**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. 3341 |  | Шаповаленко Е.В. |
| Преподаватель |  | Иванов Д.В. |

Санкт-Петербург

2023

## Цель работы

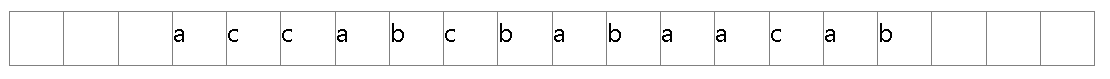
Написать программу, которая удаляет в исходной строке символы в соответствии с позицией первого символа 'b', имитируя работу машины Тьюринга.

## Задание

Вариант 1

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

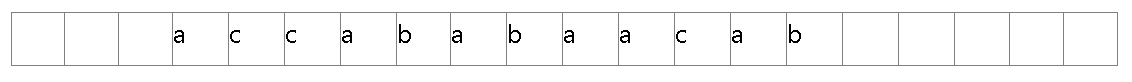
На ленте находится последовательность латинских букв из алфавита {a, b, c}.



**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.

Для примера выше лента будет выглядеть так:



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.

Ваша программа должна вывести полученную ленту после завершения работы.

В отчете предоставьте таблицу состояний. Отдельно кратко опишите каждое состояние, например:

q1 - начальное состояние, которое необходимо, чтобы найти первый встретившийся символ ‘b’.

## Основные теоретические положения

Машина Тьюринга состоит из неподвижной бесконечной ленты и автомата. Лента используется для хранения информации и разбита на клетки. Алфавит ленты - конечное множество символов, которые могут быть записаны на ленте. Автомат является активной частью Машины Тьюринга и размещается под одной из клеток ленты. Автомат видит содержимое только видимой клетки и находится в одном из конечного числа состояний. В каждом состоянии автомат выполняет определенную операцию и может совершить определенные действия. Программа для Машины Тьюринга может быть записана в виде таблицы, где указываются символы, направления и состояния.

## Выполнение работы

1. q1 - Начальное состояние, чтобы найти первый встретившийся символ 'b'. Смещение вправо с переходом на состояние q2 при обнаружении символа 'a', 'b', 'c', либо просто смещение, если встречен пробел.

2. q2 - Поиск символа 'b', если он не был найден в q1. В случае обнаружения символа 'b', переход на состояние q3 после смещения каретки на один символ вправо. Если обнаружен пробел, смещение влево и переход в состояние q18.

3. q3 - Проверка первого символа после 'b'. Если 'b' – последний, сдвиг влево и переход в состояние q17. Если нет – замена символа на пробел, сдвиг вправо, переход в состояние q4.

4. q4 - Проверка второго символа после 'b'. Если пробел – переход в конечное состояние q20. Иначе – замена на пробел, сдвиг вправо, переход в состояние q5.

5. q5 – Проверка третьего символа после 'b'. Если пробел – переход в конечное состояние q20. Иначе – сдвиг влево, переход в состояние q6.

6. q6 – Сдвиг влево, если пробел, иначе переход в состояние q7.

7. q7 – Сдвиг влево в начало слова. Если найден пробел – переход в состояние q8.

8. q8, q9, q10, q11 – Записывает в ячейку символ в соответствии с состоянием, переходит в состояние в соответствии с символом в ячейке до записи, сдвигается вправо. Таким образом осуществляется первый сдвиг ячеек на один шаг.

9. q12, q13, q14, q15, q16 – аналогичные шаги, как для q7, q8, q9, q10, q11. Таким образом осуществляется второй сдвиг ячеек на один шаг. После этого – переход в конечное состояние q20.

10. q17 - символ 'b' – последний. Замена символа на пробел, переход в конечное состояние q20.

11. q18 – символ 'b' не найден. Сдвиг влево до начала. Если найден пробел – переход в состояние q19.

12. q19 – символ заменяется на пробел, переход в конечное состояние q20.

Код осуществляет обработку каждого состояния и переходов между ними, в результате изменяя входную ленту согласно условиям задачи.

По шагам работы кода:

1. Запись ленты из потока ввода в список tape.

2. Инициализация начального состояния status = 'q1', установка каретки в начале ленты head = 0.

3. Вход в цикл while status != 'q20'.

4. Извлечение символа (data), нового состояния (status\_update) и направления сдвига (move) из таблицы состояний status\_table[status][tape[head]].

