**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работe №3**

**по дисциплине «Информатика»**

Тема: Машина Тьюринга

|  |  |  |
| --- | --- | --- |
| Студент гр. 3344 |  | Охрименко Д.И |
| Преподаватель |  | Иванов Д.В. |

Санкт-Петербург

2023

## Цель работы

Изучить принцип работы машины Тьюринга. На практике научиться составлять таблицы состояний автомата в виде словаря для написания рабочей программы на языке программирования Python.

## 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.

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.

## Выполнение работы

Вводим переменные R, L, N для сдвига головки автомата.

Считанная с клавиатуры строка сохранена в виде списка line. Задано начальное состояние автомата — state = "qStrt", head = 0,   
direction = 0. Оформлен словарь, содержащий ключи — состояния автомата и значения — словари, возвращающие по текущему символу кортеж из заменяющего символа, сдвига, последующего состояния.

Состояния:

qStrt — начальное состояние до нахождения первого символа строки;

q1 — нахождение символа b с проходом до конца строки;

q2 — случай (1): символа b нет;

q3, q4, q5posta, q5postc — состояния для замены символов для случая (1);

q6 — случай (2): b в конце строки;

q7 — обработка случая (2);

q8 — случай (3): после b идёт один символ;

q9 — обработка случая (3);

q10, q11, q12posta, q12postb, q12postc, q12postv — состояния для замены символов для случая (3);

q13 — возвращение к началу строки;

q14, q15, q16a, q16b, q16c — состояния для замены символов для случая (4): после b идёт 2 символа;

qEx — состояние останавливающее работу как автомата, так и программы.

В цикле while изменяем положение головки автомата и видимый символ до того, как не получим на входе состояние "qEx".

Таблица состояний представлена в табл. 1

Таблица 1 - Таблица состояний

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | ‘a’ | 'b’ | ‘c’ | ‘ ‘ |
| qStrt | 'a', 1, 'q1' | 'b', 1, 'q6' | 'c', 1, 'q1' | ' ', 1, 'qStrt' |
| q1 | 'a', 1, 'q1' | 'b', 1, 'q6' | 'c', 1, 'q1' | ' ', -1, 'q2' |
| q2 | 'a', -1, 'q2' | — | 'c', -1, 'q2' | ' ', 1, 'q3' |
| q3 | ' ', 1, 'q4' | — | ' ', 1, 'q4' | ' ', 1, 'q4' |
| q4 | ' ', -1, 'q5posta' | — | ' ', -1,'q5postc' | ' ', 1, 'qEx' |
| q5posta | — | — | — | 'a', 1, 'q3' |
| q5postc | — | — | — | 'c', 1, 'q3' |
| q6 | ' ', 1, 'q8' | ' ', 1, 'q8' | ' ', 1, 'q8' | ' ', -1, 'q7' |
| q7 | — | ' ', 1, 'qEx' | — | — |
| q8 | 'a', 0, 'q10' | 'b', 0,'q10' | 'c', 0, 'q10' | ' ', -1, 'q9' |
| q9 | ' ', 1, 'qEx' | ' ', 1, 'qEx' | ' ', 1, 'qEx' | ' ', 1, 'qEx' |
| q10 | ' ', 1, 'q11' | ' ', 1, 'q11' | ' ', 1, 'q11' | ' ', 1, 'q11' |
| q11 | ' ', -1, 'q12posta' | ' ', -1, 'q12postb' | ' ', -1, 'q12postc' | ' ', -1, 'q12postv' |
| q12posta | — | — | — | 'a', 1, 'q10' |
| q12postb | — | — | — | 'b', 1, 'q10' |
| q12postc | — | — | — | 'c', 1, 'q10' |
| q12postv | — | — | — | ' ', -1, 'q13' |
| q13 | 'a', -1, 'q13' | 'b', -1, 'q13' | 'c', -1, 'q13' | ' ', 0, 'q14' |
| q14 | ' ', 1, 'q15' | ' ', 1, 'q15' | ' ', 1, 'q15' | ' ', 1, 'q15' |
| q15 | ' ', -1, 'q16posta' | ' ', -1, 'q16postb' | ' ', -1, 'q16postc' | ' ', 1, 'qEx' |
| q16posta | — | — | — | 'a', 1, 'q14' |
| q16postb | — | — | — | 'b', 1, 'q14' |
| q16postc | — | — | — | 'c', 1, 'q14' |

