

Зачетное задание

Тема: МНР, машины Тьюринга

Студента группы МФ-21
Чистякова Артема

Вариант №1

1) Условие

Постройте МНР-программу, которая вычисляет функцию
 $f(x) = (2x)!! = 2 \cdot 4 \cdot \dots \cdot 2x$ ($f(0) = 1$) Результат должен быть записан в первом регистре.

1.2) Описание алгоритма

Алгоритм работает следующим образом. В программе 3 цикла: внешний, который бежит по количеству слагаемых в факториале, и два вложенных для умножения ответа. На каждой внешней итерации мы увеличиваем внешний счетчик и сохраняем значение (текущий четный факториал), на который нужно умножить ответ. Далее запускаются циклы умножения ответа на переменную четного факториала. Умножение происходит путем разложения на суммы.

1.3) Текст программы

```
1.      S (3) // тут храним ответ
2.      J (1, 2, 18) // внешний цикл по слагаемым
3.          S (2) // внешний итератор
4.          S (4) // число на котором нужно умножить ответ
5.          S (4)
6.          Z (5) // тут храним внешний итератор для умножения
7.          S (5) // один раз ответ уже взяли
8.          T (3, 7)
9.          J (4, 5, 17) // внешний цикл умножения
10.             S (5)
11.             Z (6) // тут храним вложенный итератор для умножения
12.             J (7, 6, 16) // вложенный цикл умножения
13.                 S (3)
14.                 S (6)
15.                 J (1, 1, 12)
16.                 J (1, 1, 9)
17.             J (1, 1, 2)
18.         T (3, 1) // переместить ответ в 1 регистр
```

1.4) Тесты

[illegible][illegible]

The screenshot displays the 'v1.4' software interface, which is used for configuring and monitoring a system. The interface is divided into several sections:

- Top Bar:** Contains a title bar with standard window controls and a version indicator 'v1.4'.
- Instruction List:** A table on the left side listing instructions with their addresses and names:

№	Имя
1	S (3)
2	J (1,2,18)
3	S (2)
4	S (4)
5	S (3)
6	Z (5)
7	S (5)
8	T (3,7)
9	J (4,5,17)
10	S (5)
11	Z (6)
12	J (7,6,16)
13	S (3)
14	S (6)
15	J (1,1,12)
16	J (1,1,9)
17	J (1,1,2)
18	T (3,1)
- Control Panel (ZSTJ):** A vertical panel in the center with buttons for Z, S, T, J, and a set of directional controls (left, right, up, down, and a center button).
- Data Tables:** Two tables on the right side, each with columns '№' and 'Значение' (Value).

№	Значение
1	8
2	2
3	8
4	4
5	4
6	2
7	2
- Status Bar:** At the bottom, it shows a timer '00:00:00', a plus/minus button, and a row of icons representing different system components.
- Bottom Section:** Contains a row of icons and a text label 'размер кода \$f\$' (code size \$f\$), followed by a section for 'свободные регистры' (free registers) which is currently empty.

[illegible]

К сожалению, у программы машины Тьюринга на моем компьютере напрочь слетела кодировка, и для того чтобы она хоть как-то работала, пришлось подобрать символы, соответствующие слетевшей кодировке.

Соответственно:

- Ì = П (Право)
- Ê = Л (Лево) (' . ∪ .)
- Í = Н (На месте)

