**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(ка) гр. 3343 |  | Гельман П.Е. |
| Преподаватель |  | Иванов Д.В. |

Санкт-Петербург

2023

## Цель работы

Освоить принцип работы машины Тьюринга, развить навыки разработки алгоритмов для работы автомата и создать программу на языке Python, которая отражает этот механизм.

## Задание

**Вариант 4**

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 | ‘-‘ | ‘ ‘ |
| q0 | a; R; q0 | b; R; q0 | c; R; q0 | - | ‘ ‘; L; q1 |
| q1 | a; L; q2 | b; L; q2 | c; L; q2 | - | ‘ ‘; L; q1 |
| q2 | -; R; q3 | -; R; q4 | -; R; q5 | -; L; q2 | ‘ ‘; R; q7 |
| q3 | a; R; q3 | b; R; q3 | c; R; q3 | -; R; q3 | a; L; q6 |
| q4 | a; R; q4 | b; R; q4 | c; R; q4 | -; R; q4 | b; L;q6 |
| q5 | a; R; q5 | b; R; q5 | c; R; q5 | -; R; q5 | c; L; q6 |
| q6 | a; L; q6 | b; L; q6 | c; L; q6 | -; N; q2 | - |
| q7 | a; R; q7 | b; R; q7 | c; L; q7 | ‘ ‘; R; q7 | ‘ ‘; N; qT |

q0 – начальное состояние, которое необходимо, чтобы обнаружить конец строки

q1 – состояние, в котором пропускается последняя буква строки

q2 – начинается анализ строки с конца, та или иная буква заменяется на знак “-“

q3 – если была произведена замена буквы “a”, то автомат переходит в это состояние и идет влево до пробела, который заменяется на эту букву

q4 – аналогично ищется пробел после текущей строки, который будет заменен на букву “b”

q5 – та же операция происходит с буквой “c”, если она была заменена на знак “-“

q6 – после того как та или иная буква была перенесена в конец строки, автомат проходится обратно вправо до тех пор, пока не встретит “-“, когда знак нашелся, мы переключаемся обратно в состояние поиска новой буквы q2

q7 – когда автомат прошелся по всем буквам справа налево и встретил пробел, он переходит в это состояние и удаляет все ненужные “-“. Таким образом, получается перевернутая строка, смещенная на количество символов в ней минус один.

Таблица состояний в программе представляет собой словарь, ключами в котором являются строки ‘q0’, ‘q1’…’q7’, а значения – словари, где ключ – символ алфавита машины Тьюринга, значение – кортеж с соответствующими операциями.

Далее пользователь вводит нужную строку, добавляется некоторое количество пробелов, чтобы машина Тьюринга работала корректно.

В цикле while реализована работа самой машины Тьюринга, происходит замена символов, изменение состояния и положения на ленте, что приводит к нужному решению задачи.

## Тестирование

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Входные данные | Выходные данные | Комментарии |
|  | abcabc | cbacba | Выходные данные задачи соответствуют ожиданиям |
|  | abacbbc | cbbcaba | Выходные данные задачи соответствуют ожиданиям |

## Выводы

В ходе лабораторной работы я изучила механизм машины Тьюринга, научилась составлять таблицы состояний для разных задач, создавать аналог машины Тьюринга на языке Python, который использует словари и цикл while.

# Приложение А Исходный код программы

Название файла: main.py

R, L, N = 1, -1, 0

t = {

'q0':{'a': ('a', R, 'q0'),

'b': ('b', R, 'q0'),

'c': ('c', R, 'q0'),

' ': (' ', L, 'q1')},

'q1':{'a': ('a', L, 'q2'),

'b': ('b', L, 'q2'),

'c': ('c', L, 'q2'),

' ': (' ', L, 'q1')}, # скип ласт букву

'q2': {'a': ('-', R, 'q3'),

'b': ('-', R, 'q4'),

'c': ('-', R, 'q5'),

'-' : ('-', L, 'q2'),

' ': (' ', R, 'q7')},

#a

'q3':{'a': ('a', R, 'q3'),

'b': ('b', R, 'q3'),

'c': ('c', R, 'q3'),

'-': ('-', R, 'q3'),

' ': ('a', L, 'q6')},

#b

'q4': {'a': ('a', R, 'q4'),

'b': ('b', R, 'q4'),

'c': ('c', R, 'q4'),

'-': ('-', R, 'q4'),

' ': ('b', L, 'q6')},

#c

'q5': {'a': ('a', R, 'q5'),

'b': ('b', R, 'q5'),

'c': ('c', R, 'q5'),

'-': ('-', R, 'q5'),

' ': ('c', L, 'q6'),

},

'q6': {'a': ('a', L, 'q6'),

'b': ('b', L, 'q6'),

'c': ('c', L, 'q6'),

'-': ('-', N, 'q2')},

'q7': {'a' : ('a', R, 'q7'),

'b' : ('b', R, 'q7'),

'c' : ('c', R, 'q7'),

'-': (' ', R, 'q7'),

' ': (' ',N, 'qT')}

}

s = input()

lenta = list(' '\*100 + s + ' '\*100)

state = 'q0'

index = 0

states = [state]

while state != 'qT':

curr = lenta[index]

next = t[state][curr]

lenta[index] = next[0]

index += next[1]

state = next[2]

states += [state]

s = ''

for i in lenta:

s += i

print(s)