



# Теория Конечных Автоматов

Лабораторный практикум



Антон Николаевич Гайдук

[gaiduk-edu.github.io/course-fsm/](https://gaiduk-edu.github.io/course-fsm/)

2026

# Лабораторная работа №1 Знакомство с JFLAP

В данной лабораторной работе вы познакомитесь и начнете использовать инструмент моделирования автоматов JFLAP (Java Formal Language and Automata Package), программное обеспечение на основе Java разработанное исследовательской группой университета Дьюка (Duke University).

**Задание 1.1** Скачать программное обеспечение JFLAP7.1.jar

<https://www.jflap.org/jflaptmp/>

Чтобы запустить новый КА, запустите JFLAP и выберите в меню опцию "автомат Мили"(Mealy Machine).

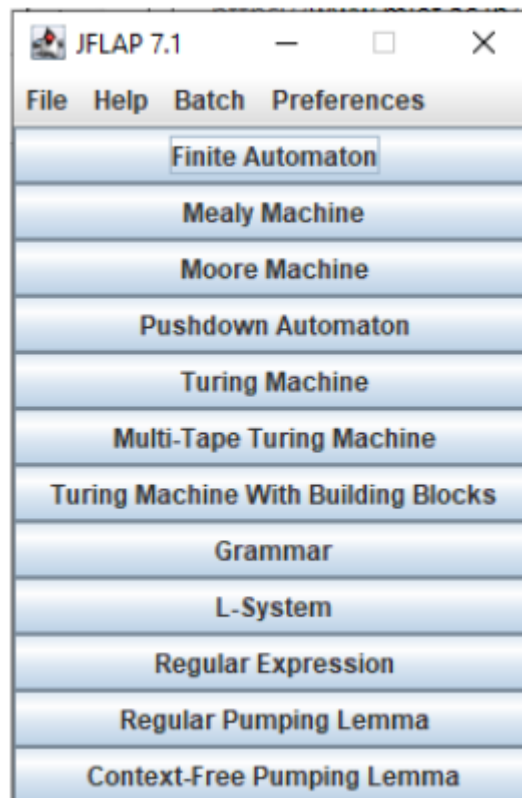


Рисунок 1 – Запуск JFLAP

При этом должно появиться новое окно, которое позволит вам создавать

и редактировать КА. Редактор разделен на две основные области: холст, на котором вы можете создать свой автомат, и панель инструментов, которая содержит инструменты, необходимые для создания вашего автомата.

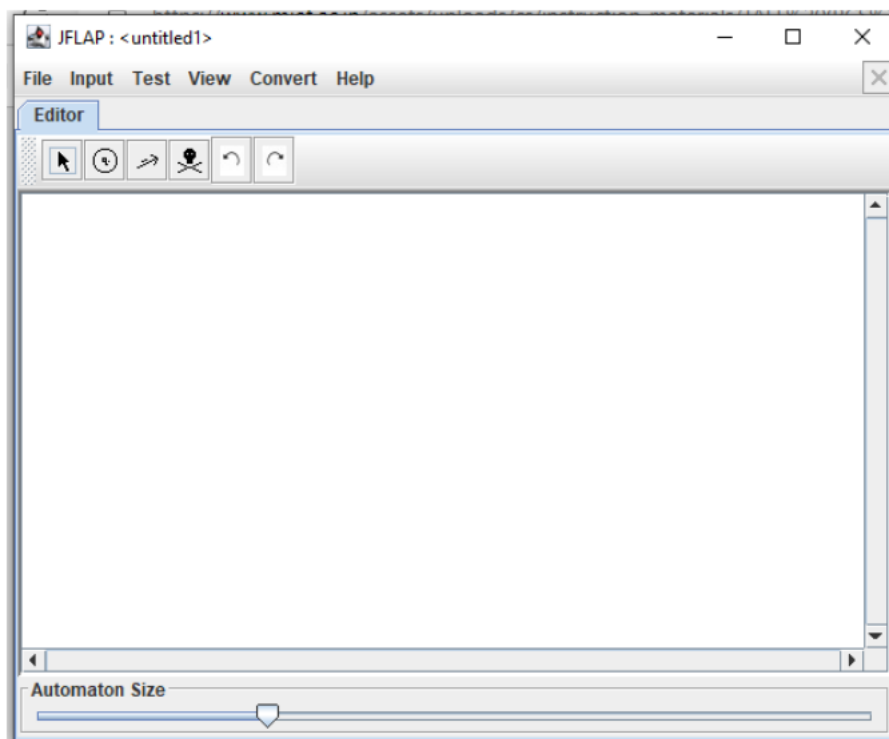


Рисунок 2 – Окно редактора JFLAP

Давайте подробнее рассмотрим панель инструментов.

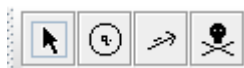






Рисунок 3 – Панель инструментов JFLAP

Как вы можете видеть, панель инструментов содержит четыре инструмента:

- Инструмент редактирования атрибутов : устанавливает начальное и конечное состояния;
- Инструмент создания состояний : создает состояния;
- Инструмент создания переходов : создает переходы;
- Инструмент удаления : удаляет состояния и переходы.

Чтобы выбрать инструмент, щелкните мышью на соответствующую икон-

ку. Когда инструмент выбран, он затенен, как, например, инструмент **Редактор атрибутов**. Выбор инструмента переводит его в соответствующий режим. Например, с панелью инструментов, показанной выше, мы находимся в режиме “Редактор атрибутов”. Различные режимы определяют, каким образом щелчки мыши влияют на работу машины. Например, если мы находимся в режиме “Создание состояний”, то щелчок на холсте приводит к созданию новых состояний.


Теперь давайте приступим к созданию нашего КА. Наш конечный автомат будет реализовать D-тригер (элемент задержки).

**Элемент задержки.** Вход: 0 - отсутствие сигнала, 1 - наличие сигнала. Выход: 0 или 1. Множество входов  $A = \{0, 1\}$ . Множество состояний  $Q = \{q_0, q_1\}$ . Множество выходов  $B = \{0, 1\}$ . В состояние  $q_0$  элемент задержки переходит, если на вход поступил символ 0. В состояние  $q_1$  элемент задержки переходит, если на вход поступил символ 1. Функции  $\phi, \psi$  задаются с помощью следующей таблицы

Входной символ	Функция переходов $\varphi$		Функция выходов $\psi$	
	$q_0$	$q_1$	$q_0$	$q_1$
0	$q_0$	$q_0$	0	1
1	$q_1$	$q_1$	0	1

**Задание 1.2** Используя программное обеспечение JFLAP7.1.jar задать модель конечного автомата D-тригера.

## Создание состояний

В первую очередь, создадим несколько состояний. Для этого нам нужно активировать инструмент **Создание состояний**, нажав  на панели инструментов. Затем щёлкнуть по рабочему полю в различных местах для создания состояний. Здесь создано 2 состояния. Окно вашего редактора

должно выглядеть примерно так:

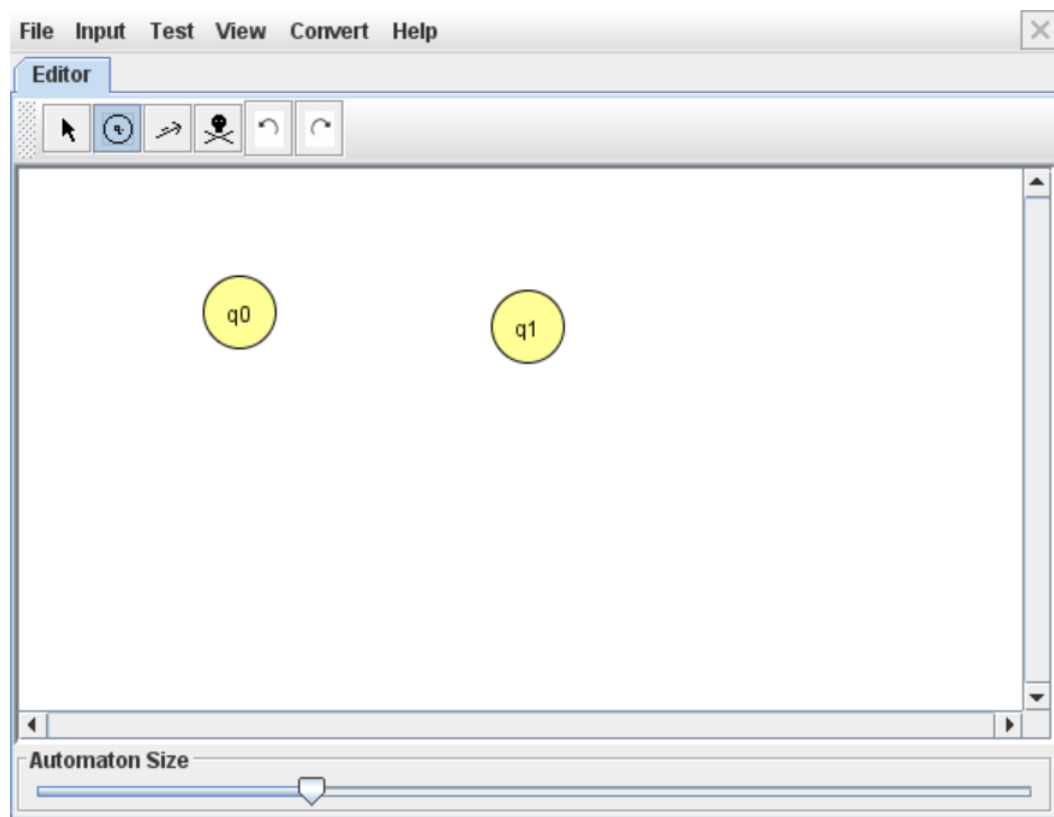



Рисунок 4 – Создание состояний

Теперь, когда мы создали наши состояния, определим начальное состояние.

## Определение начального состояния

Выбираем  $q_0$  нашим начальным состоянием. Чтобы определить  $q_0$  как начальное состояние сначала выберем инструмент Редактор атрибутов  на панели инструментов. Теперь, когда мы находимся в режиме редактора атрибутов, щелкните правой кнопкой мыши на  $q_0$ . Появится всплывающее меню, которое выглядит следующим образом:

Во всплывающем меню установите флажок **Initial**. Слева от  $q_0$  появится белая стрелка, указывающая на то, что это начальное состояние.

Определив начальное состояние, давайте перейдем к созданию переходов.



Рисунок 5 – Меню состояний

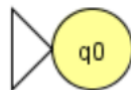



Рисунок 6 –  $q_0$  определен как начальное состояние.

## Создание переходов

Мы знаем, что входные последовательности в нашем языке могут начинаться с 0, поэтому исходное состояние должно иметь исходящий переход для входного символа 0. Мы также знаем, что входная последовательность может начинаться с любого количества нулей, а это означает, что конечный автомат должен находиться в том же состоянии после обработки любого количества нулей. Таким образом, исходящий переход для входного символа 0 из  $q_0$  возвращается сам в себя.

Чтобы создать такой переход, сначала выберите инструмент **Создание переходов**  на панели инструментов. Затем нажмите  $q_0$  на рабочем поле. Над состоянием должно появиться такое текстовое поле:

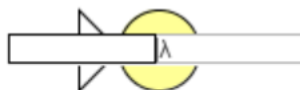


Рисунок 7 – Создание перехода

Обратите внимание, что  $\lambda$ , представляющая пустую строку, изначально

заполняется автоматически. Если вы хотите, чтобы  $\epsilon$  представляла пустую строку, выберите **Preferences: Set the Empty String Character** в главном меню, чтобы изменить символ, обозначающий пустую строку.