Исходный код см. в приложении A

## Тестирование

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Входные данные | Выходные данные | Комментарии |
| 1. | aaaabbbbbbb | aaaabbbbb | Верный вывод |
| 2. | abcabcabc | abbcabc | Верный вывод |
| 3. | aaaaaaaaaaaaaab | aaaaaaaaaaaaaa | Верный вывод |

## Выводы

Была исследована схема работы машины Тьюринга. Изменяя состояния автомата происходило посимвольное изменение входящей строки. Написан словарь со всеми состояниями и описанием действий головки автомата, с помощью чего реализована программа.

# Приложение А Исходный код программы

Название файла: Okhrimenko\_Denis\_lb3.py

R = 1

L = -R

N = 0

states = {

"qStrt": {

"a": ("a", R, "q1"),

"b": ("b", R, "q6"),

"c": ("c", R, "q1"),

" ": (" ", R, "qStrt"),

},

"q1": {

"a": ("a", R, "q1"),

"b": ("b", R, "q6"),

"c": ("c", R, "q1"),

" ": (" ", L, "q2"),

},

"q2": {"a": ("a", L, "q2"), "c": ("c", L, "q2"), " ": (" ", R, "q3")},

"q3": {"a": (" ", R, "q4"), "c": (" ", R, "q4"), " ": (" ", R, "q4")},

"q4": {"a": (" ", L, "q5posta"), "c": (" ", L, "q5postc"), " ": (" ", R, "qEx")},

"q5posta": {" ": ("a", R, "q3")},

"q5postc": {" ": ("c", R, "q3")},

"q6": {

"a": (" ", R, "q8"),

"b": (" ", R, "q8"),

"c": (" ", R, "q8"),

" ": (" ", L, "q7"),

},

"q7": {"b": (" ", R, "qEx")},

"q8": {

"a": ("a", 0, "q10"),

"b": ("b", 0, "q10"),

"c": ("c", 0, "q10"),

" ": (" ", L, "q9"),

},

"q9": {

"a": (" ", R, "qEx"),

"b": (" ", R, "qEx"),

"c": (" ", R, "qEx"),

" ": (" ", R, "qEx"),

},

"q10": {

"a": (" ", R, "q11"),

"b": (" ", R, "q11"),

"c": (" ", R, "q11"),

" ": (" ", R, "q11"),

},

"q11": {

"a": (" ", L, "q12posta"),

"b": (" ", L, "q12postb"),

"c": (" ", L, "q12postc"),

" ": (" ", L, "q12postv"),

},

"q12posta": {" ": ("a", R, "q10")},

"q12postb": {" ": ("b", R, "q10")},

"q12postc": {" ": ("c", R, "q10")},

"q12postv": {" ": (" ", L, "q13")},

"q13": {

"a": ("a", L, "q13"),

"b": ("b", L, "q13"),

"c": ("c", L, "q13"),

" ": (" ", 0, "q14"),

},

"q14": {

"a": (" ", R, "q15"),

"b": (" ", R, "q15"),

"c": (" ", R, "q15"),

" ": (" ", R, "q15"),

},

"q15": {

"a": (" ", L, "q16posta"),

"b": (" ", L, "q16postb"),

"c": (" ", L, "q16postc"),

" ": (" ", R, "qEx"),

},

"q16posta": {" ": ("a", R, "q14")},

"q16postb": {" ": ("b", R, "q14")},

"q16postc": {" ": ("c", R, "q14")},

}

line = list(input())

state = "qStrt"

head = 0

direction = 0

ans\_states = ["qStrt"]

while state != "qEx":

line[head], direction, state = states[state][line[head]]

head += direction

ans\_states += [state]

print("".join(line))