МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МОЭВМ

ОТЧЕТ

по лабораторной работе №2
по дисциплине «Информатика»
Тема: «Машина Тьюринга».

Студент гр. 1304 _____ Байков Е.С.

Преподаватель _____ Берленко Т.А.

Санкт-Петербург
2021

Цель работы

Изучение структуры машины Тьюринга, практика реализации на языке Python алгоритма машины Тьюринга.

Задание

На вход программе подается строка неизвестной длины. Каждый элемент является значением в ячейке памяти ленты Машины Тьюринга.

На ленте находится троичное число, знак (плюс или минус) и троичная цифра. Напишите программу, которая выполнит арифметическую операцию. Указатель на текущее состояние Машины Тьюринга изначально находится слева от числа (но не на первом его символе). По обе стороны от числа находятся пробелы. Результат арифметической операции запишите на месте первого числа. Для примера выше лента будет выглядеть так:

Ваша программа должна вывести полученную ленту после завершения работы.

Алфавит:

- 0
- 1
- 2
- +
- -
- " " (пробел)

Соглашения:

1. Направление движения автомата может быть одно из R (направо), L (налево), N (неподвижно).

2. Число обязательно начинается с единицы или двойки.

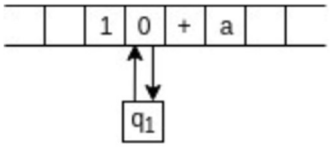
3. Числа и знак операции между ними идут непрерывно.

4. Гарантируется, что в результате операции вычитания не может получиться отрицательного числа.

В отчет включите таблицу состояний. Отдельно кратко опишите каждое состояние.

Основные теоретические положения.

Машина Тьюринга (МТ) состоит из двух частей: неподвижной бесконечной ленты (памяти) и автомата (процессора).



1. **Лента** используется для хранения информации. Она бесконечна в обе стороны и разбита на клетки, которые никак не нумеруются и не именуются. В каждой клетке может быть записан один символ или ничего не записано. Память пассивна: она ничего не делает, просто хранит данные.

2. **Алфавит ленты** - конечное множество всех возможных символов ленты. Если предположить, что видимые символы - весь алфавит ленты из примера выше, то мы имеем следующий алфавит: {1, 0, +, 'a', "}. Последний символ - пустой, означает пустое содержимое клетки.

3. **Автомат** – это активная часть Машины Тьюринга. В каждый момент он размещается под одной из клеток ленты и видит её содержимое; это **видимая клетка**, а находящийся в ней символ – **видимый символ**; содержимое же соседних и других клеток автомат не видит. Кроме того, в каждый момент автомат находится в одном из **состояний**, которые обычно обозначаются буквой q с номерами: q0, q1, q2 и т.д. Существует конечное число таких состояний.

В каждом из состояний автомат выполняет какую-то конкретную операцию. Существует заключительное состояние, в котором автомат останавливается.

Автомат за один такт (шаг) может выполнить следующие действия:

1. считать видимый символ;
2. записывать в видимую клетку новый символ (в том числе пустой символ);
3. сдвигаться на одну клетку влево или вправо («перепрыгивать» сразу через несколько клеток автомат не может);
4. перейти в следующее состояние.

Выполнение работы

Перед написание программы на языке Python была создана таблица состояний, представленная в таблице 1.

Таблица 1 – Таблица состояний машины Тьюринга.

	'0'	'1'	'2'	'+'	'-'	''
q_start	'0';R;q_start	'1';R;q_start	'2';R;q_start	'+';R;q_num	'-';R;q_num	'';R;q_start
q_num	'0';N;q_end	'1';L;q_action_1	'2';L;q_action_2			
q_action_1	'1';N;q_end	'2';N;q_end	'0';L;q_action_1	'+';L;q_action_1	'-';L;q_action_-1	'1';N;q_end
q_action_2	'2';N;q_end	'0';L;q_action_1	'1';L;q_action_1	'+';L;q_action_2	'-';L;q_action_-2	'2';N;q_end
q_action_-1	'2';L;q_action_-1	'0';L;q_test_0	'1';N;q_end			
q_action_-2	'1';L;q_action_-1	'2';L;q_action_-1	'0';L;q_test_0			
q_test_0	'0';N;q_end	'1';N;q_end	'2';N;q_end			'';R;q_del_0

q_del_0	';R;q_del_0	'1';N;q_end	'2';N;q_end	'+';L;q_del_0	'-';L;q_del_0	'0';N;q_end
q_end						

Описание каждого состояния машины Тьюринга:

q_start – начальное состояние, которое доходит до знака «+» или «-» и переходит в состояние *q_num*.

q_num – состояние, которое определяет какую цифру надо прибавить к числу. В случае если надо прибавить 0 то переходит в состояние *q_end*, завершая выполнение программы. Иначе переходит в состояния *q_action_1* или *q_action_2*.

q_action_1 – состояние, в которое переходит автомат машины при прибавлении (вычитании) единицы. Если находит «-» то переходит в состояние *q_action_-1*. Иначе прибавляет к числу единицу. Если требуется увеличить следующий разряд числа на единицу, совершается переход в следующий разряд.

