Министерство науки и высшего образования Российской Федерации Федеральное государственное автономное образовательное учреждение высшего образования

**«Пермский национальный исследовательский политехнический университет»**

Электротехнический факультет

Кафедра «Информационные технологии и автоматизированные системы» направление подготовки: 09.03.04 – «Программная инженерия»

**Лабораторная работа №1.**

**«Модели Машины Тьюринга и Алгоритмов Маркова»**

Выполнил студент гр. РИС-24-2б

Поспелов Василий Сергеевич

Проверил:

Доц. Каф. ИТАС

Ольга Андреевна Полякова

(оценка) (подпись)

(дата)

г. Пермь, 2024

**Нормальные алгорифмы Маркова**

* **Модель алгорифмов Маркова** преобразует исходные **слова** с помощью заданных **правил**, которые могут заменять часть исходной строки.
* **Слова** – это строки, состоящие из символов.
* **Также в алгорифмах Маркова используются приоритеты применения этих правил:**  
  1. **Правило** находит первое своё вхождение.  
  2. **Правила** выполняются по порядку их номеров.  
  3. Если **правило** не применимо к данной строке, то алгоритм переходит к выполнению следующего по порядку **правила**.  
  4. Если **правило** выполнено, то алгоритм возвращается к проверке **правил**, снова начиная с начала списка.  
  5. Алгоритм завершается, если не одно **правило** не применимо.
* **Виды подстановок:**  
  1. **Терминальная** – это подстановка, после которой выполнение алгоритма заканчивается.   
  2. **Нетерминальные** – это подстановка, которая предполагает дальнейшее применение правил.

**Решение задач**

1. **A={a,b}. Из непустого слова P удалить его первый символ. Пустое слово не менять.**

**Алгоритм:**

1. \*a > . (терминальная подстановка)
2. \*b > . (терминальная подстановка)
3. \* > . (терминальная подстановка)
4. > \* (нетерминальная подстановка)

**Пример:**

Исходная строка: ababbabba

**Ход решения:**

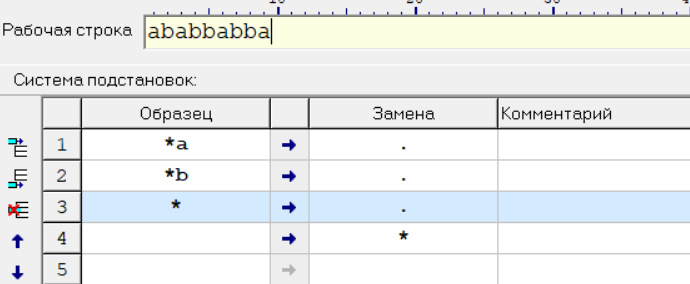
1-й шаг: Выполняется 4-е правило, так как подстрок “\*a” и “\*b” в исходной строке нет, и приписывается символ ‘\*’ в начало строки. Программа продолжает выполнение, так как 4-е правило является нетерминальной подстановкой.

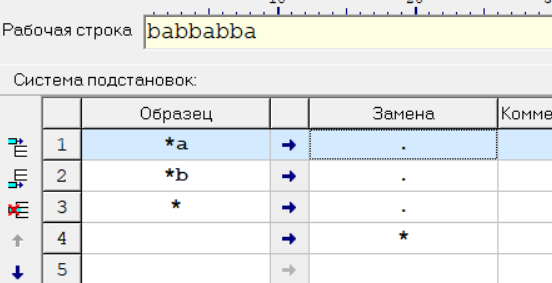
**ababbabba => \*ababbabba**

2-й шаг: Выполняется 1-е правило, и из строки исчезает подстрока “\*a”. После этого программа завершает работу, так как 1-е правило является терминальной подстановкой.

**\*ababbabba => babbabba**

**Приложение:**





1. **A={a,b,c,d}. В слове P требуется удалить все вхождения символа ‘c’, а затем заменить первое вхождение подслова “bb” на “ddd”.**

**Алгоритм:**

1. c >
2. bb > ddd.

**Пример:**

Исходная строка: baccdcbb

**Ход решения:**

1-й шаг: Выполняется 1-е правило, и удаляется первое вхождение символа ‘c’ слева. Программа продолжает выполнение, так как 1-е правило является нетерминальной подстановкой.

**baccdcbb => bacdcbb**

2-й шаг: Выполняется 1-е правило, и удаляется первое вхождение символа ‘c’ слева. Программа продолжает выполнение, так как 1-е правило является нетерминальной подстановкой.

