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. 2381 _____ Кузнецов И.И.

Преподаватель _____ Шевская Н.В.

Санкт-Петербург
2022

Цель работы.

Изучить механизм работы машины, реализовать алгоритм для решения поставленной задачи при помощи машины Тьюринга.

Задание.

Вариант 3.

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}.

Напишите программу, которая заменяет в исходной строке символ, предшествующий первому встретившемуся символу ‘c' на символ, следующий за первым встретившимся символом ‘a'. Если первый встретившийся символ ‘a' в конце строки, то используйте его в качестве заменяющего.

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:

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.

Выполнение работы.

Таблица 1 - Программа для машины Тьюринга

Состояние	a	b	c	‘ ‘(пробел)
start	a,R,found_a	b,R,start	c,R,found_c	‘ ‘,R,start
found_a	a,L,found_a_a	b,L,found_a_b	c,L,found_a_c	-
found_a_c	c,N,end	c,N,end	c,N,end	-
found_a_b	a,R,found_a_b	b,R,found_a_b	c,L,found_a_b_c	‘ ‘,R,found_a_b
found_a_a	a,R,found_a_a	b,R,found_a_a	c,L,found_a_b_c	‘ ‘,R,found_a_a
found_a_b_c	b,N,end	b,N,end	b,N,end	b,N,end
found_a_a_c	a,N,end	a,N,end	a,N,end	a,N,end
found_c	a,R,found_c_a	b,R,found_c	c,R,found_c	‘ ‘,R,found_c
found_c_a	a,L,found_c_a_a	b,L,found_c_a_b	c,L,found_c_a_c	‘ ‘L,found_c_a_a
found_c_a_b	a,L,found_c_a_b	b,L,found_c_a_b	c,L,found_c_a_b	‘ ‘,L,found_c_a_b_start
found_c_a_c	a,L,found_c_a_c	b,L,found_c_a_c	c,L,found_c_a_c	‘ ‘,L,found_c_a_c_start
found_c_a_a	a,L,found_c_a_a	b,L,found_c_a_a	c,L,found_c_a_a	‘ ,L,found_c_a_a_start
found_c_a_b_start	a,R,found_c_a_b_start	a,R,found_c_a_b_start	a,R,found_c_a_b_start_c	a,R,found_c_a_b_start
found_c_a_a_start	a,R,found_c_a_a_start	a,R,found_c_a_a_start	a,R,found_c_a_a_start_c	a,R,found_c_a_a_start
found_c_a_c_start	a,R,found_c_a_c_start	a,R,found_c_a_c_start	a,R,found_c_a_c_start_c	a,R,found_c_a_c_start
found_c_a_b_start_c	b,N,end	b,N,end	b,N,end	b,N,end
found_c_a_c_start_c	c,N,end	c,N,end	c,N,end	c,N,end
found_c_a_a_start_c	a,N,end	a,N,end	a,N,end	a,N,end

Описание положений:

"start": начальное положение, ищем “b” или “c”

"found_a": нашли “a”, смотрим какой символ идет после

"found_a_c": после “a” идет “c”, меняем символ и заканчиваем раюоту

"found_a_b": после “a” идет “b” ищем “c”

"found_a_a": после “a” идет “a” ищем “c”

"found_a_b_c": нашли “c” в заданной последовательности

"found_a_a_c": нашли “c” в заданной последовательности

"found_c": первым ключевым найденным символом был “c”, ищем “a”

"found_c_a": нашли “a”, смотрим следующий символ

"found_c_a_b": следующий символ “b” возвращаемся в начало

"found_c_a_a": следующий символ “a” возвращаемся в начало

"found_c_a_c": следующий символ “c” возвращаемся в начало

"found_c_a_a_start": вернулись в начало с заданной последовательностью ищем “c”, после нахождения переключаемся на предыдущий символ

"found_c_a_b_start": вернулись в начало с заданной последовательностью ищем “c”, после нахождения переключаемся на предыдущий символ

"found_c_a_c_start": вернулись в начало с заданной последовательностью, ищем “c”, после нахождения переключаемся на предыдущий символ

"found_c_a_c_start_c": меняем символ на “c”, заканчиваем работу программу

"found_c_a_b_start_c": меняем символ на “b”, заканчиваем работу программу

"found_c_a_a_start_c": меняем символ на “a”, заканчиваем работу программу

“end”: конечное положение, работа программы закончена.

В переменных R,L,N находится сдвиг по ленте, соответственно 1,-1,0.

В переменной mem хранится введенная строка, в переменной state начальное положение, в ind текущий индекс строки, с которой мы взаимодействуем.

Запуская цикл до того момента, пока state не примет значения end, обращаемся по ключам к ячейкам таблицы, записывая новый символ symbol, шаг автомата ind, и новое состояние state. Записываем новый символ на ленту, обновляем текущее состояние, делаем шаг. Итоговая лента выводится на экран.

Разработанный программный код см. в приложении А.

Тестирование.