q_action_2 – состояние, в которое переходит автомат машины при прибавлении (вычитании) двойки. Если находит «-» то переходит в состояние *q_action_-2*. Иначе прибавляет к числу 2. Если требуется увеличить следующий разряд на единицу переходит в состояние *q_action_1*.

q_action_-1 – состояние, в которое переходит автомат машины при вычитании единицы из числа. Если нужно вычесть единицу из старшего разряда, то переходит в старший разряд. Если появляется 0 при вычитании, то переходит в состояние *q_test_0*.

q_action_-2 – состояние, в которое переходит автомат машины при вычитании двойки из числа. Если нужно вычесть единицу из старшего разряда, то переходит в состояние *q_action_-1*. Если появляется 0 при вычитании, то переходит в состояние *q_test_0*.

q_test_0 – состояние, в которое переходит автомат машины для проверки существования незначащих нулей. При нахождении на пустой клетке автомат переходит в состояние *q_del_0*.

q_del_0 – состояние, в котором происходит удаление незначащих нулей и добавление значащего если вдруг число поле операций стало нулем.

q_end – состояние, которое завершает работу автомата машины Тьюринга.

Описание работы программы:

Данная таблица была перенесена в программу в виде словаря *states*, ключи которого являются состояниями автомата, а значениями – словари, в которых ключами являются символы на ленте, а значением каждого ключа является список вида: *[sign, step, state]*, где *sign* – значение, которое надо вписать в клетку; *step* – шаг, который надо совершить после записи значения; *state* – состояние в которое должен перейти автомат, после завершения предыдущих действий. Значение *step* записывается в виде переменных L, R, N, которые хранят в себе значения -1, 1, 0 соответственно. Переменные инициализированы до словаря *states*.

После инициализируется три переменные: *current_state*, *index*, *line*. Переменная *current_state* принимает значение начального состояния автомата, *index* принимает значение 0, а *line* принимает в себя список символов строки, введенной пользователем.

С помощью цикла *while* проводится работа машины Тьюринга, до того момента пока значение *current_state* не станет равно конечному состоянию. Внутри цикла трем переменным *new_sign, step, new_state* присваиваются значения списка внутри словаря *states* (список находится с помощью следующей конструкции – *states[current_state][line[index]]*). После символу строки (*line*) с индексом *index* присваивается значение *new_sign*. К *index* прибавляется значение *step*, а значение *current_state* изменится на значение *new_state*. С помощью метода *join* изменяется значение переменной *line* и

присваивается строка, состоящая из символов списка *line*. С помощью *print* выводится значение *line*.

Тестирование

Результаты тестирования представлены в табл. 2.

Таблица 2 – Результаты тестирования.

№ п/п	Входные данные	Выходные данные	Комментарии
1.	111+1	112+1	Корректно
2.	20+2	22+2	Корректно
3.	11111-2	11102-2	Корректно
4.	121+2	200+2	Корректно
5.	1-1	0-1	Корректно

Выводы

В ходе выполнения лабораторной работы были изучены принципы работы машины Тьюринга. Была разработана программа, имитирующая работу машины Тьюринга. Была сделана таблица состояний автомата для машины Тьюринга, выполняя инструкции, которой машина производит сложение(вычитание) чисел в троичной системе счисления.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: Baykov_Egor_lb2.py

```
L, R, N = -1, 1, 0
states = {
    'q_start': {'0': ['0', R, 'q_start'], '1': ['1', R, 'q_start'], '2':
['2', R, 'q_start'], '+': ['+', R, 'q_num'], '-': ['-', R, 'q_num'], '
': [' ', R, 'q_start']},
    'q_num': {'0': ['0', N, 'q_end'], '1': ['1', L, 'q_action_1'], '2':
['2', L, 'q_action_2']},
    'q_action_1': {'0': ['1', N, 'q_end'], '1': ['2', N, 'q_end'], '2':
['0', L, 'q_action_1'], '+': ['+', L, 'q_action_1'], '-': ['-', L,
'q_action_-1'], ' ': ['1', N, 'q_end']},
    'q_action_2': {'0': ['2', N, 'q_end'], '1': ['0', L, 'q_action_1'],
'2': ['1', L, 'q_action_1'], '+': ['+', L, 'q_action_2'], '-': ['-', L,
'q_action_-2'], ' ': ['2', N, 'q_end']},
    'q_action_-1': {'0': ['2', L, 'q_action_-1'], '1': ['0', L,
'q_test_0'], '2': ['1', N, 'q_end']},
    'q_action_-2': {'0': ['1', L, 'q_action_-1'], '1': ['2', L,
'q_action_-1'], '2': ['0', L, 'q_test_0']},
    'q_test_0': {'0': ['0', N, 'q_end'], '1': ['1', N, 'q_end'], '2':
['2', N, 'q_end'], ' ': [' ', R, 'q_del_0']},
    'q_del_0': {'0': [' ', R, 'q_del_0'], '1': ['1', N, 'q_end'], '2':
['2', N, 'q_end'], '+': ['+', L, 'q_del_0'], '-': ['-', L, 'q_del_0'],
' ': ['0', N, 'q_end']}
}

current_state = 'q_start'
index = 0
line = list(input())
while current_state != 'q_end':
    new_sign, step, new_state = states[current_state][line[index]]
    line[index] = new_sign
    current_state = new_state
    index += step

line = ''.join(line)
print(line)
```