**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**](https://e.moevm.info/mod/quiz/view.php?id=2027)

|  |  |  |
| --- | --- | --- |
| Студент гр. 3342 |  | Малахов А.И. |
| Преподаватель |  | Иванов Д.В. |

Санкт-Петербург

2023

## Цель работы

Целью работы реализация машины Тьюринга на языке программирования Python.

## Задание

Вариант 3.

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 заменяет в исходной строке символ, предшествующий первому встретившемуся символу ‘c' на символ, следующий за первым встретившимся символом ‘a'. Если первый встретившийся символ ‘a' в конце строки, то используйте его в качестве заменяющего.

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 длина 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

На основе условия задачи была составлена таблица состояний №1.

Таблица 1: Таблица состояний №1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | «a» | «b» | «c» | « » |
| q0 | «a», R, q1 | «c», R, q0 | «c», R, q0 | « », R, q0 |
| q1 | «a», L, q2- | «b», L, q3- | «c», L, q4- | « », L, q2- |
| q2- | «a», L, q2- | «b», L, q2- | «c», L, q2- | « », R, q2+ |
| q2+ | «a», R, q2+ | «b», R, q2+ | «c», L, q5 |  |
| q5 | «a», N, q8 | «a», N, q8 | «a», N, q8 | «a», N, q8 |
| q3- | «a», L, q3- | «b», L, q3- | «c», L, q3- | « », R, q3+ |
| q3+ | «a», R, q3+ | «b», R, q3+ | «c», L, q6 |  |
| q6 | «b», N, q8 | «b», N, q8 | «b», N, q8 | «b», N, q8 |
| q4- | «a», L, q4- | «b», L, q4- | «c», L, q4- | « », R, q4+ |
| q4+ | «a», R, q4+ | «b», R, q4+ | «c», L, q7 |  |
| q7 | «c», N, q8 | «c», N, q8 | «c», N, q8 | «c», N, q8 |

* «q0» - начальное состояние, поиск первого символа «а»
* «q1» - нахождение символа, стоящего после «а»
* «q2-», «q3-», «q4-» - возвращение к началу строки и сохранение найденного символа в зависимости от состояния
* «q2+», «q3+», «q4+» - поиск первого символа «с»
* «q5», «q6», «q7» - перезаписывание символа, стоящего перед «с», на символ, найденный после символа «а»
* «q8» - конечное положение

## Тестирование

Результаты тестирования представлены в табл. 2.

Таблица 2 – Результаты тестирования

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Входные данные | Выходные данные | Комментарии |
|  | abcabc | abcabc | Функция работает корректно |
|  | cbbaa | acbbaa | Функция работает корректно |
|  | cbbab | bcbbab | Функция работает корректно |

## Выводы

Была разработана программа на языке Python, которая проводит работу со строкой по определенному алгоритму Машины Тьюринга. Изучена Машина Тьюринга.

# Приложение А Исходный код программы

Название файла: main.py

table = {

'q0':{

'a': ['a', 1, 'q1'],

'b': ['b', 1, 'q0'],

'c': ['c', 1, 'q0'],

' ': [' ', 1, 'q0']

},

'q1':{

'a': ['a', -1, 'q2-'],

'b': ['b', -1, 'q3-'],

'c': ['c', -1, 'q4-'],

' ': [' ', -1, 'q2-']

},

'q2-':{

'a': ['a', -1, 'q2-'],

'b': ['b', -1, 'q2-'],

'c': ['c', -1, 'q2-'],

' ': [' ', 1, 'q2+']

},

'q2+': {

'a': ['a', 1, 'q2+'],

'b': ['b', 1, 'q2+'],

'c': ['c', -1, 'q5']

},

'q5': {

'a': ['a', 0, 'q8'],

'b': ['a', 0, 'q8'],

'c': ['a', 0, 'q8'],

' ': ['a', 0, 'q8']

},

'q3-': {

'a': ['a', -1, 'q3-'],

'b': ['b', -1, 'q3-'],

'c': ['c', -1, 'q3-'],

' ': [' ', 1, 'q3+']

},

'q3+': {

'a': ['a', 1, 'q3+'],

'b': ['b', 1, 'q3+'],

'c': ['c', -1, 'q6']

},

'q6': {

'a': ['b', 0, 'q8'],

'b': ['b', 0, 'q8'],

'c': ['b', 0, 'q8'],

' ': ['b', 0, 'q8']

},

'q4-': {

'a': ['a', -1, 'q4-'],

'b': ['b', -1, 'q4-'],

'c': ['c', -1, 'q4-'],

' ': [' ', 1, 'q4+']

},

'q4+': {

'a': ['a', 1, 'q4+'],

'b': ['b', 1, 'q4+'],

'c': ['c', -1, 'q7']

},

'q7': {

'a': ['c', 0, 'q8'],

'b': ['c', 0, 'q8'],

'c': ['c', 0, 'q8'],

' ': ['c', 0, 'q8']

},

}

memory = list(input())

q = 'q0'

index = 0

while q != 'q8':

s, move, state = table[q][memory[index]]

memory[index] = s

index += move

q = state

print(''.join(memory), sep='\n')