

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

ОТЧЕТ
по лабораторной работе №2
по дисциплине «Информатика»
ТЕМА: МОДЕЛИРОВАНИЕ РАБОТЫ МАШИНЫ ТЬЮРИНГА

Студент гр.0382

Тюленев Т.В.

Преподаватель

Шевская Н.В.

Санкт-Петербург

2020

Цель работы.

Смоделировать работу машины Тьюринга на языке программирования Python.

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

		1	2	1	+	2			
--	--	---	---	---	---	---	--	--	--

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 про-белы. Результат арифметической операции запишите на месте первого числа. Для примера выше лента будет выглядеть так:

		2	0	0	+	2			
--	--	---	---	---	---	---	--	--	--

Программа должна вывести полученную ленту после завершения работы.

Алфавит:

- 0
- 1

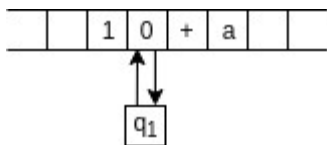
- 2
- +
- -
- " " (пробел)

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.

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: $\{1, 0, +, 'a', '\text{ }'\}$. 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: q_0 , q_1 , q_2 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.

В данной работе были использованы такие конструкции языка Python как:

- Встроенные функции:

- `input()` – считывает входные данные, возвращает строку
- `print()` – выводит аргумент на консоль

- Цикл:

- `While <...>` - выполняет тело цикла, пока условие верно

- Методы:

- `str.split(sep)` – метод класса `str`, принимает на вход разделитель (по умолчанию – пробел) и разбивает строку, к которой применён, на подстроки по разделителю и возвращает список этих подстрок
- `list.append()` – добавляет аргумент в конец списка `list`
- `“...”.join(list)` – преобразует элементы объекта в строку с разделителем “...”

Выполнение работы.

Составим таблицу состояний (таблица 1)

Таблица 1 – Таблица состояний

	‘0’	‘1’	‘2’	‘+’	‘-’	‘ ’
q1	‘0’,1,’q2’	‘1’,1,’q2’	‘2’,1,’q2’			‘ ’,1,’q1’
q2	‘0’,1,’q2’	‘1’,1,’q2’	‘2’,1,’q2’	‘+’,1,’q3’	‘-’,1,’q7’	
q3	‘0’,-2,’q4’	‘1’,-2,’q5’	‘2’,-2,’q6’			
q4	‘0’,-1,’q4’	‘1’,-1,’q4’	‘2’,-1,’q4’			‘ ’,0,’qT’
q5	‘1’,-1,’qT’	‘2’,-1,’qT’	‘0’,-1,’qT’			‘1’,-1,’qT’
q6	‘2’,-1,’qT’	‘0’,-1,’q5’	‘2’,-1,’q5’			
q7	‘0’,-2,’q4’	‘1’,-2,’q8’	‘2’,-2,’q9’			
q8	‘2’,-1,’q8’	‘0’,-1,’qT’	‘1’,-1,’qT’			
q9	‘1’,-1,’q8’	‘2’,-1,’q8’	‘0’,-1,’qT’			
qT	‘0’,-1,’qT’	‘1’,-1,’qT’	‘2’,-1,’qT’			‘ ’,1,’qC’
qC	‘0’,1,’qC1’	‘1’,1,’qCT’	‘2’,1,’qCT’			
qC1		‘1’,-1,’qC2’	‘2’,-1,’qC2’	‘+’,0,’qCT’	‘-’,0,’qCT’	
qC2	‘ ’,0,’qCT’					

Описание состояний

q1 – начальное состояние, машина ищет начало слова;

q2 – состояние для определения арифметической операции, которую необходимо выполнить;

q3 – состояние для определения цифры, которую необходимо

прибавить; q4 – состояние, в котором автомат к числу

прибавляет/вычитает 0 ;

q5 – состояние, в котором автомат к числу прибавляет

единицу; q6 – состояние, в котором автомат к числу

прибавляет двойку;

q7 – состояние для определения цифры, которую необходимо

вычесть; q8 – состояние, в котором автомат вычитает из числа

единицу;

q9 – состояние, в котором автомат вычитает из числа двойку;

qT – состояние, в котором автомат передвигается к началу слова;

qC – состояние, в котором автомат проверяет, какая цифра стоит в

начале слова;

qC1 – состояние, в котором автомат проверяет, что находится в

следующей клетке после нуля;

qC2 – состояние, в котором убирается незначащий ноль;

