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 Санников В.А.

Преподаватель Шевская Н.В.

Санкт-Петербург

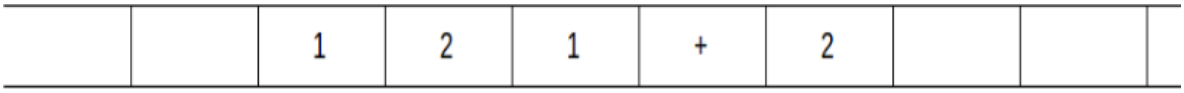
2020

Цель работы.

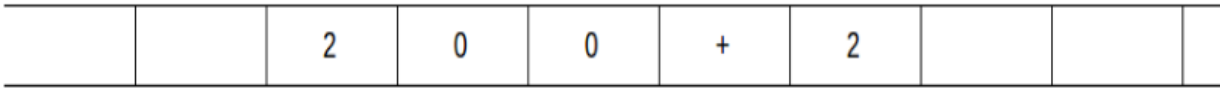
Смоделировать работу машины Тьюринга на языке Python.

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.



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.

Основные теоретические положения.

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

Алфавит ленты - конечное множество всех возможных символов ленты{0, 1, 2, +, -, ' '}. 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 3 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. В данной лабораторной работе были использованы следующие конструкции языка Python:

Встроенные функции Python:

◦input() - возвращает считываемое с консоли значение

◦print() - выводит на консоль принимаемое в качестве аргумента значение

циклы: ◦while: <логическое выражение>:— если значение выражения после оператора while и перед «:» истинно, то выполняется последовательность действий, выделенная табуляцией.

Выполнение работы.

Ход работы:

Чтобы понять, как работает наша программа, составим таблицу состояний (таблица 1)

Таблица 1 – Таблица Состояний

	0	1	2	+	-	« »
q1	0,R,q2	1,R,q2	2,R,q2	+,R,q2	-,R,q2	« »,R,q1
q2	0,R,q2	1,R,q2	2,R,q2	+,R,q3	-,R,q3	
q3	0,N,q13	1,L,q4	2,L,q7			
q4				+,L,q5	-,L,q6	
q5	1,N,q10	2,N,q10	0,L,q5			1,N,q10
q6	2,L,q6	0,N,q10	1,N,q10			
q7				+,L,q8	-,L,q9	
q8	2,N,q10	0,L,q5	1,L,q5			
q9	1,L,q6	2,L,q6	0,N,q10			
q10	« »,L,q10	1,L,q10	2,L,q10	+,L,q10	-,L,q10	« »,R,q11
q11	« »,R,q11	1,N,q13	2,N,q13	+,L,q12	-,L,q12	
q12						0,R,q13

Описание состояний:

◦q1 — поиск начала слова

◦q2 — нахождение цифры, которую необходимо прибавить или отнять

◦q3 — автомат определяет какую цифру необходимо прибавить или отнять(если это 0, то сразу выход из цикла)

◦q4 — определение действия для 1

◦q5 — прибавляем 1

◦q6 — вычитаем 1

◦q7 — определение действия для 2

◦q8 — прибавляем 2

◦q9 — вычитаем 2

◦q10 — определение начала числа

◦q11 — удаление незначащих 0

◦q12 — если в числе не осталось цифр, возвращает 1

◦q13 — конечное положение автомата, выход из цикла

Краткий алгоритм: Считываем данные с клавиатуры в переменную memory и с помощью функции list() преобразуем входную строку в список символов, определяемся начальное состояние, i — перемещение автомата по «ленте», число троичное → «Лента» обрабатывается с помощью цикла while (пока не конечное состояние), таблица состояний записана в виде двумерного словаря table → измененная «лента» выводится на консоль с помощью функции print() и метода .join()

Тестирование.

Результаты тестирования представлены в табл. 2.

Таблица 2 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1.	122+0	122+0	Программа работает верно
2.	100-1	22-1	Программа работает верно
3.	2-2	0-2	Программа работает верно

4.	01212-2	1210-2	Программа работает верно

Выводы.

В ходе работы над программой был изучен принцип работы и моделирования МТ. Разработана программа (см.Приложение А), которая выполняется следующий алгоритм: Считываем данные с клавиатуры в переменную memory и с помощью функции list() преобразуем входную строку в список символов, определяемся начальное состояние, i — перемещение автомата по «ленте», число троичное → «Лента» обрабатывается с помощью цикла while (пока не конечное состояние), таблица состояний записана в виде двумерного словаря table → измененная «лента» выводится на консоль с помощью функции print() и метода .join()

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.py

```
memory = list(input())
i = 0
q = 'q1'

table = {'q1': {'0': ['0', 1, 'q2'],
                '1': ['1', 1, 'q2'],
                '2': ['2', 1, 'q2'],
                '+': ['+', 1, 'q2'],
                '-': ['-', 1, 'q2'],
                ' ': [' ', 1, 'q1'],},

         'q2': {'0': ['0', 1, 'q2'],
                '1': ['1', 1, 'q2'],
                '2': ['2', 1, 'q2'],
                '+': ['+', 1, 'q3'],
                '-': ['-', 1, 'q3'],},

         'q3': {'0': ['0', 0, 'q13'],
                '1': ['1', -1, 'q4'],
                '2': ['2', -1, 'q7'],},

         'q4': {'+': ['+', -1, 'q5'],
                '-': ['-', -1, 'q6'],},

         'q5': {'0': ['1', 0, 'q10'],
                '1': ['2', 0, 'q10'],
                '2': ['0', -1, 'q5'],
```

```
        ' ': ['1', 0, 'q13'],},

 'q6': {'0': ['2', -1, 'q6'],
        '1': ['0', 0, 'q10'],
        '2': ['1', 0, 'q10'],},

 'q7': {'+': ['+', -1, 'q8'],
        '-': ['-', -1, 'q9'],},

 'q8': {'0': ['2', 0, 'q10'],
        '1': ['0', -1, 'q5'],
        '2': ['1', -1, 'q5'],},

 'q9': {'0': ['1', -1, 'q6'],
        '1': ['2', -1, 'q6'],
        '2': ['0', 0, 'q10'],},

'q10': {'0': ['0', -1, 'q10'],
        '1': ['1', -1, 'q10'],
        '2': ['2', -1, 'q10'],
        '+': ['+', -1, 'q10'],
        '-': ['-', -1, 'q10'],
        ' ': [' ', 1, 'q11'],},
'q11': {'0': [' ', 1, 'q11'],
        '1': ['1', 0, 'q13'],
        '2': ['2', 0, 'q13'],
        '+': ['+', -1, 'q12'],
        '-': ['-', -1, 'q12']},
'q12': {' ': ['0', 1, 'q13']}
}
```

```
while q != 'q13':
    sym, delta, state = table[q][memory[i]]
    memory[i] = sym
    i += delta
    q = state
print(''.join(memory))
```