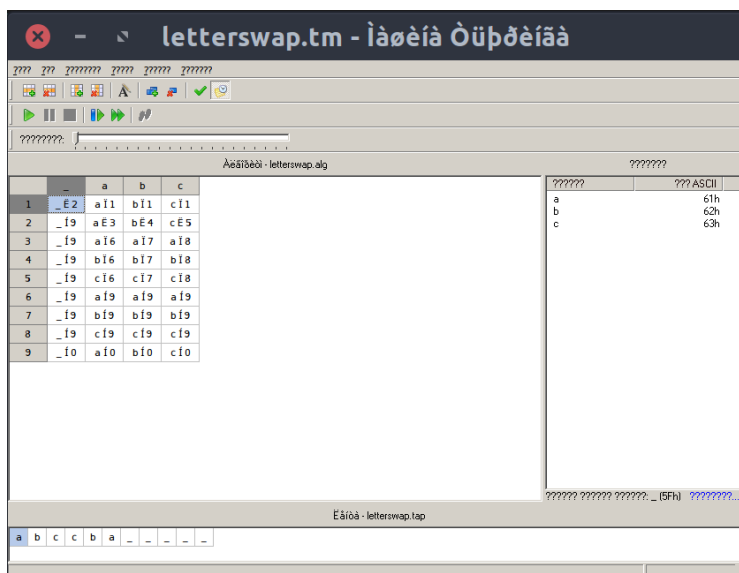
2) Условие

На ленте задано произвольное слово (возможно, пустое) в алфавите $A = \{a, b, c\}$ (остальные символы ленты пустые). Напишите программу для машины Тьюринга, которая в данном слове меняет местами последние две буквы (например, из *abca* делает *abac*). Считайте, что первоначально головка смотрит на первую букву слова.

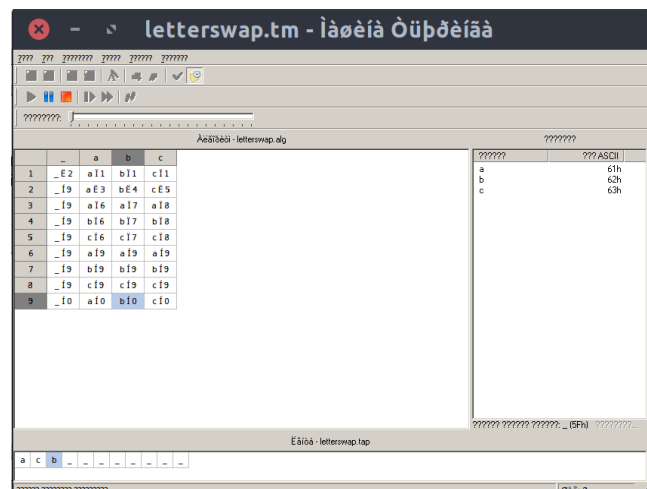
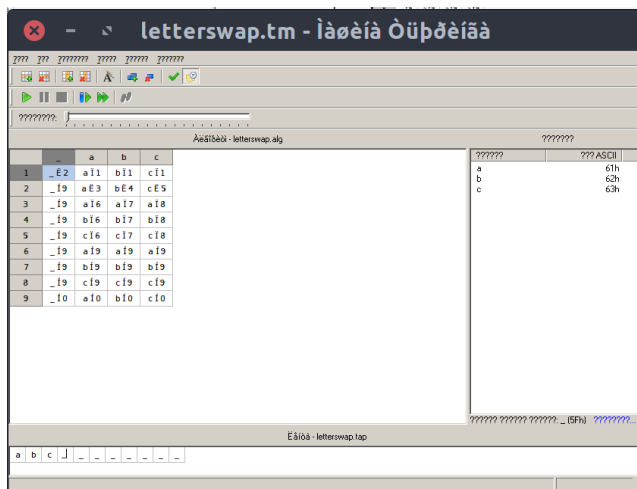
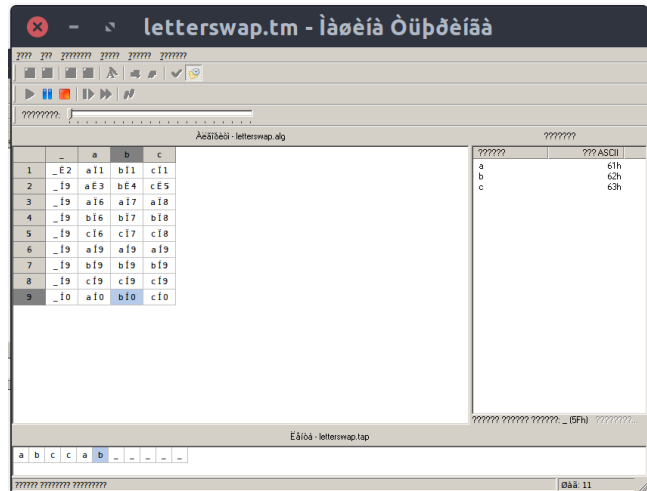
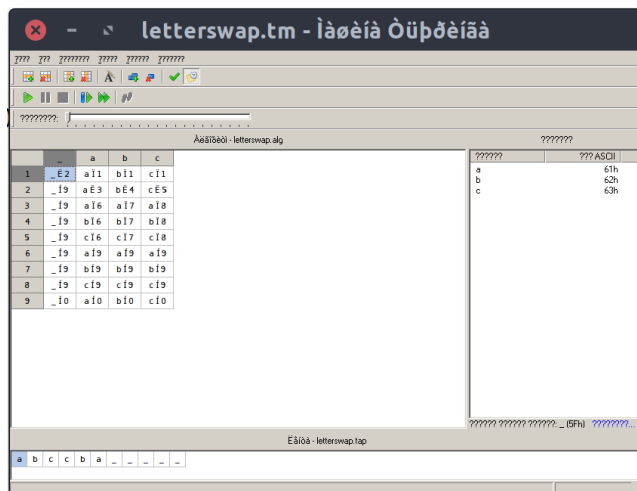
2.2) Описание алгоритма