**bacdcbb => badcbb**

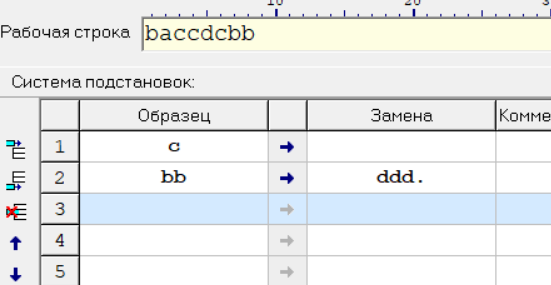
3-й шаг: Выполняется 1-е правило, и удаляется первое вхождение символа ‘c’ слева. Программа продолжает выполнение, так как 1-е правило является нетерминальной подстановкой.

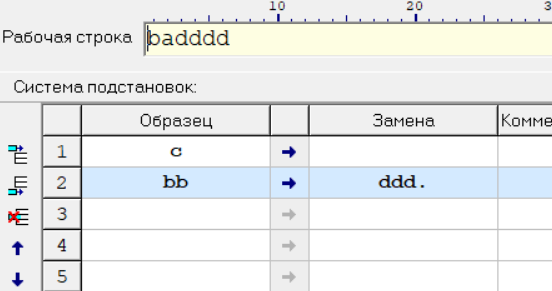
**badcbb => badbb**

4-й шаг: Выполняется 2-е правило, и первое вхождение слева подстроки “bb” заменяется на “ddd”. Программа завершает свою работу, так как 2-е правило является терминальной подстановкой.

**badbb => badddd**

**Приложение:**





1. **A={a,b}. Требуется приписать символ ‘a’ к концу слова P.**

**Алгоритм:**

1. \*a > a\*
2. \*b > b\*
3. \* > a.
4. > \*

**Пример:**

Исходная строка: ababaa

**Ход решения:**

1-й шаг: Выполняется 4-е правило, так как подстрок “\*a”, “\*b” и “\*” нет в исходной строке. Программа продолжает выполнение, так как 4-е правило является нетерминальной подстановкой.

**ababaa => \*ababaa**

2-й шаг: Выполняется 1-е правило, и первое вхождение “\*a” заменяется на “a\*”. Программа продолжает выполнение, так как 1-е правило является нетерминальной подстановкой.

**\*ababaa => a\*babaa**

3-й шаг: Выполняется 2-е правило, и первое вхождение “\*b” заменяется на “b\*”. Программа продолжает выполнение, так как 2-е правило является нетерминальной подстановкой.

**a\*babaa => ab\*abaa**

4-й шаг: Выполняется 1-е правило, и первое вхождение “\*a” заменяется на “a\*”. Программа продолжает выполнение, так как 1-е правило является нетерминальной подстановкой.

**ab\*abaa => aba\*baa**

5-й шаг: Выполняется 2-е правило, и первое вхождение “\*b” заменяется на “b\*”. Программа продолжает выполнение, так как 2-е правило является нетерминальной подстановкой.

**aba\*baa => abab\*aa**

6-й шаг: Выполняется 1-е правило, и первое вхождение “\*a” заменяется на “a\*”. Программа продолжает выполнение, так как 1-е правило является нетерминальной подстановкой.

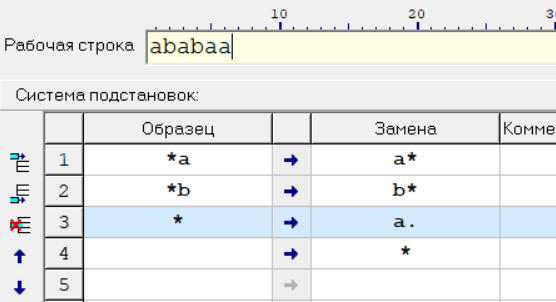
**abab\*aa => ababa\*a**

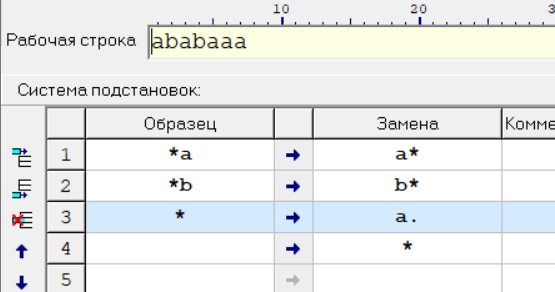
7-й шаг: Выполняется 1-е правило, и первое вхождение “\*a” заменяется на “a\*”. Программа продолжает выполнение, так как 1-е правило является нетерминальной подстановкой.

