 3. $\{\langle a_i\rangle\}$, где $a_i \in \mathcal{A}$ – это регулярный язык, обозначаемый a_i; 4. Пусть $P, Q \subseteq \mathcal{A}^*$ - регулярные выражения, тогда: 4.1. $P.Q = \{\langle p_1, \dots, p_n, q_1, \dots, q_m\rangle \mid \langle p_1, \dots, p_n\rangle \in P \mid \langle q_1, \dots, q_m\rangle \in Q\}$ – также является регулярным выражением; 4.2. $P \mid Q = \{\sigma \mid \sigma \in P \cup Q\}$ - также является регулярным выражением; 4.3. $P^* = U_{i=0}^{\infty} P^n$, где $P^0 = \{\langle\rangle\}, P^1 = P$, и $P^n = P^{n-1}.P$ при $n > 0$ - также является регулярным выражением.

ТЕРМИНОЛОГИЯ

Таблица 2.2

Термин (рус)	Термин (англ)	Определение
Конечный автомат	**Finite state machine (FSM)**	Конечный автомат – это кортеж $FSM = (S, E, T, s_i, S_f)$, где S – конечное множество состояний, E– конечное множество событий, $T \subseteq (S \times E \times S)$ – конечное множество переходов, $s_i \subseteq S$ – начальное состояние, и $S_f \subseteq S$ – множество конечных состояний.
Начальное состояние	**Initial state**	Стартовое состояние автомата.
Конечные (допускающие) состояния	**Accepting states**	Состояния, в которых автомат может завершить работу.
Событие (Процесс)	**Event**	Действие, способное изменить состояние автомата.
Переход	**Transition**	Событие, связывающее два состояния автомата.
Длина регулярного выражений	**Length of the regular expression**	Количество символов без учета ключевых знаков.
Максимальная вложенность	**Maximum nesting**	Максимальное количество замен символов переходов, с помощью которых были получены символы из алфавита.
Подпроцесс	**Subprocess**	Регулярное выражение, которое является частью регулярного выражения, задающего автомат, и может быть заменено одним символом.
Длина подпроцесса	**Length of the subprocess**	Длина регулярного выражения, задающего подпроцесс.
Алгоритм минимизации конечного автомата	**Minimization algorithm**	Алгоритм, позволяющий получить автомат, эквивалентный исходному, с наименьшим возможным числом состояний.
Метод построения регулярного выражения из конечного автомата с заменой подпроцессов	**Method of constructing regular expression out of an FSM using subprocess changing**	Метод построения регулярного выражения из конечного автомата, при котором переходы отождествляются с регулярными выражениями, повторяющиеся регулярные выражения именуются подпроцессами и заменяются ключевыми символами.
Префикс	**Prefix**	Подстрока, с которой начинается строка символов.

ПРИЛОЖЕНИЕ 2

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

1. ГОСТ 19.101-77 Виды программ и программных документов. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
2. ГОСТ 19.102-77 Стадии разработки. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
3. ГОСТ 19.103-77 Обозначения программ и программных документов. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
4. ГОСТ 19.104-78 Основные надписи. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
5. ГОСТ 19.105-78 Общие требования к программным документам. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
6. ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
7. ГОСТ 19.201-78 Техническое задание. Требования к содержанию и оформлению. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
8. ГОСТ 19.603-78 Общие правила внесения изменений. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
9. ГОСТ 19.604-78 Правила внесения изменений в программные документы, выполненные печатным способом. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
10. ГОСТ 15150-69 Машины, приборы и другие технические изделия. Исполнения для различных климатических районов. Категории, условия эксплуатации, хранения и транспортирования в части воздействия климатических факторов внешней среды. – М.: Изд-во стандартов, 1997.
11. ГОСТ 19.301-79 Программа и методика испытаний. Требования к содержанию и оформлению. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
12. Hopcroft, J. Introduction to Automata Theory, Languages, and Computation / John E. Hopcroft, Jeffrey D. Ullman – 3rd edition – USA: Addison-Wesley, 1979 – 358 p.
13. Neumann, C. Converting Deterministic Finite Automata to Regular Expressions [Электронный ресурс] – Режим доступа: http://liacs.leidenuniv.nl/~bonsanguemm/FI2/DFA_to_RE.pdf, свободный. (дата обращения: 24.02.18)
14. van der Aalst, W.M.P. Process Mining: Data Science in Action / Wil van der Aalst- 2nd edition – Heidelberg, Germany: Springer, 2016.
15. Leemans, S. Discovering Block-Structured Process Models from Event Logs - A Constructive Approach/ Leemans, S., Fahland, D., van der Aalst, W// Application and Theory of Petri Nets and Concurrency/ Colom, J., Desel, J., eds.// 34th International Conference, PETRI NETS 2013 - Milan, Italy// Volume 7927 of Lecture Notes in Computer Science - Berlin, Heidelberg, Germany: Springer, 2013 – p. 311–329.
16. Leemans, S. Discovering Block-Structured Process Models from Incomplete Event Logs/ Leemans, S., Fahland, D., van der Aalst, W// Application and Theory of Petri Nets and

Concurrency/ Ciardo, G., Kindler, E., eds.// 35th International Conference, PETRI NETS 2014 - Tunis, Tunisia// Volume 8489 of Lecture Notes in Computer Science. Springer International Publishing – Cham, Germany: 2014- p. 91–110.

17. van der Aalst, W.M.P. Process mining: a two-step approach to balance between underfitting and overfitting/ van der Aalst, W.M.P., Rubin, V., Verbeek, H.M.W., van Dongen, B.F., Kindler, E., Günther, C.W.// Software & Systems Modeling 9(1) – 2008 – p. 87.
18. Kleene, S.C. Representation of events in nerve nets and finite automata/ Kleene, S.C// Automata Studies/ Shannon, C., McCarthy, J., eds - Princeton, NJ, USA: Princeton University Press, 1956 – p. 3–4.1
19. Brzozowski, J.A. Derivatives of regular expressions/ Brzozowski, J.A.// J. ACM 11(4) - October 1964 – p. 481–494.
20. Linz, P. An Introduction to Formal Language and Automata/Peter Linz – USA: Jones and Bartlett Publishers, Inc., 2006.
21. Delgado, M. Approximation to the smallest regular expression for a given regular language// Delgado, M., Morais, J.// Implementation and Application of Automata/ Domaratzki, M., Okhotin, A., Salomaa, K., Yu, S., eds. - Berlin, Heidelberg: Springer Berlin Heidelberg, 2005 -p. 312–314.
22. van Dongen, B.F. The prom framework: A new era in process mining tool support/ van Dongen, B.F., de Medeiros, A.K.A., Verbeek, H.M.W., Weijters, A.J.M.M., van der Aalst, W.M.P.// Applications and Theory of Petri Nets 2005/ Ciardo, G., Darondeau, P., eds. - Berlin, Heidelberg: Springer Berlin Heidelberg, 2005 – p. 444–454.

ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

Лист регистрации изменений									
Номера листов (страниц)					Всего листов (страниц в докум.)	№ документа	Входящий № сопроводительного докум. и дата	Подп.	Дата
Изм.	Измененных	Заменен ных	Новых	Аннулированных					

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.502610-01 51				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата