**Глава 12. Понятие алгоритма**

**12.1 Понятие алгоритмов**

Одним из инструментов современных технологий моделирования и визуализации процессов и явлений являются алгоритмы.

Алгоритм – последовательность действий, ведущих от исходных данных к заданному результату.

Свойства алгоритмов:

*Детерминированность - однозначность:* каждый шаг алгоритма ясно (однозначно) определен и не допускает двусмысленной интерпретации. После выполнения каждого шага алгоритма определяется следующий шаг или дается команда остановки, сигнализирующая о завершении работы алгоритма.

*Массовость - универсальность:* алгоритмы способны решать не только одну конкретную задачу, но и целый класс подобных задач, которые отличаются значениями исходных данных.

*Конечность - ограниченность:* алгоритмы гарантируют получение результата за конечное время. Однако существуют алгоритмически неразрешимые задачи, для которых невозможно построить алгоритмическое решение. Например, задача о трисекции угла (о разделении угла на три равные части с помощью циркуля и линейки) [4].

Формы представления алгоритмов:

*Словесное описание*: текстовое объяснение последовательности действий, необходимых для достижения результата.

*Программа на алгоритмическом языке высокого уровня (ЯВУ)*: набор инструкций, написанных на языке программирования.

*Блок-схемы*: графическое представление алгоритма, основанное на использовании различных блоков и стрелок, которые указывают на последовательность выполнения операций. Блок-схемы – это наглядная блочная логическая структура алгоритма, которая облегчает его анализ и понимание. Ниже представлен словесный алгоритм перехода проспекта, разделенного сквером, который делит проспект на две части с односторонним движением. Переход происходит вдоль улицы, которая пересекает проспект со сквером. Алгоритм не соответствует свойствам детерминированности и массовости, так как применим лишь для одного перекрестка:

1. Смотрим на светофор: зеленый – смотрите шаг 3, иначе смотрите шаг 2;

2. Стоим и ждем;

3. Смотрим на транспорт слева и идем до середины дороги;

4. Смотрим на светофор: зеленый – смотрите шаг 6, иначе смотрите шаг 5;

5. Стоим и ждем;

6. Смотрим на транспорт справа и идем до конца дороги;

7. Конец алгоритма.

Эффективность и удобство использования алгоритмов обеспечиваются требованиями к тем формам – моделям алгоритмов, которыми они представлены:

*Простота*: модели должны быть понятными и легкими для восприятия.

*Универсальность:* модели должны решать широкий класс задач. Они должны быть гибкими и адаптивными, чтобы справляться с различными типами данных и условиями, что обеспечивает их широкое применение в различных областях.

**12.2 Нормальные алгоритмы Маркова**

Нормальные алгоритмы Маркова (НАМ) – это строгая математическая форма записи алгоритмов обработки символьных строк, которую можно использовать для доказательства разрешимости или неразрешимости различных задач. Нормальные алгоритмы Маркова представляют собой правила по преобразованию слов в заданном алфавите.

*Алфавит* – непустое конечное множество символов. Может включать в себя цифры, знаки препинания.

*Слово* – произвольная последовательность символов, входящих в алфавит.

Для корректной работы НАМ задается:

1. Алфавит алгоритма
2. Конечное множество допустимых подстановок – правил
3. Порядок применения правил

Правилом подстановки является запись, показанная в формуле (читается «альфа заменить на бета») α → β, где α – левая часть правила, β – правая часть правила. Правило предписывает переменную альфа заменить на значение переменной бета.

Правила работы алгоритма Маркова:

1. На каждом шаге входящие в алгоритм правила подстановки просматриваются сверху вниз.

2. Выбирается первое из правил, применимых к входному слову. То есть в исходном слове найдена последовательность символов, равная левой части правила.

3. Часть слова, равная левой части правила, заменяется на правую часть правила.

4. Правила вновь просматриваются сверху вниз для нахождения применимых правил.

5. Если ни одно правило не может быть применено к слову, алгоритм считается завершенным.

6. Результатом работы алгоритма считается слово, получившееся после применения правил.

Пример алгоритма Маркова представлен ниже, где заданное слово «аааа» необходимо обработать при помощи двух правил и получить слово «dd»:

Слово: аааа

Правила: аа → ac, ac → d;

Решение: aaaa = acaa = acac = dac = dd

Использование правил в алгоритмах Маркова:

1. Если левая часть некоторого правила входит в слово более одного раза, то на одном шаге заменяется только самое левое вхождение.

2. Правая часть правила может быть пустой. Результатом применения такого правила является удаление из слова последовательности символов, совпадающих с левой частью правила.

3. Левая часть правила может быть пустой. В таком случае к началу слова добавляется правая часть правила. Такое преобразование считается применимым к любому слову, поэтому оно должно всегда быть последним в списке правил, иначе заблокируются все последующие действия. Если такое правило существует, то для завершения преобразований необходимо хотя бы одно терминальное правило [10].

Правило называется терминальным, если после его выполнения преобразования прекращаются, а алгоритм считается завершенным. Терминальное правило имеет собственное обозначение: ↦. В одном алгоритме может быть как несколько терминальных правил, так и ни одного. Также алгоритм может иметь одно терминальное правило. В примере представлена работа алгоритма Маркова:

Дано слово «аа» и правила подстановки:

1. \*а → аа\*
2. \* ↦
3. → \*

Решение: аа = \*аа = аа\*а = аааа\* = аааа

На рисунке 12.1 представлен второй пример нормального алгоритма Маркова, в котором работают восемь правил. Имеется исходное слово СЛОН, необходимо получить слово МУХА. После пяти применений правил выполнение алгоритма останавливается и получается итоговое слово.

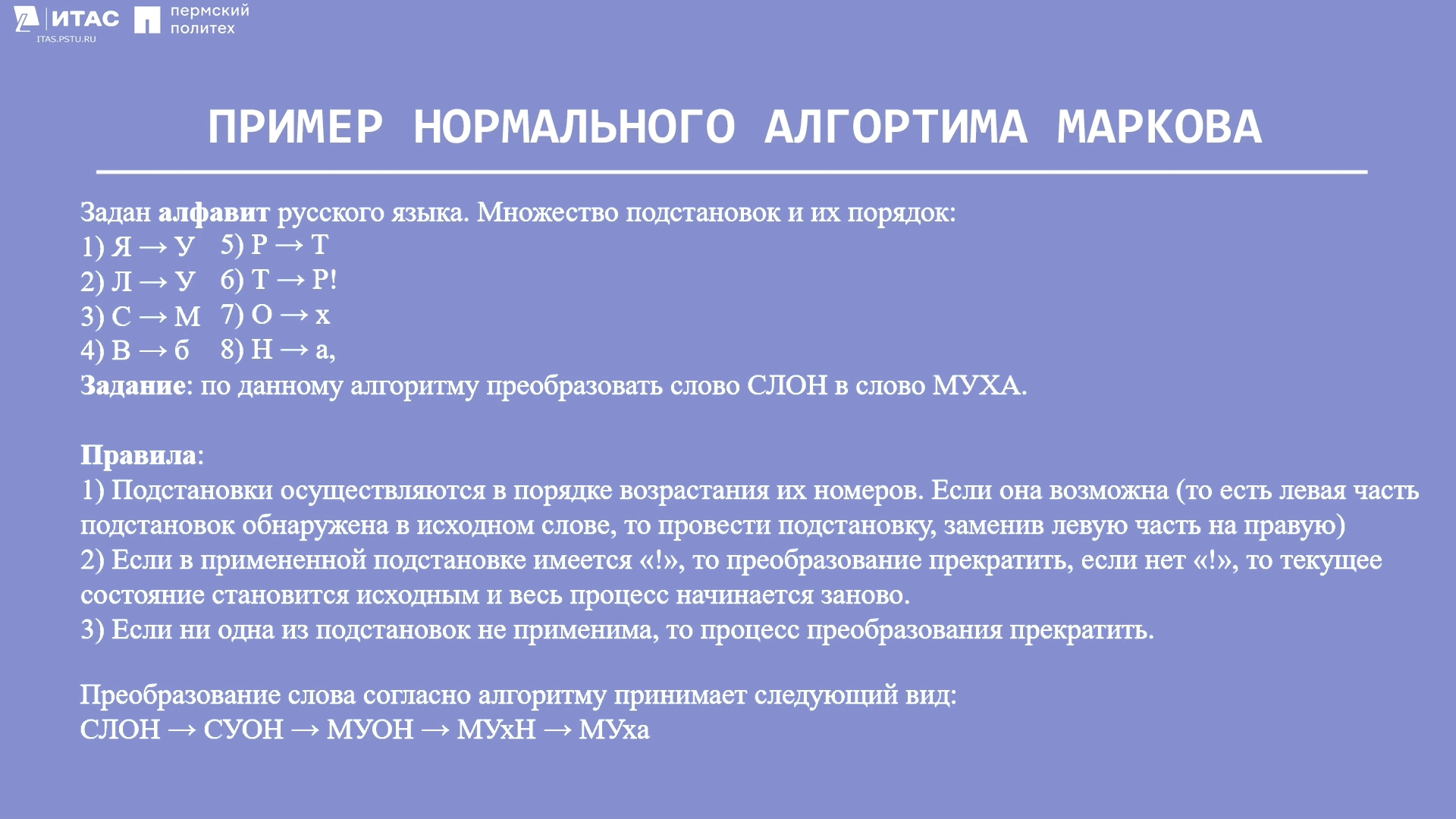
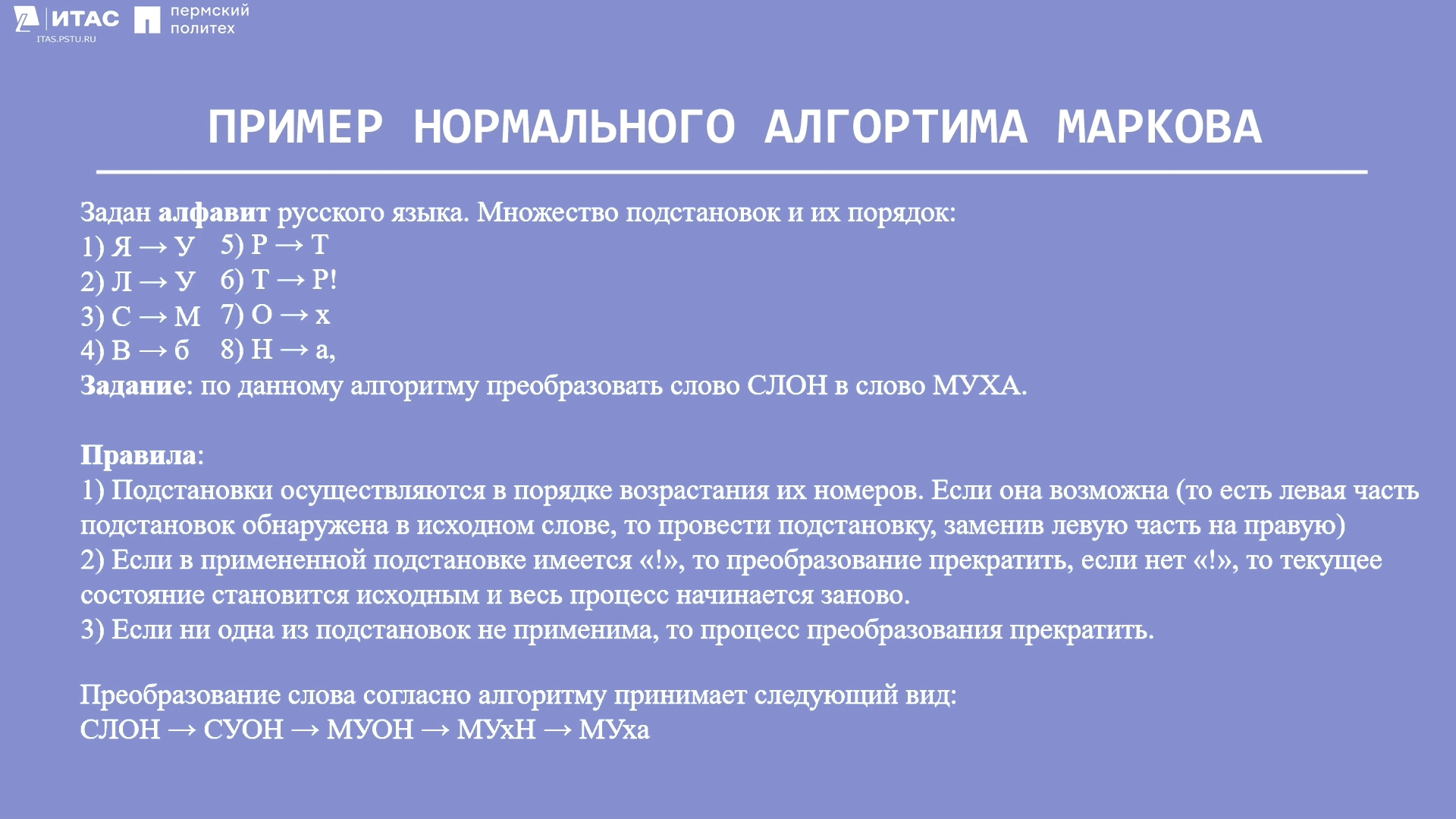


Рисунок 12.1 – Пример работы алгоритма Маркова

**12.3 Абстрактная машина Тьюринга**

Машина Тьюринга – это вычислительная машина с линейной памятью, которая, согласно заданным правилам перехода, преобразует входные данные с помощью последовательности элементарных действий.

