**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. 3344 |  | Кузнецов Р.А. |
| Преподаватель |  | Иванов Д.В. |

Санкт-Петербург

2023

## Цель работы

Изучение и реализация базовых принципов работы машины Тьюринга на языке 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 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

Для работы был создан словарь состояний, который задает поведение машины Тьюринга. Состояния являются ключами к значениям словарей поменьше. Состояние q\_start ищет первый символ «а» в ленте. При встрече же других символов, курсор передвигается дальше и остается в состоянии q\_start. Когда символ найден, то машина переходит в состояние q\_after\_a. Это состояние проверяет какой символ находится после «a» и переходит в состояние соответствующее символу. Курсор двигается в начало ленты(q1.a, q1.b, q1.c). Затем курсор двигается, пока не встретится символ «c» (q2.a, q2.b, q2.c). В зависимости от символа, перед «c» ставится соответствующий символ(q3.a, q3.b, q3.c) и машина переходит в состояние q\_end. Если же символ после «a» это пробел, то машина переходит в состояние q\_void. Дальше курсор переносится в начало строки, ищется символ «c» и перед ним вставляется символ «a». Машина переходит в состояние q\_end. Основное тело программы: считывается строка и приводится к списку. Задается начальное состояние и позиция курсора. Выполняется цикл пока состояние не будет q\_end. В переменные считываются соответствующее значение. На изначальное место символа, ставится символ указанный в словаре. Затем проверяется движение курсора и добавляется к переменной position. Потом машина переходит в другое состояние или же остается в текущем. Выводится полученная лента.

## Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Входные данные | Выходные данные | Комментарии |
|  | abcabc | abcabc | - |
|  | cbbaa | acbbaa | - |
|  | bbcbbcaa | bacbbcaa | - |

## Выводы

Были изучены базовые принципы работы машины Тьюринга. Была реализована машина Тьюринга на языке Python.

# Приложение А Исходный код программы

Название файла: lb3.py

tape = list(input())

machine = {

#ищем а

"q\_start": {"a": ("a", "R", "q\_after\_a"), "b": ("b", "R", "q\_start"), "c": ("c", "R", "q\_start"), " ": (" ", "R", "q\_start")},

# символ после а

"q\_after\_a": {"a": ("a", "N", "q\_a.1"), "b": ("b", "N", "q\_b.1"), "c": ("c", "N", "q\_c.1"), " ": (" ", "L", "q\_void")},

#вставляем а

"q\_a.1": {"a": ("a", "L", "q\_a.1"), "b": ("b", "L", "q\_a.1"), "c": ("c", "L", "q\_a.1"), " ": (" ", "R", "q\_a.2")},

"q\_a.2": {"a": ("a", "R", "q\_a.2"), "b": ("b", "R", "q\_a.2"), "c": ("c", "L", "q\_a.3")},

"q\_a.3": {"a": ("a", "N", "q\_end"), "b": ("a", "N", "q\_end"), "c": ("a", "N", "q\_end"), " ": ("a", "N", "q\_end")},

#вставляем б

"q\_b.1": {"a": ("a", "L", "q\_b.1"), "b": ("b", "L", "q\_b.1"), "c": ("c", "L", "q\_b.1"), " ": (" ", "R", "q\_b.2")},

"q\_b.2": {"a": ("a", "R", "q\_b.2"), "b": ("b", "R", "q\_b.2"), "c": ("c", "L", "q\_b.3")},

"q\_b.3": {"a": ("b", "N", "q\_end"), "b": ("b", "N", "q\_end"), "c": ("b", "N", "q\_end"), " ": ("b", "N", "q\_end")},

#вставляем с

"q\_c.1": {"a": ("a", "L", "q\_c.1"), "b": ("b", "L", "q\_c.1"), "c": ("c", "L", "q\_c.1"), " ": (" ", "R", "q\_c.2")},

"q\_c.2": {"a": ("a", "R", "q\_c.2"), "b": ("b", "R", "q\_c.2"), "c": ("c", "L", "q\_c.3")},

"q\_c.3": {"a": ("c", "N", "q\_end"), "b": ("c", "N", "q\_end"), "c": ("c", "N", "q\_end"), " ": ("c", "N", "q\_end")},

#если пробел

"q\_void": {"a": ("a", "L", "q\_void"), "b": ("b", "L", "q\_void"), "c": ("c", "L", "q\_void"), " ": (" ", "R", "q\_void.2")},

"q\_void.2": {"a": ("a", "R", "q\_void.2"), "b": ("b", "R", "q\_void.2"), "c": ("c", "L", "q\_void.3")},

"q\_void.3": {"a": ("a", "N", "q\_end"), "b": ("a", "N", "q\_end"), "c": ("a", "N", "q\_end"), " ": ("a", "N", "q\_end")}

}

state = "q\_start"

position = 0

while state != "q\_end":

insert\_symbol, move, next\_state = machine[state][tape[position]]

tape[position] = insert\_symbol

position += 1 if move == "R" else -1 if move == "L" else 0

state = next\_state

print("".join(tape))