Введите "0" в текстовое поле и нажмите **Enter**. Если текстовое поле не выбрано, нажмите **Tab**, чтобы выбрать его, затем введите "0" и "0". Когда вы закончите, это должно выглядеть так:

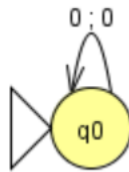



Рисунок 8 – Переход создан

Далее, мы знаем, что входные последовательности могут заканчиваться 0 и 1. Таким образом, исходящий переход для входного символа 1 из  $q_0$  должен быть в конечное состояние, так как входная последовательность, заканчивающаяся 1, должна быть принята. Чтобы создать переход от начального состояния  $q_0$  к конечному состоянию  $q_1$ , убедитесь, что инструмент **Создание переходов**  выбран на панели инструментов. Затем нажмите и удерживайте  $q_0$ , перетащите указатель мыши на  $q_1$  и отпустите кнопку мыши. Введите "1" и "0" в текстовое поле так же, как вы ввели "0" и "0" для предыдущего перехода. Переход между двумя состояниями должен выглядеть так:

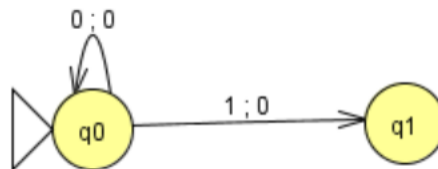
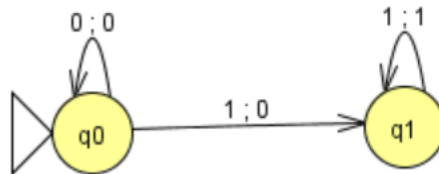


Рисунок 9 – Второй переход создан

Наконец, мы знаем, что входные последовательности могут заканчиваться как на "0" так и на "1". Таким образом,  $q_1$  может вернуться в  $q_1$  или перейти

обратно в  $q_0$ .

Создадим переход для "1" из  $q_1$  в  $q_1$ : Введите "1" в текстовое поле и нажмите **Enter**. Если текстовое поле не выбрано, нажмите **Tab**, чтобы выбрать его, затем введите "1" и "1". Когда вы закончите, это должно выглядеть так:



Похожим образом создадим переход на 0 из  $q_1$  в  $q_0$ . Это должно выглядеть примерно так:

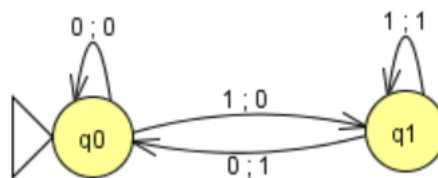
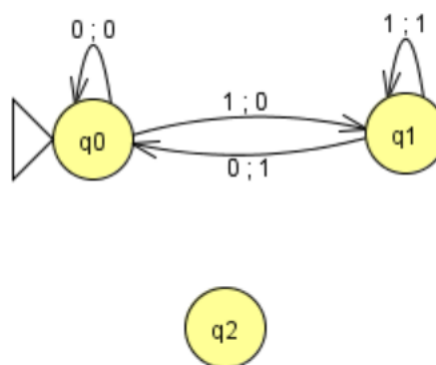



Рисунок 10 – D-триггер

Далее мы опишем, как удалять состояния и переходы.

## Удаление состояний и переходов

Добавим еще одно состояние  $q_2$ .



Чтобы удалить  $q_2$ , выберите инструмент **Удаление**  на панели инструментов. Затем нажмите на состояние  $q_2$ . Ваше окно редактора должно выглядеть примерно так:

