**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 |  | Силяев Р.А. |
| Преподаватель |  | Иванов Д.В. |

Санкт-Петербург

2023

## Цель работы

Понять принцип работы машины Тьюринга, научиться писать прототип машины Тьюринга на Python.

## 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}, **которая начинается с символа 'a'.**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | a | c | c | a | b | c | b | a | b | a | a | c | a | b |  |  |  |

**Напишите программу, которая оборачивает исходную строку. Результат работы алгоритма - исходная последовательность символов в обратном порядке.**

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:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | b | a | c | a | a | b | a | b | c | b | a | c | c | a |  |  |  |

 Алфавит (можно расширять при необходимости):

* 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.

6. Нельзя использовать дополнительную ленту, в которую записывается результат.

**Ваша программа должна вывести полученную ленту после завершения работы.**

В отчет включите таблицу состояний. Отдельно кратко опишите каждое состояние, например:

q1 - начальное состояние, которое необходимо, чтобы обнаружить конец строки.

## Выполнение работы

**Таблица состояний:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | ‘a’ | ‘b’ | ‘c’ | ‘!’ | ‘ ’ |
| q1 | "a", R, "q2" | "b", R, "q2" | "c", R, "q2" |  | " ", R, "q1" |
| q2 | "a", R, "q2" | "b", R, "q2" | "c", R, "q2" |  | " ", L, "q3" |
| q3 | "!", N, "q4" | "!", N, "q6" | "!", N, "q8" |  |  |
| q4 | "a", R, "q4" | "b", R, "q4" | "c", R, "q4" | "!", R, "q4" | " ", R, "q5" |
| q5 | "a", R, "q5" | "b", R, "q5" | "c", R, "q5" |  | "a", L, "q10" |
| q6 | "a", R, "q6" | "b", R, "q6" | "c", R, "q6" | "!", R, "q6" | " ", R, "q7" |
| q7 | "a", R, "q7" | "b", R, "q7" | "c", R, "q7" |  | "b", L, "q10" |
| q8 | "a", R, "q8" | "b", R, "q8" | "c", R, "q8" | "!", R, "q8" | " ", R, "q9" |
| q9 | "a", R, "q9" | "b", R, "q9" | "c", R, "q9" |  | "c", L, "q10" |
| q10 | "a", L, "q10" | "b", L, "q10" | "c", L, "q10" |  | " ", L, "q11" |
| q11 | "!", N, "q4" | "!", N, "q6" | "!", N, "q8" | "!", L, "q11" | " ", R, "q12" |
| q12 |  |  |  | " ", R, "q12" | " ", N, "q13" |

****Описание состояний:****

* q1 – перемещение к первой букве
* q2 – перемещение к последнему символу
* q3 – замена последнего символа на «!» , если ячейка содержит «a», то вызывается q4, если «b», то q6, если «с», то q8
* q4 – перемещение к первому символу перевернутой строки, вызов q5
* q5 – перемещение к первому пробелу после строки, запись символа «a», переход к концу инвертированной строки
* q6 – аналогично q4, но вызов q7
* q7 – аналогично q5, но запись «b»
* q8 - аналогично q4, но вызов q9
* q9 -аналогично q5, но запись «с»
* q10 – перемещение от конца инвертированной строки к концу начальной
* q11 – проход по строке, при нахождении символов «a», «b», «c» возврат к начальный состояниям, в противном случае переход к началу строки.
* q12 – удаление «!»

Таблица для машины Тьюринга в программе задается словарем *table*, в котором по ключу(фазе) задается еще один словарь, в нем ключами уже является алфавит машины Тьюринга. Подаваемая на вход строка сохраняется в массив arr, к которому с двух сторон прибавляются в пробелы для правильной работы программы. После начинает обрабатываться лента: меняется значение в ячейке, происходит перемещение указателя, обновляется фаза. Программа завершается, когда фаза становится равной q13.

Разработанный программный код см. в приложении А.

## Тестирование

Результаты тестирования содержатся в таблице 1.

Таблица 1 – Результаты тестирования

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Входные данные | Выходные данные | Комментарии |
|  | abbbcccca | accccbbba |  |
|  | ababcbcbabcab | bacbabcbcbaba |  |
|  | aaaaaacccccc | ccccccaaaaaa |  |

## Выводы

В результате работы были изучены принципы работы машины Тьюринга, а также разработана программа, которая инвертирует строку, на основе принципов работы машины Тьюринга.

# Приложение А Исходный код программы

Название файла: main.py

L = -1

R = +1

N = 0

place = list(" "\*15)

arr = place + list(input()) + place

state = "q1"

ind = 0

table = {"q1": {"a": ["a", R, "q2"], "b": ["b", R, "q2"], "c":

["c", R, "q2"], " ": [" ", R, "q1"]},

"q2": {"a": ["a", R, "q2"], "b": ["b", R, "q2"], "c": ["c", R,

"q2"], " ": [" ", L, "q3"]},

"q3": {"a": ["!", N, "q4"], "b": ["!", N, "q6"], "c": ["!", N,

"q8"]},

"q4": {"a": ["a", R, "q4"], "b": ["b", R, "q4"], "c": ["c", R,

"q4"], "!": ["!", R, "q4"], " ": [" ", R, "q5"]},

"q5": {"a": ["a", R, "q5"], "b": ["b", R, "q5"], "c": ["c", R,

"q5"], " ": ["a", L, "q10"]},

"q6": {"a": ["a", R, "q6"], "b": ["b", R, "q6"], "c": ["c", R,

"q6"], "!": ["!", R, "q6"], " ": [" ", R, "q7"]},

"q7": {"a": ["a", R, "q7"], "b": ["b", R, "q7"], "c": ["c", R,

"q7"], " ": ["b", L, "q10"]},

"q8": {"a": ["a", R, "q8"], "b": ["b", R, "q8"], "c": ["c", R,

"q8"], "!": ["!", R, "q8"], " ": [" ", R, "q9"]},

"q9": {"a": ["a", R, "q9"], "b": ["b", R, "q9"], "c": ["c", R,

"q9"], " ": ["c", L, "q10"]},

"q10": {"a": ["a", L, "q10"], "b": ["b", L, "q10"], "c": ["c",

L, "q10"], " ": [" ", L, "q11"]},

"q11": {"a": ["!", N, "q4"], "b": ["!", N, "q6"], "c": ["!", N,

"q8"], "!": ["!", L, "q11"], " ": [" ", R, "q12"]},

"q12": {"!": [" ", R, "q12"], " ": [" ", N, "q13"]}}

while(state!="q13"):

dict = table[state][arr[ind]]

arr[ind] = dict[0]

ind += dict[1]

state = dict[2]

print(\*arr, sep='')