Главные элементы машины Тьюринга (рисунок 12.2):

1. Устройство управления.

2. Голова машины.

3. Исполняющая лента.

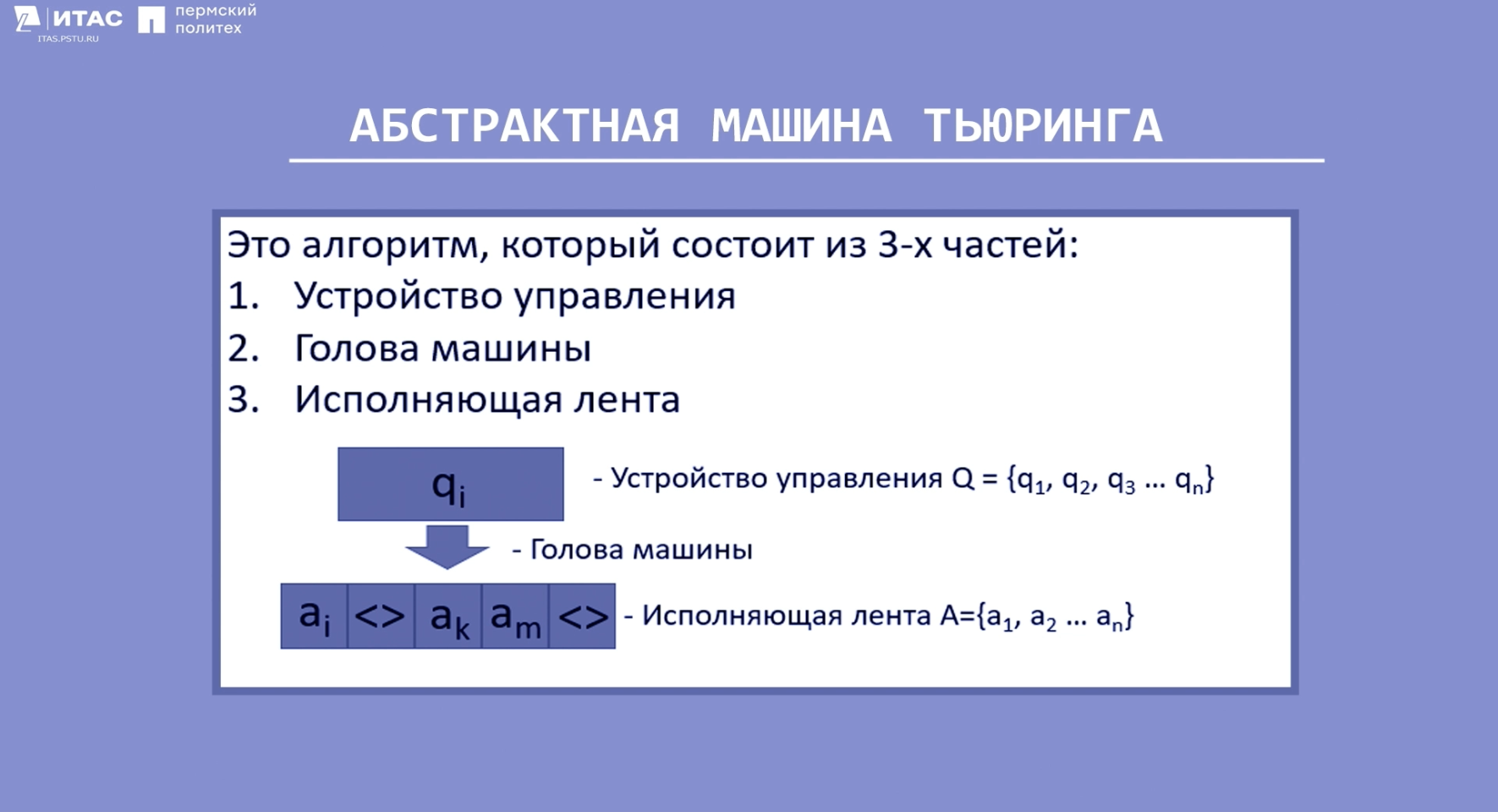


Рисунок 12.2 – Главные элементы машины Тьюринга

Машина работает следующим образом: голова может прочитать содержимое конкретной ячейки, стереть, перезаписать и передвинуться на ячейку влево или вправо и повторить считывание или запись. Действия, которые может совершать машина, записываются в виде таблицы, где строками и столбцами являются внешний и внутренний алфавиты соответственно. Внешний алфавит – это конечное множество, элементы которого называются буквами или символами (они могут быть изначально отображены на ленте). Внутренний алфавит – это конечное множество состояний головы машины. Команда, которую должна выполнить машина, определяется пересечением внешнего и внутреннего алфавитов в таблице. Пример программы машины Тьюринга представлен на рисунке 12.3.

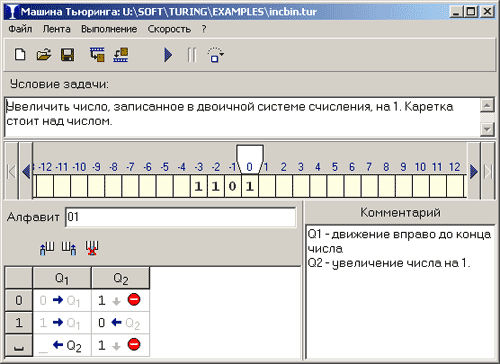


Рисунок 12.3 – Пример абстрактной машины Тьюринга

Перед началом работы представленной машины Тьюринга ее голова находится над первой слева цифрой числа.

Исходная строка на ленте = 1101

Устройство управления Q = {q1, q2}

Алфавит А = {0, 1}

Алгоритм должен прибавить к числу в двоичной системе счисления единицу. Все действия должны производиться в указанной системе счисления.

Конечная строка на ленте машины = 1110

Чтобы прибавить единицу к числу, нужно голову машины переместить так, чтоб она была над последней цифрой числа. Для этого есть состояние q1, при котором голова машины будет сдвигаться вправо до тех пор, пока не окажется за пределами числа. В таком случае голова будет сдвинута влево на одну позицию и будет запущено состояние q2. В случае, когда последняя цифра числа равна нулю, машина перезаписывает его в единицу и завершает