**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 |  | Жучков О.Д. |
| Преподаватель |  | Иванов Д.В. |

Санкт-Петербург

2023

## Цель работы

Цель работы заключается в изучении принципа работы Машины Тьюринга и реализации данного абстрактного исполнителя на языке Python. Небходимо составить алгоритм для Машины Тьюринга для выполнения некоторой задачи по обработке ввода.

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.

Ваша программа должна вывести полученную ленту после завершения работы.

## Выполнение работы

Для реализации Машины Тьюринга в программе создан класс TuringMachine, который при инициализации принимает таблицу состояний, изначальное и конечное состояния. Таблица состояний подается в виде списка форматированных строк. В поля данного класса входят: алфавит, размер алфавита, двумерный список состояний, количество состояний, конечное состояние, лента, с которой работает строка, текущее состояние машины, текущая ячейка ленты. В классе реализован метод step, который выполняет обработку текущего состояния машины и переход на следующее. Метод compute принимает изначальное состояние ленты и выполняет алгоритм машины (вызывает метод step, пока машина не перейдет в конечное состояние). Также в процессе выполнения в список сохраняются состояния, в которые входила машина, что может быть полезно для проверки корректной работы алгоритма.

Для упрощения алгоритма в алфавит добавлен вспомогательный символ X, который используется для обозначения начала и конца строки. Перевернутая строка строится справа от конца исходной строки. Для переворота строки машиной выполняются следующие шаги:

1. Движение вправо и поиск начала строки (первый не пробельный символ) и пометка его символом X слева.
2. Движение вправо и поиск конца строки и пометка его символом X справа.
3. Движение влево, поиск и определение самого близкого к концу строки символа (не пробела). Символ запоминается в состоянии и заменяется пробелом.
4. Движение вправо, поиск конца новой строки. В пустое место на конце новой строки записывается считанный символ.
5. Предыдущие два шага (3, 4) повторяются, пока все символы исходной строки не будут перенесены в новую. Оставшиеся вспомогательные символы X удаляются.

Таблица состояний для Машины Тьюринга представлена в табл. 1.

Таблица 1 – Результаты тестирования

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | a | b | c | \_ | X |
| q1 | a L q2 | b L q2 | c L q2 | \_ R q1 |  |
| q2 |  |  |  | X R q3 |  |
| q3 | a R q3 | b R q3 | c R q3 | X L q5 |  |
| q4 | A L q4 | B L q4 | C L q4 |  | X L q5 |
| q5 | \_ R q6 | \_ R q7 | \_ R q8 | \_ L q5 | \_ N q13 |
| q6 |  |  |  | \_ R q6 | X R q9 |
| q7 |  |  |  | \_ R q7 | X R q10 |
| q8 |  |  |  | \_ R q8 | X R q11 |
| q9 | a R q9 | b R q9 | c R q9 | a L q4 |  |
| q10 | a R q10 | b R q10 | c R q10 | b L q4 |  |
| q11 | a R q11 | b R q11 | c R q11 | c L q4 |  |
| q12 |  |  |  | \_ R q12 | \_ N q13 |

## Тестирование

Результаты тестирования представлены в табл. 2.

Таблица 2 – Результаты тестирования

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Входные данные | Выходные данные | Комментарии |
|  | abbcab | bacbba | Вывод соответствует ожидаемому |
|  | baba | abab | Вывод соответствует ожидаемому |

## Выводы

В ходе выполнения работы были изучены и реализованы на языке программирования Python принципы работы Машины Тьюринга. Также была составлена таблица состояний для Машины Тьюринга для переворота строки.

# Приложение А Исходный код программы

Название файла: main.py

class TuringMachine:

def \_\_init\_\_(self, statetable, start, end):

self.alphabet = statetable[0].split("|")

self.state\_count = len(statetable)

self.alphabet\_size = len(self.alphabet)

states = dict()

for i in range(1, self.state\_count):

state = dict()

state\_list = statetable[i].split("|")

for j in range(self.alphabet\_size):

state[self.alphabet[j]] = state\_list[j+1].split(";")

states[state\_list[0]] = state

self.states = states

self.endstate = end

self.cur\_state = start

self.cur\_pos = 0

self.string = " "

def step(self):

if self.cur\_pos == len(self.string):

self.string.append(" ")

action=self.states[self.cur\_state][self.string[self.cur\_pos]]

self.string[self.cur\_pos] = action[0]

if action[1] == "L":

self.cur\_pos -= 1

elif action[1] == "R":

self.cur\_pos += 1

self.cur\_state = action[2]

def compute(self, string):

self.string = list(string)

log = [self.cur\_state]

while self.cur\_state != self.endstate:

self.step()

log.append(self.cur\_state)

return ''.join(self.string), ' '.join(log)

table = ["a|b|c| |X",

"q1|a;L;q2|b;L;q2|c;L;q2| ;R;q1|null",

"q2|null|null|null|X;R;q3|null",

"q3|a;R;q3|b;R;q3|c;R;q3|X;L;l2|null",

"l1|a;L;l1|b;L;l1|c;L;l1|null|X;L;l2",

"l2| ;R;ra1| ;R;rb1| ;R;rc1| ;L;l2| ;N;q4",

"ra1|null|null|null| ;R;ra1|X;R;ra2",

"rb1|null|null|null| ;R;rb1|X;R;rb2",

"rc1|null|null|null| ;R;rc1|X;R;rc2",

"ra2|a;R;ra2|b;R;ra2|c;R;ra2|a;L;l1|null",

"rb2|a;R;rb2|b;R;rb2|c;R;rb2|b;L;l1|null",

"rc2|a;R;rc2|b;R;rc2|c;R;rc2|c;L;l1|null",

"q4|null|null|null| ;R;q4| ;N;end"

]

tm = TuringMachine(table, "q1", "end")

print(tm.compute(input())[0]) image