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. 8303 _____ Гришин К. И.

Преподаватель _____ Берленко Т. А.

Санкт-Петербург

2018

Цель работы

Научиться выстраивать алгоритмы на языке Python3, моделирующие работу 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.

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

		2	0	0	+	2			

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.

Ход работы

Машина Тьюринга 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.

2. **Алфавит ленты** - конечное множество всех возможных символов ленты.

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.**

На вход программе подается строка вида трочиное число+троичная цифра по краям которой находятся пробелы. Инициализируем переменную *«state»,* которая будет использоваться для определения состояния, в котором находится Машина Тьюринга. Также инициализируем переменную *«i»*, которая будет использоваться как счетчик для того, чтобы шагать по ленте.

Цикл *«while»* запускает машину, которая будет работать пока не дойдет до заключительного состояния *«p11».* В ходе достижения заключительного состояния машина проходит по «рабочим» состояниям:

p0 — Ход по ленте вправо до первой цифры.

p1 — Ход по ленте вправо до знака сложения или вычитания.

p+ — Состояние для сложения. Делает шаг вправо и смотрит на цифру которую нужно прибавить.

p3 — Ход по ленте влево и прибавление единицы, если после знака *«+»* стоял знак *«1»*.

p4 — Ход по ленте влево и прибавление двойки, если после знака *«+»* стоял знак *«2»*.

p- — Состояние для вычитания. Делает шаг вправо и смотрит на цифру, которую нужно отнять.

p6 — Ход по ленте влево и вычитание единицы, если после знака *«-»* стоял знак *«1»*. Если число получилось меньшего разряда, чем исходное, то последний разряд становится равен нулю.

p7 — Ход по ленте влево и вычитание двойки, если после знака *«-»* стоял знак *«2»*. Если число получилось меньшего разряда, чем исходное, то последний разряд становится равен нулю.

Машина Тьюринга дошла до состояния p8 исходное число претерпело нужные преобразования. Но может существовать такая ситуация, когда в итоговом числе располагаются незначащие нули.

Добавим состояний для того, чтобы убрать незначащие нули.

p8 — Ход влево до пробела, затем один шаг вправо.

p9 — Нули заменяются пробелами и при этом делается шаг вправо.

p10 — Если был встречен знак сложения или вычитания делается шаг вправо и пробел заменяется знаком *«0»*.

Программа завершается с состоянием p11

Таблица состояний:

	0	1	2	+	-	«»
p0	0, p1, R	1, p1, R	2, p1, R	+, p1, R	-, p1, R	«», p0, R
p1	0, p1, R	1, p1, R	2, p1, R	+, p+, R	-, p-, R	
p+	0, p8, H	1, p3, L	2, p4, L			«», p8, H
p3	1, p8, H	2, p8, H	0, p3, L	+, p3, L		1, p8, H
p4	2, p8, H	0, p3, L	1, p3, L	+, p4, L		2, p8, H
p-	0, p8, H	1, p6, L	2, p7, L			«», p8, H
p6	2, p6, L	0, p8, H	1, p8, H		-, p6, L	
p7	1, p6, L	2, p6, L	0, p8, H		-, p7, L	
p8	0, p8, L	1, p8, L	2, p8, L			«», p9, R
p9	«», p9, R	1, p11, H	2, p11, H	+, p10, L	-, p10, L	
p10	0, p11, H	1, p11, H	2, p11, H			0, p11, H
p11						

Выводы.

В ходе лабороторной работы были изучены алгоритм прибавления цифры к числу в Машине Тьюринга, а также механизм работы Машины Тьюринга.

Разработана программа, которая путем работы со списком с помощью условных операторов *«if»* выполняет операцию сложения числа с цифрой по правилам Машины Тьюринга.

ПРИЛОЖЕНИЕ

КОД ПРОГРАММЫ

```
1     memory = list(input())
2
3     state = 'p0'
4     i = 0
5     while state != 'p11':
6         if state == 'p0':
7             if memory[i] == '0' or memory[i] == '1' or memory[i] == '2' or memory[i]
== '+' or memory[i] == '-':
8                 i += 1
9                 state = 'p1'
10            elif memory[i] == ' ':
11                i += 1
12
13        elif state == 'p1':
14            if memory[i] == '0' or memory[i] == '1' or memory[i] == '2':
15                i += 1
16            elif memory[i] == '+':
17                state = 'p+'
18                i += 1
19            elif memory[i] == '-':
20                state = 'p-'
21                i +=1
22
23        elif state == 'p+':
24            if memory[i] == '0' or memory[i] == ' ':
25                state = 'p8'
26            elif memory[i] == '1':
27                state = 'p3'
28                i -= 1
29            elif memory[i] == '2':
30                state = 'p4'
31                i -= 1
32
33        elif state == 'p3':
34            if memory[i] == '+':
35                i -= 1
36            elif memory[i] == '0' or memory[i] == '1':
37                state = 'p8'
38                memory[i] = str(int(memory[i]) + 1)
39            elif memory[i] == '2':
40                memory[i] = '0'
41                i -= 1
42            elif memory[i] == ' ':
43                state = 'p8'
44                memory[i] = '1'
45
46        elif state == 'p4':
47            if memory[i] == '+':
48                i -= 1
49            elif memory[i] == '1' or memory[i] == '2':
50                state = 'p3'
51                memory[i] = str(int(memory[i]) - 1)
52                i -= 1
53            elif memory[i] == '0':
54                state = 'p8'
55                memory[i] = '2'
56            elif memory[i] == ' ':
57                state = 'p8'
58                memory[i] = '2'
59
60        elif state == 'p-':
61            if memory[i] == '0' or memory[i] == ' ':
62                state = 'p8'
63            elif memory[i] == '1':
64                state = 'p6'
65                i -= 1
```

```
66            elif memory[i] == '2':
67                state = 'p7'
68                i -= 1
69
70        elif state == 'p6':
71            if memory[i] == '1' or memory[i] == '2':
72                state = 'p8'
73                memory[i] = str(int(memory[i]) - 1)
74            elif memory[i] == '0':
75                memory[i] = '2'
76                i -= 1
77            elif memory[i] == '-':
78                i -= 1
79
80        elif state == 'p7':
81            if memory[i] == '0' or memory[i] == '1':
82                state = 'p6'
83                memory[i] = str(int(memory[i]) + 1)
84                i -= 1
85            elif memory[i] == '2':
86                memory[i] = '0'
87                state ='p8'
88            elif memory[i] == '-':
89                i -= 1
90
91        elif state == 'p8':
92            if memory[i] == '0' or memory[i] == '1' or memory[i] == '2':
93                i -= 1
94            elif memory[i] == ' ':
95                state = 'p9'
96                i += 1
97
98        elif state == 'p9':
99            if memory[i] == '1' or memory[i] == '2':
100               state = 'p11'
101           elif memory[i] == '0':
102               memory[i] = ' '
103               i += 1
104           elif memory[i] == '+' or memory[i] == '-':
105               state = 'p10'
106               i -= 1
107
108       elif state == 'p10':
109           if memory[i] == '0' or memory[i] == '1' or memory[i] == '2':
110               state = 'p11'
111           if memory[i] == ' ':
112               state = 'p11'
113               memory[i] = '0'
114
115   print(''.join(memory))
```