**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. 3341 |  | Самокрутов А.Р. |
| Преподаватель |  | Иванов Д.В. |

Санкт-Петербург

2023

## Цель работы

Целью лабораторной работы является изучение работы конечных автоматов, в частности — машины Тьюринга. Для этого требуется написать программу на языке программирования Python, которая моделирует работу машины Тьюринга и выполняет определённую задачу.

## Задание

Вариант 2

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}.

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

***Напишите программу, которая заменяет в исходной строке символ, идущий после последних двух встретившихся символов 'a', на предшествующий им символ(гарантируется, что это не пробел). Наличие в строке двух подряд идущих символов 'a' гарантируется.***

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:

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

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.**

В отчете предоставьте таблицу состояний. Отдельно кратко опишите каждое состояние, например:

q1 - начальное состояние, которое необходимо, чтобы найти первый встретившийся символ ‘a’.

## Выполнение работы

Объявляются константы *R*, *L* и *N* соответственно равные *1*, *-1* и *0*. В данной работе лента машины Тьюринга будет рассматриваться как список, образованный элементами строки, считанной программой из потока ввода. Движению вправо по ленте будет соответствовать инкрементация индекса элемента списка, движению влево — декрементация, остановке на месте — отсутствие изменения индекса. Далее инициализируются константы *CHANGE\_TO*, *MOVE\_TO* и *NEXT\_STATE* со значениями *0*, *1* и *2*. Их смысл будет раскрыт далее в тексте.

Создаётся функция *turing\_machine()* с аргументами program (словарь, ключи которого являются состояниями программы, а значения — словари типа элемент ленты — кортеж из трёх элементов: первый элемент — какой символ необходимо записать в текущую ячейку, второй — в каком направлении двигаться, третий — в какое состояние перейти. Индексу каждого из этих элементов-действий соответствует переменная *CHANGE\_TO*, *MOVE\_TO* и *NEXT\_STATE*), *tape* (массив символов, полученный из строки, поданной в поток ввода), *start\_state* (начальное состояние машины), *current\_cell* (номер клетки, в которой размещён автомат), *end\_state* (конечное состояние).

Функция *turing\_machine(program, tape, start\_state, current\_cell, end\_state)*.

Значение *start\_state* — начального состояния — записывается в переменную *current\_state* — текущее состояние. В бесконечном цикле объявляется переменная *rows* со значением, полученным из словаря *program* по ключу *current\_state* — словарь типа символ алфавита — кортеж из элементов-действий автомата; объявляется переменная *current\_row*, ей присваивается значение словаря rows по ключу *tape[current\_cell]* (символ в ленте, на который в данный момент указывает автомат); значение текущей ячейки ленты изменяется на *current\_row[CHANGE\_TO]*; если в *current\_row[MOVE\_TO]* находится *0*, т.е. *N*, то цикл прерывается — программа останавливается; к значению *current\_cell* прибавляется значение *current\_row[MOVE\_TO]* — *1* или *-1*, т.е. *R* или *L*; переменной *current\_state* присваивается значение *current\_row[NEXT\_STATE]*.

Функция возвращает строку, полученную из списка *tape* методом *join()*.

Далее создаётся переменная *program*, в которой описывается программа машины Тьюринга. Ей соответствует следующая таблица:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 'a' | 'b' | 'c' | ' ' |
| q0 | 'a'; R; q1 | 'b'; R; q1 | 'c'; R; q1 | ' '; R; q0 |
| q1 | 'a'; R; q1 | 'b'; R; q1 | 'c'; R; q1 | ' '; L; q2 |
| q2 | 'a'; L; q3 | 'b'; L; q2 | 'c'; L; q2 |  |
| q3 | 'a'; L; q4 | 'b'; L; q2 | 'c'; L; q2 |  |
| q4 | 'a'; R; q5 | 'b'; R; q8 | 'c'; R; q11 |  |
| q5 | 'a'; R; q6 |  |  |  |
| q6 | 'a'; R; q7 |  |  |  |
| q7 | 'a'; N; q14 | 'a'; N; q14 | 'a'; N; q14 | 'a'; N; q14 |
| q8 | 'a'; R; q9 |  |  |  |
| q9 | 'a'; R; q10 |  |  |  |
| q10 | 'b'; N; q14 | 'b'; N; q14 | 'b'; N; q14 | 'b'; N; q14 |
| q11 | 'a'; R; q12 |  |  |  |
| q12 | 'a'; R; q13 |  |  |  |
| q13 | 'c'; N; q14 | 'c'; N; q14 | 'c'; N; q14 | 'c'; N; q14 |

Описание каждого состояния:

q0 (*'start'*) — начальное состояние. Автомат движется вправо пока не встретит символ, отличный от пробела (пустой клетки). При встрече такого символа автомат движется вправо и переходит в состояние q1.

q1 (*'skip through'*) — автомат движется по вправо по строке пока не дойдёт до её конца. Считав символ пробела, автомат сдвигается на клетку левее и переходит в состояние q2.