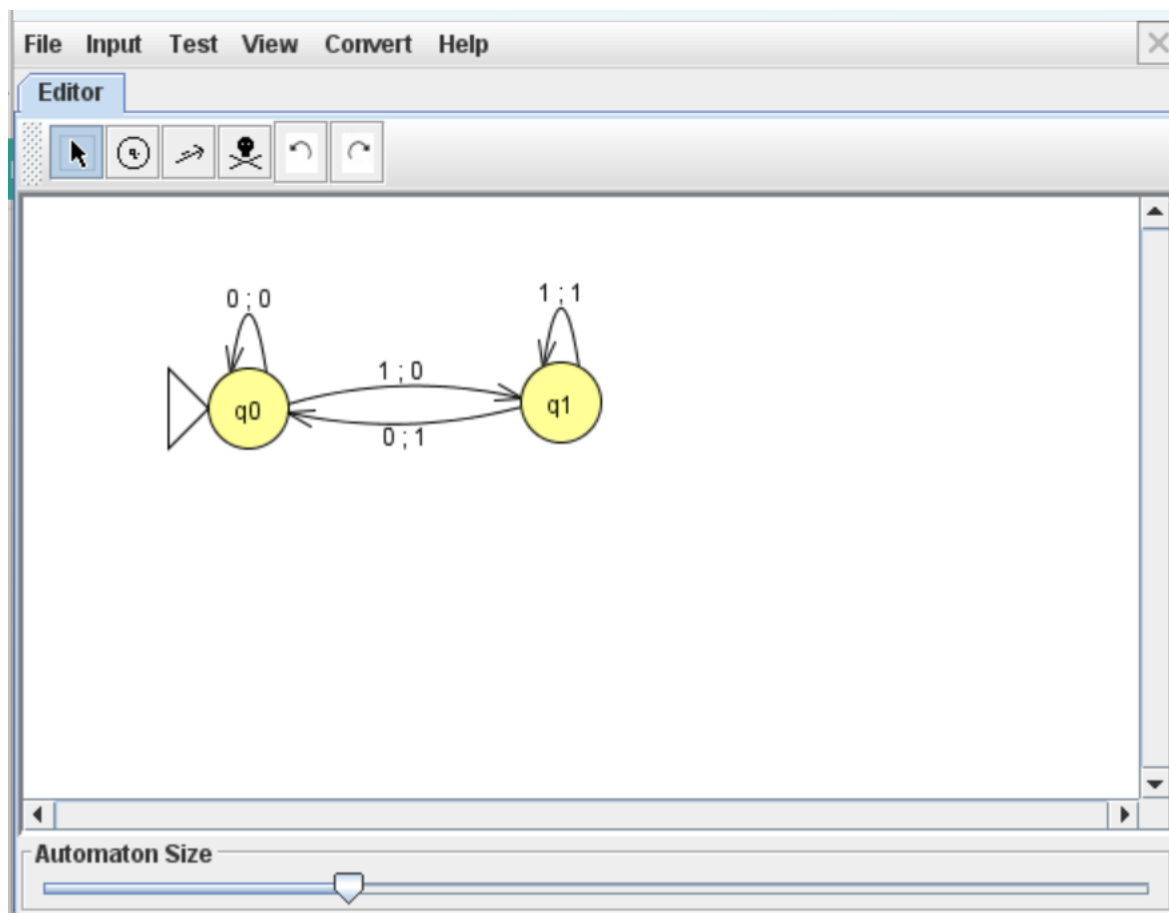


Рисунок 11 –  $q_2$  удален

Аналогично, для удаления перехода достаточно щелкнуть на входной символ перехода в режиме "Удаление".

## Обработка входных строк

Теперь, когда работа с КА закончена, можно протестировать его, чтобы убедиться, что он действительно обрабатывает входные последовательности. Для этого выберите команду **Input :Multiple Run** в строке меню:

Появится новая вкладка, на левой панели которой будет отображаться автомат, а на правой — таблица входных данных:

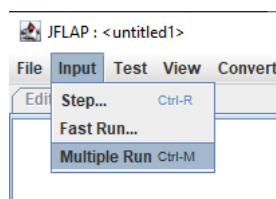


Рисунок 12 – Запуск вкладки многократного выполнения

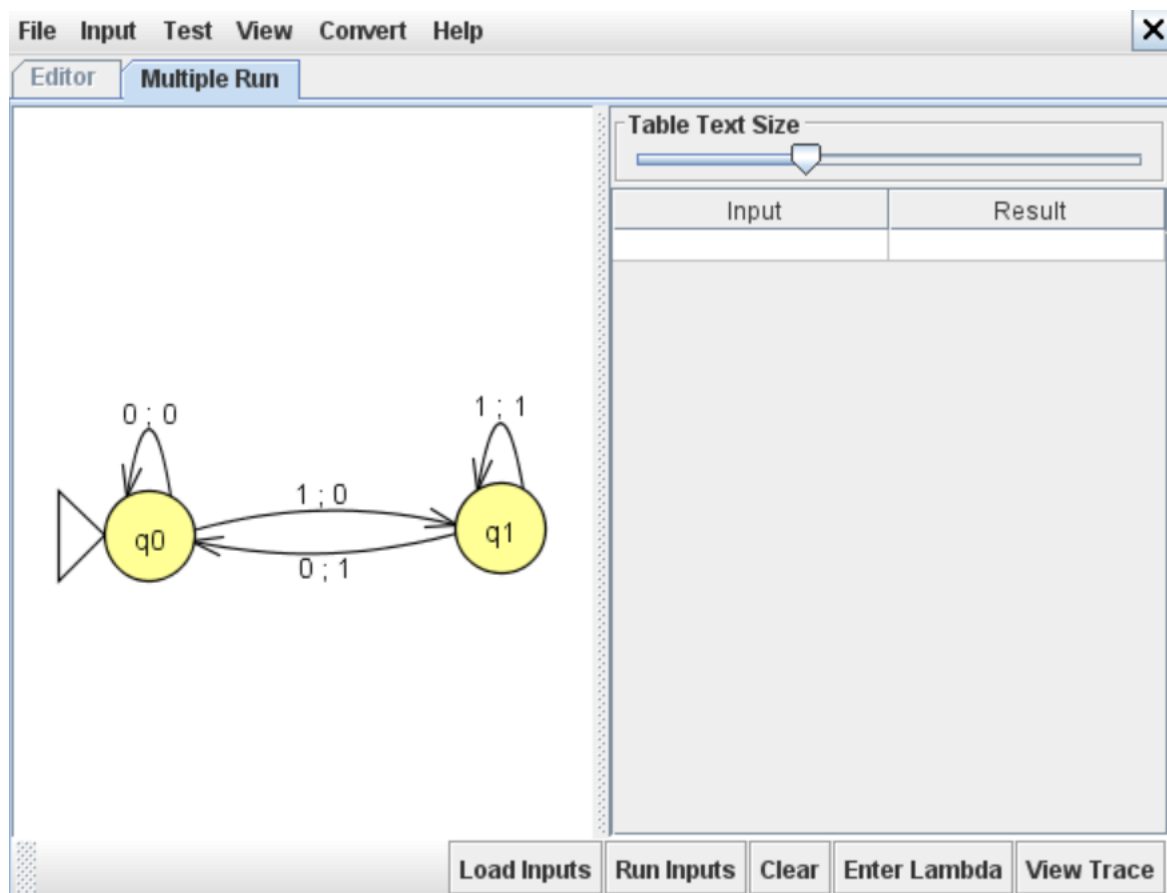


Рисунок 13 – Новая вкладка многократного выполнения

Для ввода строк щелкните по первой строке в столбце **Input** и введите входную последовательность. Нажмите **Enter**, чтобы перейти к следующей строке ввода. Когда все будет готово, нажмите кнопку **RunInputs** (Запустить), чтобы протестировать КА на всех строках ввода. Результаты отображаются в столбце **Result**. Можно также загружать входные данные из файла, разграниченного пробелами. Просто нажмите на кнопку **Load Inputs** и загрузите файл для добавления дополнительных входных строк в панель мультизапуска.