5. Обновление символа на ленте tape[head] = data.

6. Сдвиг каретки в нужную сторону в соответствии с направлением head\_moves[move].

7. Обновление текущего состояния status = status\_update.

8. Повторение шагов 4-7 до достижения конечного состояния q20.

9. Вывод измененной ленты после завершения работы Машины Тьюринга.

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 |
|  | abca | ab | - |
|  | abc | ab | - |
|  | ab | a | - |
|  | acc | cc | - |

## Выводы

В результате работы было продемонстрировано применение Машины Тьюринга для решения задачи поиска и удаления символа в строке.

# Приложение А Исходный код программы

Название файла: main.py

head\_moves = {'R': 1, 'L': -1, 'N': 0}

status\_table = {

'q1': {'a': 'a,R,q2', 'b': 'b,R,q3', 'c': 'c,R,q2', ' ': ' ,R,q1'}, #поиск начала строки

'q2': {'a': 'a,R,q2', 'b': 'b,R,q3', 'c': 'c,R,q2', ' ': ' ,L,q18'}, #поиск 1го символа b или его отсутствия

'q3': {'a': ' ,R,q4', 'b': ' ,R,q4', 'c': ' ,R,q4', ' ': ' ,L,q17'}, #поиск 1го символа после b(замена на ' ') или его отсутствия

'q4': {'a': ' ,R,q5', 'b': ' ,R,q5', 'c': ' ,R,q5', ' ': ' ,N,q20'}, #поиск 2го символа b(замена на ' ') или его отсутствия

'q5': {'a': 'a,L,q6', 'b': 'b,L,q6', 'c': 'c,L,q6', ' ': ' ,N,q20'}, #поиск хвоста

'q6': {'a': 'a,L,q7', 'b': 'b,L,q7', 'c': 'c,L,q7', ' ': ' ,L,q6'}, #сдвиг до разрыва(конец головы)

'q7': {'a': 'a,L,q7', 'b': 'b,L,q7', 'c': 'c,L,q7', ' ': ' ,R,q8'}, #1й сдвиг до начала

'q8': {'a': ' ,R,q9', 'b': ' ,R,q10', 'c': ' ,R,q11'}, #начало 1го сдвига посимвольно на 1 шаг

'q9': {'a': 'a,R,q9', 'b': 'a,R,q10', 'c': 'a,R,q11', ' ': 'a,L,q12'}, #сдвиг головы к хвосту на 1 шаг. Если найден пробел -- 2й сдвиг до начала

'q10': {'a': 'b,R,q9', 'b': 'b,R,q10', 'c': 'b,R,q11', ' ': 'b,L,q12'},

'q11': {'a': 'c,R,q9', 'b': 'c,R,q10', 'c': 'c,R,q11', ' ': 'c,L,q12'},

'q12': {'a': 'a,L,q12', 'b': 'b,L,q12', 'c': 'c,L,q12', ' ': ' ,R,q13'},#2й сдвиг до начала

'q13': {'a': ' ,R,q14', 'b': ' ,R,q15', 'c': ' ,R,q16'}, #начало 2го сдвига посимвольно на 1 шаг

'q14': {'a': 'a,R,q14', 'b': 'a,R,q15', 'c': 'a,R,q16', ' ': 'a,N,q20'}, #сдвиг головы к хвосту на 1 шаг. Если найден пробел -- конец работы

'q15': {'a': 'b,R,q14', 'b': 'b,R,q15', 'c': 'b,R,q16', ' ': 'b,N,q20'},

'q16': {'a': 'c,R,q14', 'b': 'c,R,q15', 'c': 'c,R,q16', ' ': 'c,N,q20'},

'q17': {'b': ' ,N,q20'}, #b -- последний, замена на ' '

'q18': {'a': 'a,L,q18', 'b': 'b,L,q18', 'c': 'c,L,q18', ' ': ' ,R,q19'}, #сдвиг до начала для удаления 1го символа

'q19': {'a': ' ,N,q20', 'b': ' ,N,q20', 'c': ' ,N,q20'}, #удаление 1го символа

}

tape = list(input())

status = 'q1'

head = 0

while status != 'q20':

data, move, status\_update = status\_table[status][tape[head]].split(',')

tape[head] = data

head += head\_moves[move]

status = status\_update

print(''.join(tape))