На вход, с помощью функции *input()*, поддается строка *memory*, которая является начальным состоянием ленты МТ, эта строка далее с помощью строкового метода *.split(sep)* преобразуется в список. Номер ячейки, которую в данный момент рассматривает машина записывается в переменную *index*. Переменной *q* присваивается начальное состояние машины – “*q1*”, далее с помощью словаря *table* реализуется таблица 1. Обработка ленты производится с помощью цикла *while* с условием, что работа продолжается пока состояние машины не достигнет конечного состояния “*qCT*”. Каждую итерацию цикла в переменную *sum* записывается символ, который необходимо записать на ленту, в переменную *delta* записывается число, на которое изменится индекс, а в переменную *state* записывается состояние, в которое перейдет машина Тьюринга. Далее с

помощью команды $memory[index]=sym$ на ленту записывается новое значение, с помощью команда $index+=delta$ происходит движение по ленте, а $q=state$ – изменяет состояние машины Тьюринга. С помощью команды $print("".join(memory))$ выводится измененная лента машины Тьюринга.

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 2 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1.	101-2	22-2	Программа работает правильно
2.	1-1	0-1	Программа работает правильно
3.	101+2	110+2	Программа работает правильно
4	21+0	21+0	Программа работает правильно
5	11-2	2-2	Программа работает правильно
6	11+1	12+1	Программа работает правильно

Выводы.

Была смоделирована работа машины Тьюринга на языке программирования Python.

Разработана программа, выполняющая считывание с клавиатуры исходных данных о состоянии ленты машины Тьюринга с помощью функции `input()`, в самой программе с помощью двумерного словаря реализована таблица состояний. Программа выполняет сложение или вычитание из троичного числа цифры, процесс изменения исходной ленты реализован с помощью цикла `while`.

С помощью функции `print()` на экран выводится изменённая лента памяти.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: lab2.py

```
memory=list(input())
index=0
q='q1'
table={
    'q1': {'0': ['0', 1, 'q2'], '1': ['1', 1, 'q2'], '2': ['2', 1, 'q2'], " ": ["
", 1, 'q1']},
    'q2':  {'0': ['0', 1, 'q2'], '1': ['1', 1, 'q2'], '2': ['2', 1, 'q2'],
'+': ['+', 1, 'q3'], '-': ['-', 1, 'q7']},
    'q3':  {'0': ['0', -2, 'q4'], '1': ['1', -2, 'q5'], '2': ['2', -2, 'q6']},
    'q4':  {'0': ['0', -1, 'q4'], '1': ['1', -1, 'q4'], '2': ['2', -1, 'q4'], "
": [" ", 0, 'qT']},
    'q5':  {'0': ['1', -1, 'qT'], '1': ['2', -1, 'qT'], '2': ['0', -1, 'q5'], "
": ['1', -1, 'qT']},
    'q6':  {'0': ['2', -1, 'qT'], '1': ['0', -1, 'q5'], '2': ['1', -1, 'q5']},
    'q7':  {'0': ['0', -2, 'q4'], '1': ['1', -2, 'q8'], '2': ['2', -2, 'q9']},
    'q8':  {'0': ['2', -1, 'q8'], '1': ['0', -1, 'qT'], '2': ['1', -1, 'qT']},
    'q9':  {'0': ['1', -1, 'q8'], '1': ['2', -1, 'q8'], '2': ['0', -1, 'qT']},
    'qT':  {'0': ['0', -1, 'qT'], '1': ['1', -1, 'qT'], '2': ['2', -1, 'qT'], "
": [" ", 1, 'qC']},
    'qC':  {'0': ['0', 1, 'qC1'], '1': ['1', 1, 'qCT'], '2': ['2', 1, 'qCT']},
    'qC1': {'1': ['1', -1, 'qC2'], '2': ['2', -
1, 'qC2'], '+': ['+', 0, 'qCT'], '-': ['-', 0, 'qCT']},
    'qC2': {'0': [' ', 0, 'qCT']}
}
while q!='qCT':
    sym,delta,state = table[q][memory[index]]
    memory[index]=sym
    index +=delta
    q=state

print(''.join(memory))
```