**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**

|  |  |  |
| --- | --- | --- |
| Студент гр. 3343 |  | Гребнев Е.Д. |
| Преподаватель |  | Иванов Д.В. |

Санкт-Петербург

2023

## Цель работы

Разработать программу, способную моделировать работу машины Тьюринга, для работы со строками.

## Задание

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}.

Напишите программу, которая удаляет в исходной строке два символа, следующих за первым встретившимся символом «b». Если первый встретившийся символ «b» – последний в строке, то удалить его. Если первый встретившийся символ «b» – предпоследний в строке, то удалить один символ, следующий за ним, т.е. последний в строке. Если в строке символ «b» отсутствует, то удалить самый первый символ строки. После удаления в строке не должно оставаться пробелов и пустых мест.

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.

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 13.

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.

Программа должна вывести полученную ленту после завершения работы.

## Выполнение работы

Была написана программа на языке Python, где возможные состояния машины были реализованы с помощью словарей.

q1 – начальное состояние машины. Пока машина находится в этом состоянии, она передвигается вправо по ленте пока не встретит любую букву.

q1\_a – состояние, в которое машина попадает, если первая встреченная буква не была «b». В этом состоянии машина продолжает идти до ключевой буквы или символа пробела.

q2 – в это состояние машина попадает если встретила букву «b». В этом состоянии машина может находится только на следующем символе, который она удалит и перейдет к следующему состоянию.

q3 – состояние, в котором машина удаляет символ (2 справа после «b») либо останавливается, если считанный символ – пробел.

q4 – в этом состоянии машина удаляет любой символ.

q5 – данное состояние предусмотрено для того, чтобы машина могла завершить работу без изменения символов, после того, как удалит символы соответственно заданию.

q6 – состояние, рассчитанное для того сценария, если в считанной ленте нет ключевой буквы. В этом состоянии машина передвигается от правого пробела к левому, чтобы перейти от левого пробела к ближайшему символу и удалить его.

Разработанный программный код см. в приложении А.

## Тестирование

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Входные данные | Выходные данные | Комментарии |
|  | «ccacac» | «cacac» | Сценарий без буквы «b» |
|  | «acacb» | «acac» | Сценарий, где «b» находится на последнем месте |
|  | «abaacb» | «abcb» | Сценарий, когда после «b» находится >=2 символов. |

## Выводы

Была написана программа, которая может моделировать работу машины Тьюринга. Программа способна обрабатывать вводимый текст согласно заданию. В ходе разработки требовалось написать словарь, где будут описаны различные состояния машины. Затем в цикле нужные символы на считанной «ленте» заменялись согласно условию.

# ПРИЛОЖЕНИЕ А ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.py

LEFT, RIGHT, NO\_MOVE = -1, 1, 0

alphabet = {

    'q1': {'a': ['a', RIGHT, 'q1\_a'], 'b': ['b', RIGHT, 'q2'], 'c': ['c', RIGHT, 'q1\_a'], ' ': [' ', RIGHT, 'q1']},

    'q1\_a': {'a': ['a', RIGHT, 'q1\_a'], 'b': ['b', RIGHT, 'q2'], 'c': ['c', RIGHT, 'q1\_a'], ' ': ['', LEFT, 'q6']},

    'q2': {'a': ['', RIGHT, 'q3'], 'b': ['', RIGHT, 'q3'], 'c': ['', RIGHT, 'q3'], ' ': ['', LEFT, 'q4']},

    'q3': {'a': ['', RIGHT, 'q5'], 'b': ['', RIGHT, 'q5'], 'c': ['', RIGHT, 'q5'], ' ': ['', NO\_MOVE, 'qT']},

    'q4': {'a': ['', NO\_MOVE, 'qT'], 'b': ['', NO\_MOVE, 'qT'], 'c': ['', NO\_MOVE, 'qT']},

    'q5': {'a': ['a', RIGHT, 'q5'], 'b': ['b', RIGHT, 'q5'], 'c': ['c', RIGHT, 'q5'], ' ': ['', RIGHT, 'qT']},

    'q6': {'a': ['a', LEFT, 'q6'], 'b': ['b', LEFT, 'q6'], 'c': ['c', LEFT, 'q6'], ' ': ['', RIGHT, 'q4']},

}

current\_state = 'q1'

cursor\_position = 0

tape = list(' ' + input() + ' ')

while current\_state != 'qT':

    symbol, move, next\_state = alphabet[current\_state][tape[cursor\_position]]

    tape[cursor\_position], cursor\_position, current\_state = symbol, cursor\_position + move, next\_state

print(''.join(tape))