Сначала пройдем до конца вводимого слова (об этом нам сообщит пустой символ). Далее подвинемся на 1 влево и рассмотрим 3 ситуации: считали *a* *b* или *c* . Из каждой буквы запустим свое состояние, которое отвечает за запись этой буквы в ячейку ленты левее. Повторим те же действия для новой ячейки, только с продвижением на 1 вправо.

2.3) Код программы



2.4) Тесты



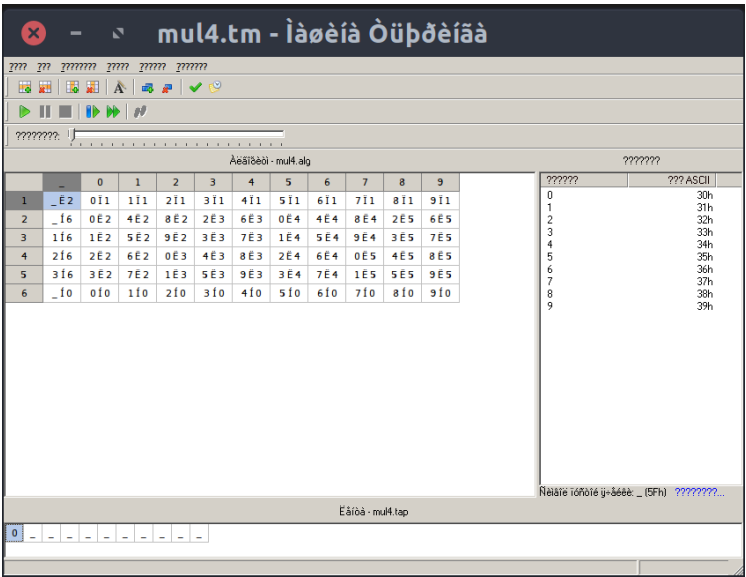
3) Условие

Напишите программу для машины Тьюринга, которая правильно вычисляет функцию $f(x) = 4x$.

3.2) Описание алгоритма

Пройдем до конца данного числа (об этом нам сообщит пустой символ). Далее будет двигаться влево по ленте, умножая числа на 4 и прибавляя переносимый разряд. Чтобы запомнить разряд, создадим 4 дополнительные состояния: разряд равен 0, разряд равен 1, разряд равен 2 и разряд равен 3.

3.3) Код программы



3.4) Тесты

