**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 и конечные автоматы

|  |  |  |
| --- | --- | --- |
| Студент гр. 3343 |  | Малиновский А. А. |
| Преподаватель |  | Иванов Д.В. |

Санкт-Петербург

2023

## Цель работы

Изучить, как работает машина Тьюринга, реализовать её механизм на языке Python. Освоить знания на практике, написав программу по данному заданию.

## 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'.

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.

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.

## Выполнение работы

1. Реализовывается механизм машины Тьюринга в словаре *table*.

Таблица состояний представлена в табл. 1.

Таблица 1 – Таблица состояний

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | a | b | c | ‘ ’ (пробел) | ‘ ’ |
| 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 | a,L,q4 | b,L,q4 | c,L,q4 | - | - |
| q4 | x,R,q5 | x,R,q6 | x,R,q6 | - | - |
| q5 | a,R,q9 | b,R,q9 | c,R,q9 | - | x,R,q5 |
| q6 | a,R,q10 | b,R,q10 | c,R,q10 | - | x,R,q6 |
| q7 | a,R,q11 | b,R,q11 | c,R,q11 | - | x,R,q7 |
| q8 | a,L,q8 | b,L,q8 | c,L,q8 | - | x,L,q12 |
| q9 | a, R,q9 | b,R,q9 | c,R,q9 | a,R,q9 | - |
| q10 | a,R,q10 | b,R,q10 | c,R,q10 | b,R,q10 | - |
| q11 | a,R,q11 | b,R,q11 | c,R,q11 | c,R,q11 | x,L,q12 |
| q12 | x,R,q5 | x,R,q6 | x,R,q7 | ‘ ‘,R,q13 | x,L,q12 |
| q13 | a,N,qT | b,N,qT | c,N,qT | - | ‘ ‘,R,q13 |

Для работы машины Тьюринга алфавит расширяется. Добавляется символ p, обозначающий то, что во введённой строке этот элемент уже обработан.

Описание состояний:

q1 – начальное положение, ищем конец строки.

q2 - в этом состоянии отмечаем, что первый пробел был пройден.

q3 - в данном состоянии мы находимся на последнем символе исходной строки, то есть на первом символе обработанной строки.

q4 - определяем, какой символ будет следующим в преобразованной строке, и записываем вместо него «x».

q5 - следующий символ, который необходимо записать это «a», идем по строке вправо.

q6 - следующий символ, который необходимо записать это «b», идем по строке вправо.

q7 - следующий символ, который необходимо записать это «c», идем по строке вправо.

q8 - в этом состоянии осуществляется поиск следующего символа, который нужно записать в обернутую строку.

q9 - доходим до конца строки и записываем символ «a» в конец преобразованной строки.

q10 - доходим до конца строки и записываем символ «b» в конец преобразованной строки.

q11 - доходим до конца строки и записываем символ «c» в конец преобразованной строки.

q12 - записываем вместо символа исходной строки «x», чтобы отметить, что он будет добавлен в конец измененной строки.

q13 - все элементы исходной строки были записаны в обратном порядке; вместо символов «x» записываем пробелы.

qT – конечное состояние, работа программы завершена.

2. Создаётся и записывается в переменную *table* список из введённой строки, добавляется 10 пробелов слева и 10 пробелов справа для исправной работы программы, даже если введённая строка полностью без пробелов.

3. В цикле *while* реализуется работа машины Тьюринга, пока она не переходит в конечное состояние (*qT*). Обновляются значение видимого символа, позиция на ленте и состояние машины Тьюринга.

4. Выводится лента после обработки с помощью *print*.

## Тестирование

Результаты тестирования представлены в табл. 2.

Таблица 2 – Результаты тестирования

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Входные данные | Выходные данные | Комментарии |
|  | abcbccccb | bccccbcba | Программа работает корректно. |
|  | aaaaaaaabaac | caabaaaaaaaa | Программа работает корректно. |
| 1. 3. | acacabac | cabacaca | Программа работает корректно. |

## Выводы

Была освоена машина Тьюринга и реализован её механизм на языке Python.

Разработана программа, содержащая реализацию автомата, описанную в словаре, согласно заданию, и осуществляющая разворот строки.

# Приложение А Исходный код программы

Название файла: main.py

L, R, N = -1, 1, 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': ['a', L, 'q4'], 'b': ['b', L, 'q4'], 'c': ['c', L, 'q4']},

'q4': {'a': ['x', R, 'q5'], 'b': ['x', R, 'q6'], 'c': ['x', R, 'q7']},

'q5': {'a': ['a', R, 'q9'], 'b': ['b', R, 'q9'], 'c': ['c', R, 'q9'], 'x': ['x', R, 'q5']},

'q6': {'a': ['a', R, 'q10'], 'b': ['b', R, 'q10'], 'c': ['c', R, 'q10'], 'x': ['x', R, 'q6']},

'q7': {'a': ['a', R, 'q11'], 'b': ['b', R, 'q11'], 'c': ['c', R, 'q11'], 'x': ['x', R, 'q7']},

'q8': {'a': ['a', L, 'q8'], 'b': ['b', L, 'q8'], 'c': ['c', L, 'q8'], 'x': ['x', L, 'q12']},

'q9': {'a': ['a', R, 'q9'], 'b': ['b', R, 'q9'], 'c': ['c', R, 'q9'], ' ': ['a', L, 'q8']},

'q10': {'a': ['a', R, 'q10'], 'b': ['b', R, 'q10'], 'c': ['c', R, 'q10'], ' ': ['b', L, 'q8']},

'q11': {'a': ['a', R, 'q11'], 'b': ['b', R, 'q11'], 'c': ['c', R, 'q11'], ' ': ['c', L, 'q8']},

'q12': {'a': ['x', R, 'q5'], 'b': ['x', R, 'q6'], 'c': ['x', R, 'q7'], 'x': ['x', L, 'q12'], ' ': [' ', R, 'q13']},

'q13': {'x': [' ', R, 'q13'], 'a': ['a', N, 'qT'], 'b': ['b', N, 'qT'], 'c': ['c', N, 'qT']}

}

memory = list(input())

buf = [' 'for i in range(10)]

memory =buf+memory+buf

state = 'q1'

index = 0

states = [state]

while state != 'qT':

current\_symbol = memory[index]

future = table[state][current\_symbol]

memory[index] = future[0]

index += future[1]

state = future[2]

states += [state]

print(\*memory, sep='')