

Домашняя работа №3

ЗАДАНИЕ 1. Операции с числами

Реализуйте машины Тьюринга, которые позволяют выполнять следующие операции:

1. Сложение двух унарных чисел

Машина Тьюринга принимает две последовательности единиц, разделённых плюсом (например, 111+111), после выполнения алгоритма машина установит каретку на начало результирующего числа. Машина будет удалять первую единицу и заменять знак + между числами на единицу, тем самым склеивая аргументы. составить таблицу переходов:

состояние	1	+	ϵ
first	toPlus, ϵ , R	done, ϵ , R	
toPlus	R	toStart, 1, L	
toStart	L		done, R
done			H

Код для реализации находится в файле 1_1.yaml

2. Умножение унарных чисел

Для того, чтобы составить машину Тьюринга определим умножение следующим образом:

$$\text{mult}(0, b) = 0$$

$$\text{mult}(a, b) = b + \text{mult}(a - 1, b)$$

Машина Тьюринга принимает две последовательности единиц, разделённых знаком умножения (например, 1111*11), после выполнения алгоритма машина установит каретку на начало результирующего числа. Суть алгоритма заключается в последовательном уменьшении a и копировании b на каждом шаге. Составим таблицу переходов:

состояние	1	*	ϵ
eachA	toB, ϵ , R	skip, *, R	
toB	R	eachB, *, R	
nextA	L	L	eachA, 1, R
skip	R		H
eachB	sep, ϵ , R		nextA, ϵ , L
sep	add, 1, R		R
add	R		1, sepL, ϵ , L
sepL	L		nextB, ϵ , L
nextB	L		eachB, 1, R

Код для реализации находится в файле 1_2.yaml

ЗАДАНИЕ 2. Операции с языками и символами

Реализуйте машины Тьюринга, которые позволяют выполнять следующие операции:

1. Принадлежность к языку $L = \{0^n 1^n 2^n\}, n \geq 0$.

Машина Тьюринга будет принимать слово и в конце своей работы записывать T, если слово принадлежит языку, и F - если не принадлежит.

Будем помечать тройки из символов 0, 1, 2 буквами a, b, c, продвигаясь по слову вперёд. Как только пометим все буквы и достигнем пустого символа, можем считать, что исходное слово принадлежит языку L , если по какой-то причине этого не удалось сделать (например, раньше чем нужно достигли конца или встретили неожиданный символ), то слово не принадлежит языку.

Составим таблицу переходов:

состояние	0	1	2	a	b	c	ε
q_0	q_1aR	$q_{end}FR$	$q_{end}FR$	$q_{end}FR$	$q_{scan}bR$	$q_{end}TR$	
q_1	R	q_2bR			R		
q_2		R	$q_{back}cR$			R	
q_{back}	L	L		q_0aR	L	L	
q_{scan}	$q_{end}FR$	$q_{end}FR$	$q_{end}FR$	$q_{end}FR$	R	R	$q_{end}TR$
q_{end}							L

Код для реализации находится в файле 2_1.yaml

2. Проверка соблюдения правильности скобок в строке

Пусть машина Тьюринга принимает последовательность скобок и в конце своей работы устанавливает Т, если последовательность правильная, F - если неправильная. Будем пользоваться следующим алгоритмом:

- Движемся вправо до появления некоторой закрывающей скобки, пусть \rangle , заменяем её буквой А (другие скобки заменяем другими буквами).
- Теперь возвращаемся назад, пока не найдём соответствующую открывающую скобку, пропуская все помеченные скобки, если найдём открывающую скобку другого типа или пустой символ (т.е. вернёмся в начало), то слово неправильное.
- Нужную открывающую скобку тоже заменяем на А и повторяем этот процесс.

Если, выполняя данный процесс, мы достигли пустого символа (в данном случае конца слова), то слово правильное.

Составим таблицу переходов:

состояние	({	})	}	}	A	B	C	ε
q_{right}	R	R	R	q_aAR	q_bBR	q_cCR	R	R	R	$q_{end}TR$
q_A	$q_{right}AR$	$q_{end}FR$	$q_{end}FR$				L	L	L	$q_{end}FR$
q_B	$q_{end}FR$	$q_{right}BR$	$q_{end}FR$				L	L	L	$q_{end}FR$
q_C	$q_{end}FR$	$q_{end}FR$	$q_{right}CR$				L	L	L	$q_{end}FR$
q_{end}							L	L	L	L

Код для реализации находится в файле 2_2.yaml

3. Поиск минимальной строки

Машина Тьюринга принимает последовательность строк из 0 и 1, разделённых тире (т.к. в turingmachine.io пробелом отмечается пустой символ) и в конце своей работы устанавливает каретку на начало наименьшей строки.

Для нахождения минимальной строки будем помечать по одному символу каждой строки до тех пор пока не дойдём до разделителя-тире. После этого считаем, что минимальная строка найдена (она позади). Зачищаем ленту от ненужных строк и возвращаем каретку.

Составим таблицу переходов:

состояние	0	1	a	b	-	ε
mark	next, a, R	next, b, R	R	R	clear, R	reset, L
next	R	R			mark, R	back, L
back	L	L	L	L	L	mark, R
clear	ε, R	ε, R	ε, R	ε, R	ε, R	return, L
return					ε, R	reset, L
reset	0, L	1, L			done, R	done, R
done	H	H				H

Код для реализации находится в файле 2_3.yaml