Нажатие кнопки **Clear** удаляет все вводимые строки, а **Enter Lambda**

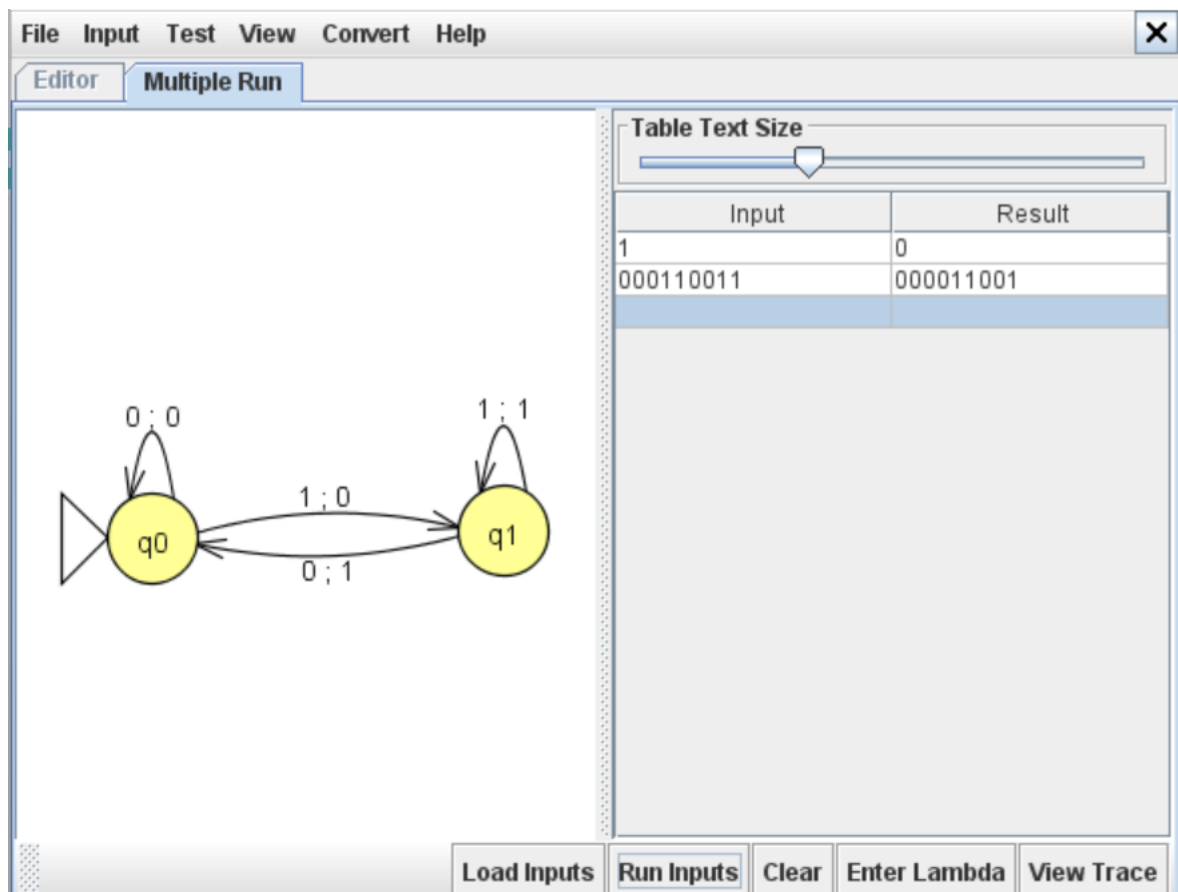


Рисунок 14 – Обработка входных строк

вводит пустую строку, находящуюся под курсором. **View Trace** вызывает отдельное окно, в котором отображается трассировка выбранного ввода. Чтобы вернуться в окно редактора, выберите в строке меню **File: Dismiss Tab**.

**Задание 1.3** Подать на вход D-тригера входные последовательности и проверить корректность их обработки.

Результаты работы оформить в виде отчета.

Требования к отчету:

1. Теоретический материал.
2. Результаты вычислительных экспериментов.
3. Выводы.