работу, так как изменение старших разрядов числа не требуется (1100102 + 12 = 1100112). Если последняя цифра числа равна единице, алгоритм перезаписывает его в ноль и таким же образом обрабатывает соседний левый разряд. Если алгоритм обработал все разряды, включая самый старший (это возможно, когда все цифры числа равны единице), слева от него записывается единица (1112 + 12 = 10002).

Задача: составить алгоритм работы машины Тьюринга, который заменяет все нули на единицы, а все единицы – на нули. Перед началом работы машины ее голова находится над первой справа цифрой числа.

Исходная строка на ленте = 101001

Устройство управления Q = {q1}

Алфавит А = {0, 1}

Алгоритм представлен на рисунке 12.4.

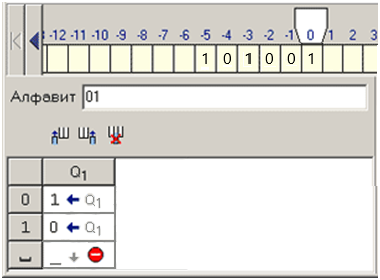


Рисунок 12.4 – Алгоритм машины Тьюринга

Принцип работы алгоритма машины Тьюринга: машина имеет единственное состояние q1. Оно отвечает за изменение символов и за перевод головы машины на соседнюю левую позицию. Изначально голова машины находится над крайней правой цифрой числа на ленте. Обрабатывая первый символ (крайний правый), алгоритм проверяет его на равенство нулю: если равенство верно, алгоритм перезаписывает ноль на единицу и сдвигает голову машины влево на одну позицию. Если вместо нуля в исходном состоянии ячейки была единица, алгоритм перезаписывает ее в ноль и точно так же сдвигает голову влево на одну позицию. Когда алгоритм обработает все символы и голова машины будет над пустой ячейкой, алгоритм завершит работу.

Таким образом, изучение базовых алгоритмов Маркова и Тьюринга позволяет пользователю лучше понимать принципы и методы работы алгоритмов в программировании. Нормальный алгоритм Маркова и абстрактная машина Тьюринга представляют собой базовую основу алгоритмов и являются основой для разработки более сложных и эффективных решений в программировании.