**ababa\*a => ababaa\***

8-й шаг: Выполняется 3-е правило, и символ ‘\*’ заменяется на ‘a’. Программа завершает работу, так как 3-е правило является терминальной подстановкой.

**ababaa\* => ababaaa**

**Приложение:**  
  




**Машина Тьюринга**

* **Машина Тьюринга состоит из 3-х частей:**

1. Устройство управления.
2. Голова машины.
3. Исполняющая лента.

* **Элементарный шаг машины Тьюринга:**

1. Голова считывает символ с ленты под ней.
2. Символ, который считан, и состояние головы обеспечивают переход машины Тьюринга в новое состояние  
   **q1a1> q2 a2 d**  
   Где:   
   **a1**– символ, который считывается

**q1**– прошлое состояние головы  
**q2**– новое состояние головы  
**a2**– новый записываемый символ  
**d** – перемещение головы

**Решение задач  
1. Дано число, состоящее из 0 и 1. Заменить все 0 на 1 и 1 на 0.**

**Ход мысли:** Просто меняем каждую цифру на противоположную.

|  |  |
| --- | --- |
|  | Q1 |
| 0 | 1<Q1 |
| 1 | 0<Q1 |
| ␣ | . |

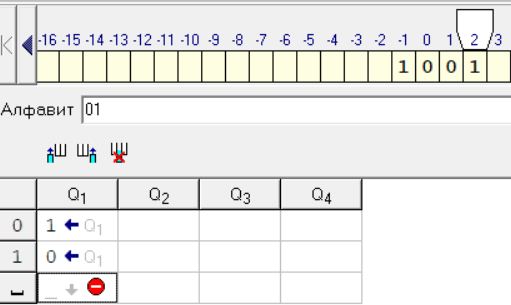
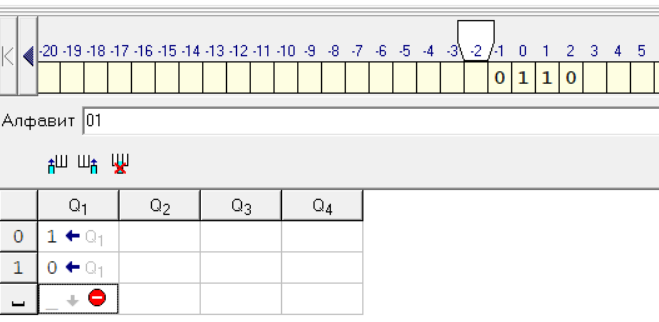
**Алгоритм:**

**Исходная строка:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  | ↓ |  |  |  |  |
|  |  |  | 1 | 0 | 0 | 1 |  |  |  |  |

**Изменения:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  | ↓ |  |  |  |  |  |
|  |  | 1 | 0 | 0 | 1 |  |  |  |  |
|  |  |  | ↓ |  |  |  |  |  |  |
|  |  | 1 | 0 | 1 | 0 |  |  |  |  |
|  |  | ↓ |  |  |  |  |  |  |  |
|  |  | 1 | 1 | 1 | 0 |  |  |  |  |
|  | ↓ |  |  |  |  |  |  |  |  |
|  |  | 0 | 1 | 1 | 0 |  |  |  |  |

**Приложение:**  
  
  
  


**2. На ввод подаётся случайное число. Прибавить 4.**

**Ход мысли:** Если цифра меньше 6, то вместо неё записываем на 4 большую и заканчиваем выполнение программы, иначе записываем вместо неё на 4 большую и уходим на следующую цифру. Далее каждую цифру увеличиваем на 1. Если цифра не равна 9, то записываем вместо неё на 1 большее и останавливаем выполнение программы, иначе записываем вместо него 0 и повторяем это заново до тех пор, пока ряд цифр не закончится.

