**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №3**

**по дисциплине «Информатика»**

Тема: Машина Тьюринга

|  |  |  |
| --- | --- | --- |
| Студентка гр. 3343 |  | Добрякова А.А. |
| Преподаватель |  | Иванов Д. В. |

Санкт-Петербург

2023

## Цель работы

Изучение принципа работы машины Тьюринга, её реализация на языке программирования Python для решения поставленных задач.

## Задание

​**Вариант 2**

На вход программе подается строка неизвестной длины. Каждый элемент является значением в ячейке памяти ленты Машины Тьюринга.

На ленте находится последовательность латинских букв из алфавита {a, b, c}.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | a | c | c | a | b | c | b | a | b | a | a | c | a | b |  |  |  |

Напишите программу, которая заменяет в исходной строке символ, идущий после последних двух встретившихся символов 'a', на предшествующий им символ(гарантируется, что это не пробел). Наличие в строке двух подряд идущих символов 'a' гарантируется.

Указатель на текущее состояние Машины Тьюринга изначально находится слева от строки с символами (но не на первом ее символе). По обе стороны от строки находятся пробелы.

 Для примера выше лента будет выглядеть так:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | c | c | c | a | b | c | b | a | b | a | a | c | a | b |  |  |  |

Алфавит:

* a
* b
* c
* " " (пробел)

Соглашения:

1. Направление движения автомата может быть одно из R (направо), L (налево), N (неподвижно).

2. Гарантируется, что длинна строки не менее 5 символов и не более 15.

3. В середине строки не могут встретиться пробелы.

4. При удалении или вставке символов направление сдвигов подстрок не принципиально (т. е. результат работы алгоритма может быть сдвинут по ленте в любую ее сторону на любое число символов).

5. Курсор по окончании работы алгоритма может находиться на любом символе.

Ваша программа должна вывести полученную ленту после завершения работы.

## Выполнение работы

Таблица 1 - Таблица состояний

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 'a' | 'b' | 'c' | ' ' |
| q1 | 'a', R, 'q2' | 'b', R, 'q1' | 'c', R, 'q1' | ' ', R, 'q1' |
| q2 | 'a', R, 'q3' | 'b', R, 'q1' | 'c', R, 'q1' | ' ', R, 'q1' |
| q3 | 'a', L, 'q7' | 'b', L, 'q4' | 'c', L, 'q10' | ' ', L, 'q7' |
| q4 | 'a', R, 'q5' | 'b', N, 'q4' | 'c', N, 'q4' | ' ', R, 'q1' |
| q5 | 'a', R, 'q6' | 'b', N, 'q4' | 'c', N, 'q4' | ' ', R, 'q1' |
| q6 | 'b', R, 'q1' | 'b', R, 'q1' | 'b', R, 'q1' | 'b', R, 'q1' |
| q7 | 'a', L, 'q8' | 'a', L, 'q8' |  |  |
| q8 | 'a', L, 'q9' | 'a', L, 'q9' |  |  |
| q9 | 'a', R, 'qx' | 'a', R, 'qx' | 'a', R, 'qx' | 'a', R, 'qx' |
| q10 | 'a', R, 'q11' | 'b', N, 'q4' | 'c', N, 'q4' | ' ', R, 'q1' |
| q11 | 'a', R, 'q12' | 'b', N, 'q4' | 'c', N, 'q4' | ' ', R, 'q1' |
| q12 | 'c', R, 'q1' | 'c', R, 'q1' | 'c', R, 'q1' | 'c ', R, 'q1' |

Описание состояний:

* q1 - начальное состояние для поиска первого вхождения в строку символа ‘a’.
* q2 - состояние, считывающее и анализирующее символ, встретишийся после предыдущего символа ‘a’.
* q3 - состояние, определяющее символ, стоящий после двух символов ‘а’ и отправляющее программу на соответствующий трек состояний.
* q4 - состояние для символа‘b’, совершающее сдвиг влево.
* q5 - состояние для символа‘b’, совершающее сдвиг влево.
* q6 - состояние,заменяющее текущий символ на ‘b’.
* q7 - состояние для символа ‘a’, совершающее сдвиг влево.
* q8 - состояние для символа ‘a’, совершающее сдвиг влево.
* q9 – состояние, заменяющее текущий символ на ‘a’.
* q10 - состояние для символа ‘c’, совершающее сдвиг влево.
* q11 - состояние для символа ‘c’, совершающее сдвиг влево.
* q12 – состояние, заменяющее текущий символ на ‘c’.

Принцип работы Машины Тьюринга в коде:

* lenta – введенная строка (лента машины Тьюринга);
* states – таблица состояний, заданная словарем;
* current\_state – текущее состояние программы, изначально равно q1;
* i – индекс текущей ячейки, изначально равен 0;
* current\_symbol — Текущий символ, рассматриваемый программой;
* step — параметры изменения текущего символа в текущем состоянии.

Разработанный программный код см. в приложении А.

## Тестирование

Результаты тестирования представлены в табл. 2.

Таблица 2 – Результаты тестирования

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Входные данные | Выходные данные | Комментарии |
|  | aabcbcbaaaba | aabcbcbaaaaa |  |
|  | aabbaaabac | aabbaaaaac |  |

## Выводы

Был изучен принцип работы машины Тьюринга.

Была реализована программа на языке Python, решающая все поставленные задачи.

С помощью словаря была создана таблица состояний, а с помощью цикла while сымитирована работа машины Тьюринга.

# Приложение А Исходный код программы

Название файла: main.py

R, L, N = 1, -1, 0

states = {

'q1': {'a': ['a', R, 'q2'], 'b': ['b', R, 'q1'], 'c': ['c', R, 'q1'], ' ': [' ', R, 'q1']},

'q2': {'a': ['a', R, 'q3'], 'b': ['b', R, 'q1'], 'c': ['c', R, 'q1'], ' ': [' ', R, 'q1']},

'q3': {'a': ['a', L, 'q7'], 'b': ['b', L, 'q4'], 'c': ['c', L, 'q10'], ' ': [' ', L, 'q7']},

'q7': {'a': ['a', L, 'q8']},

'q8': {'a': ['a', L, 'q9']},

'q9': {'a': ['a', R, 'q\_out'], 'b': ['a', R, 'q\_out'], 'c': ['a', R, 'q\_out'], ' ': ['a', R, 'q\_out']},

'q4': {'a': ['a', L, 'q5']},

'q5': {'a': ['a', L, 'q6']},

'q6': {'a': ['b', R, 'q\_out'], 'b': ['b', R, 'q\_out'], 'c': ['b', R, 'q\_out'], ' ': ['b', R, 'q\_out']},

'q10': {'a': ['a', L, 'q11']},

'q11': {'a': ['a', L, 'q12']},

'q12': {'a': ['c', R, 'q\_out'], 'b': ['c', R, 'q\_out'], 'c': ['c', R, 'q\_out'], ' ': ['c', R, 'q\_out']},

}

lenta = list(input())[::-1]

current\_state = "q1"

i = 0

space = list(' ')

lenta = space + lenta + space

while current\_state != 'q\_out':

current\_symbol = lenta[i];

step = states[current\_state][current\_symbol];

lenta[i] = step[0]

i += step[1]

if i == len(lenta):

lenta = lenta + space

current\_state = step[2]

lenta.pop(0)

lenta.pop(len(lenta) - 1)

print(\*lenta[::-1], sep='')