Результаты тестирования представлены в табл. 2.

Выводы.

Изучен механизм работы машины Тьюринг. Создан алгоритм решения поставленной задачи по замене необходимых символов в строке на ленте. Разработана программа на языке программирования Python, имитирующая работу машины Тьюринга.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.py

```
R, N, L = 1, 0, -1
table = {
    "start": {'a': ['a', R, "found_a"], 'c': ['c', R, "found_c"], '
': [' ', R, "start"], 'b': ['b', R, "start"]},
    "found_a": {"a": ["a",L,"found_a_a"], "b" : ['b',L,"found_a_b"],
'c': ['c', L, "found_a_c"]},
    "found_a_c": {"a": ["c",N,"end"], "b" : ['c',N,"end"], ' ': ['c',
N, "end"], 'c': ['c', N, "end"]},
    "found_a_b":      {"a":      ["a",R,"found_a_b"],      "b"      :
['b',R,"found_a_b"],  '  ':  ['  ',  R,  "found_a_b"],  'c':  ['c',  L,
"found_a_b_c"]},
    "found_a_a":      {"a":      ["a",R,"found_a_a"],      "b"      :
['b',R,"found_a_a"],  '  ':  ['  ',  R,  "found_a_a"],  'c':  ['c',  L,
"found_a_a_c"]},
    "found_a_b_c": {"a": ["b",N,"end"], "b" : ['b',N,"end"], ' ':
['b', N, "end"], 'c': ['b', N, "end"]},
    "found_a_a_c": {"a": ["a",N,"end"], "b" : ['a',N,"end"], ' ':
['a', N, "end"], 'c': ['a', N, "end"]},
    "found_c": {"a": ["a",R,"found_c_a"], "b" : ['b',R,"found_c"], '
': [' ', R, "found_c"], 'c': ['c', R, "found_c"]},
    "found_c_a":     {"a":     ["a",L,"found_c_a_a"],     "b"     :
['b',L,"found_c_a_b"],  '  ':  ['  ',  L,  "found_c_a_a"],  'c':  ['c',  L,
"found_c_a_c"]},
    "found_c_a_b":     {"a":     ["a",L,"found_c_a_b"],     "b"     :
['b',L,"found_c_a_b"], ' ': [' ', L, "found_c_a_b_start"], 'c': ['c', L,
"found_c_a_b"]},
    "found_c_a_c":     {"a":     ["a",L,"found_c_a_c"],     "b"     :
['b',L,"found_c_a_c"], ' ': [' ', L, "found_c_a_c_start"], 'c': ['c', L,
"found_c_a_c"]},
    "found_c_a_a":     {"a":     ["a",L,"found_c_a_a"],     "b"     :
['b',L,"found_c_a_a"], ' ': [' ', L, "found_c_a_a_start"], 'c': ['c', L,
"found_c_a_a"]},
    "found_c_a_b_start":  {"a":  ["a",R,"found_c_a_b_start"],  "b"  :
['b',R,"found_c_a_b_start"], ' ': [' ', R, "found_c_a_b_start"], 'c': ['c',
L, "found_c_a_b_start_c"]},
    "found_c_a_a_start":  {"a":  ["a",R,"found_c_a_a_start"],  "b"  :
['b',R,"found_c_a_a_start"], ' ': [' ', R, "found_c_a_a_start"], 'c': ['c',
L, "found_c_a_a_start_c"]},
    "found_c_a_c_start":  {"a":  ["a",R,"found_c_a_c_start"],  "b"  :
['b',R,"found_c_a_c_start"], ' ': [' ', R, "found_c_a_c_start"], 'c': ['c',
L, "found_c_a_c_start_c"]},
    "found_c_a_c_start_c": {"a": ["c",N,"end"], "b" : ['c',N,"end"],
' ': ['c', N, "end"], 'c': ['c', N, "end"]},
    "found_c_a_a_start_c": {"a": ["a",N,"end"], "b" : ['a',N,"end"],
' ': ['a', N, "end"], 'c': ['a', N, "end"]},
    "found_c_a_b_start_c": {"a": ["b",N,"end"], "b" : ['b',N,"end"],
' ': ['b', N, "end"], 'c': ['b', N, "end"]},
}
mem = list(input())
state = "start"
ind = 0
while state != "end":
```

```
        symbol = mem[ind]
        next_s = table[state][symbol]
        mem[ind] = next_s[0]
        ind += next_s[1]
        state = next_s[2]
    print(*mem, sep='')
```

ПРИЛОЖЕНИЕ Б
ТЕСТИРОВАНИЕ

Таблица 2 - Примеры тестовых случаев

№ п/п	Входные данные	Выходные данные	Комментарии
1.	“ abcabc”	“ abcabc”	Программа заменила символ идущий перед с – “b” на символ, идущий перед с – “b”
2.	“ cbbaa”	“ acbbaa”	Программа заменила “ ” на “a”
3.	“ aabc “	“ aaac “	Программа заменила “b” на “a”