**Алгоритм:**

|  |  |  |
| --- | --- | --- |
|  | Q1 | Q2 |
| 0 | 4<0 | 1<0 |
| 1 | 5<0 | 2<0 |
| 2 | 6<0 | 3<0 |
| 3 | 7<0 | 4<0 |
| 4 | 8<0 | 5<0 |
| 5 | 9<0 | 6<0 |
| 6 | 0<Q2 | 7<0 |
| 7 | 1<Q2 | 8<0 |
| 8 | 2<Q2 | 9<0 |
| 9 | 3<Q2 | 0<Q2 |
| ␣ | . | 1. |

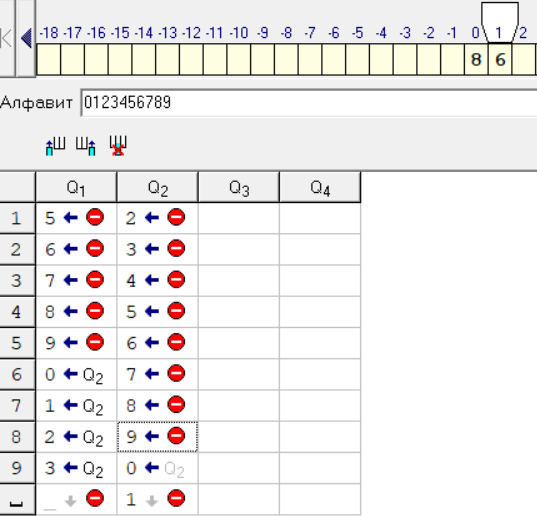
**Исходная строка:**

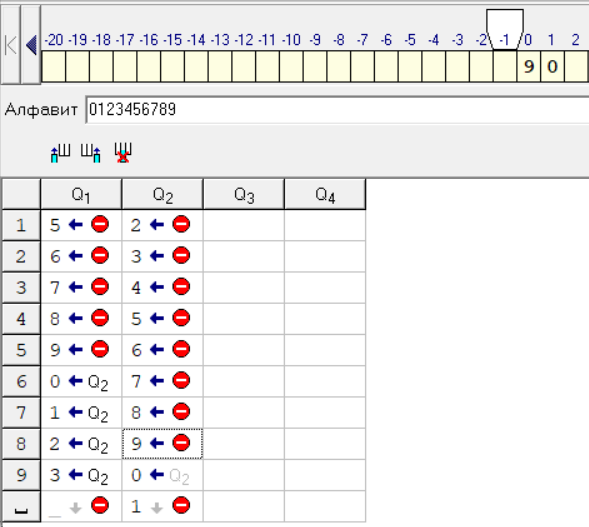
|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  | ↓ |  |  |  |
|  |  |  |  |  | 8 | 6 |  |  |  |

**Изменения:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ↓ |  |  |  |  |
|  |  |  |  |  | 8 | 0 |  |  |  |
|  |  |  |  | ↓ |  |  |  |  |  |
|  |  |  |  |  | 9 | 0 |  |  |  |

**Приложение:**





**3. На ввод подаётся случайное число. Если число чётное – заменить в нём все цифры на 0, иначе на 1.**

**Ход мысли:** Меняем все нечётные цифры на ‘1’ и чётные на ‘0’. После этого если младший разряд поменяли на ‘1’, то и все остальные цифры меняем на ‘1’, иначе все цифры меняем на ‘0’.

**Алгоритм:**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Q1 | Q2 | Q3 |
| 0 | 0<Q2 | 0<Q2 | 1<Q3 |
| 1 | 1<Q3 | 0<Q2 | 1<Q3 |
| 2 | 0<Q2 | 0<Q2 | 1<Q3 |
| 3 | 1<Q3 | 0<Q2 | 1<Q3 |
| 4 | 0<Q2 | 0<Q2 | 1<Q3 |
| 5 | 1<Q3 | 0<Q2 | 1<Q3 |
| 6 | 0<Q2 | 0<Q2 | 1<Q3 |
| 7 | 1<Q3 | 0<Q2 | 1<Q3 |
| 8 | 0<Q2 | 0<Q2 | 1<Q3 |
| 9 | 1<Q3 | 0<Q2 | 1<Q3 |
| ␣ | . | . | . |

**Исходная строка:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ↓ |  |  |  |  |
|  |  |  | 8 | 7 | 1 |  |  |  |  |

**Изменения:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  | ↓ |  |  |  |  |  |
|  |  |  | 9 | 7 | 1 |  |  |  |  |
|  |  |  | ↓ |  |  |  |  |  |  |
|  |  |  | 9 | 1 | 1 |  |  |  |  |
|  |  | ↓ |  |  |  |  |  |  |  |
|  |  |  | 1 | 1 | 1 |  |  |  |  |

**Приложение:**  
  