q2 (*'step back'*) — автомат движется влево по строке пока не встретит символ 'a'. В этом случае он сдвигается влево и переходит в состояние q3.

q3 (*'found first a'*) — автомат уже нашёл символ 'a' в предыдущей ячейке. Если в текущей клетке символ 'a', то машина переходит в состояние q4, иначе — в q2, и сдвигается влево.

q4 (*'found second a'*) — автомат уже нашёл второй символ 'a' в предыдущей ячейке. Если автомат считал символ 'a', 'b' или 'c', то он переходит в состояние q5, q8, или q11 соответственно, двигаясь вправо.

q5 (*'read an a'*) — на предыдущем шаге автомат прочитал символ 'a' и перемещается в направлении к символу справа от 'aa'. Автомат перемещается вправо и переходит в состояние q6.

q6 (*'read an a, skip'*) — автомат передвигается через второй из двух символов 'a'. Он перемещается вправо и переходит в состояние q7.

q7 (*'replace with a'*) — автомат заменяет содержимое текущей ячейки на 'a' и останавливается, переходя в состояние q14.

q8 (*'read a b'*) — на предыдущем шаге автомат прочитал символ 'b' и перемещается в направлении к символу справа от 'aa'. Автомат перемещается вправо и переходит в состояние q9.

q9 (*'read a b, skip'*) — автомат передвигается через второй из двух символов 'a'. Он перемещается вправо и переходит в состояние q10.

q10 (*'replace with b'*) — автомат заменяет содержимое текущей ячейки на 'b' и останавливается, переходя в состояние q14.

q11 (*'read a c'*) — на предыдущем шаге автомат прочитал символ 'c' и перемещается в направлении к символу справа от 'aa'. Автомат перемещается вправо и переходит в состояние q12.

q12 (*'read a c, skip'*) — автомат передвигается через второй из двух символов 'a'. Он перемещается вправо и переходит в состояние q10.

q13 (*'replace with c'*) — автомат заменяет содержимое текущей ячейки на 'b' и останавливается, переходя в состояние q14.

q14 (*'end'*) — конечное состояние автомата.

Разработанный программный код см. в приложении А.

## Тестирование

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Входные данные | Выходные данные | Комментарии |
|  | abcabcaabc | abcabcaacc |  |
|  | baa | baa | Проверка случая, когда символ, который нужно заменить, находится вне исходной строки. |
|  | aacaacaab | Aacaacaac | Проверка корректной работы программы со строкой, в которой 'aa' встречается более одного раза. |

## Выводы

Была разработана программа на языке программирования Python, симулирующая работу машины Тьюринга. Была описана программа машины Тьюринга, с помощью которой автомат способен обработать строку символов определённым образом.

# Приложение А Исходный код программы

Название файла: lab.py

R = +1

L = -1

N = 0

CHANGE\_TO = 0

MOVE\_TO = 1

NEXT\_STATE = 2

def turing\_machine(program, tape, start\_state, current\_cell, end\_state):

current\_state = start\_state

while True:

rows = program[current\_state]

current\_row = rows[tape[current\_cell]]

tape[current\_cell] = current\_row[CHANGE\_TO]

if not current\_row[MOVE\_TO]:

break

current\_cell += current\_row[MOVE\_TO]

current\_state = current\_row[NEXT\_STATE]

return ''.join(tape)

program = {'start': {'a': ('a', R, 'skip through'), 'b': ('b', R, 'skip through'), 'c': ('c', R, 'skip through'), ' ': (' ', R, 'start')},

'skip through': {'a': ('a', R, 'skip through'), 'b': ('b', R, 'skip through'), 'c': ('c', R, 'skip through'), ' ': (' ', L, 'step back')},

'step back': {'a': ('a', L, 'found first a'), 'b': ('b', L, 'step back'), 'c': ('c', L, 'step back')},

'found first a': {'a': ('a', L, 'found second a'), 'b': ('b', L, 'step back'), 'c': ('c', L, 'step back')},

'found second a': {'a': ('a', R, 'read an a'), 'b': ('b', R, 'read a b'), 'c': ('c', R, 'read a c')},

'read an a': {'a': ('a', R, 'read an a, skip')},

'read an a, skip': {'a': ('a', R, 'replace with a')},

'replace with a': {'a': ('a', N, 'end'), 'b': ('a', N, 'end'), 'c': ('a', N, 'end'), ' ': ('a', N, 'end')},

'read a b': {'a': ('a', R, 'read a b, skip')},

'read a b, skip': {'a': ('a', R, 'replace with b')},

'replace with b': {'a': ('b', N, 'end'), 'b': ('b', N, 'end'), 'c': ('b', N, 'end'), ' ': ('b', N, 'end')},

'read a c': {'a': ('a', R, 'read a c, skip')},

'read a c, skip': {'a': ('a', R, 'replace with c')},

'replace with c': {'a': ('c', N, 'end'), 'b': ('c', N, 'end'), 'c': ('c', N, 'end'), ' ': ('c', N, 'end')}}

tape = list(str(input()) + ' ')

print(turing\_machine(program, tape, 'start', 0, 'end')[:-1])