Направление подготовки: 13.03.02 Электроэнергетика и электротехника

Наименование образовательной программы: Электрический транспорт

Уровень образования: бакалавриат

Форма обучения: очная

Дисциплина:

"Моделирование систем электрической тяги"

Комаров В.Г. Лекция 2 10.09.2025 г.

Множественные модели свойств объектов. Информация и её кодирование.

Введение

Весь мир, включая все объекты и субъекты, представляет из себя диалектическое единство бесконечного множества объектов. В свою очередь любой объект моделирования (оригинал) обычно представлен, как система S — исследуемое множество взаимосвязанных элементов любой природы. Внешняя среда E при этом представляется множеством существующих вне системы элементов любой природы, оказывающих влияние на систему или находящихся под ее воздействием.

В основе любого вида моделирования лежит постановка в соответствие оригиналу другого объекта - некоторой модели, имеющей это соответствие, базирующееся на некоторых общих свойствах, которые характеризует реальный объект. Объективно реальный объект обладает некоторой формальной структурой, поэтому для любой модели характерно наличие некоторой структуры, соответствующей формальной структуре реального объекта, либо изучаемой стороне этого объекта. Эта формальная структура выступает как информация о реальном объекте. В процессе реализации модели создаётся информация о данном объекте, а в процессе эксперимента с моделью информация, как вводится, так и выводится из модели, т. е. информация лежит в основе всего процесса моделирования.

Моделирование тесно связано с абстрактным мышлением человека, т. е. процессом отображения окружающего мира в памяти человека и предметной области его деятельности. Отображение окружающего мира формируется в виде структуры модельных представлений. Абстрактная модель это отображение реальных процессов и объектов на другие объекты, мысленные (в сознании), абстрактные (система знаков) или физические (система сигналов). Система знаков и система сигналов базируется на кодировании. Научным формальным методом процесса адекватного отображения окружающего мира являются формальные языки, как формальный метод анализа и синтеза структуры окружающего мира. Следовательно базой формализации структуры окружающего мира является кодирование информации.

Понятие информации. Кодирование информации как основа отображения структуры окружающего мира.

С современных философских позиций можно считать, что физический объект представляет собой структуру вещества. Его отображение есть структура данных. Классифицированная структура

данных отображающая свойства реальных объектов есть информация. Исследованием соотношения классов и преобразования структур занимается наука, а методов их обобщённого описания - математика, базирующаяся на теории множеств и формальных языках. Действие, как последовательность тоже может отображаться структурой данных, отображающей методы их преобразования. В этом случае такая структура данных будет исполняемой (executable) в отличие от статической структуры - библиотеки (reusable). Последовательность исполнения действий называется алгоритмом. Компьютер (вычислитель) в этом случае может рассматриваться, как физическая среда реализации (интерпретации) структуры данных в виде алгоритма.

Компьютер в этом информационном процессе выступает, как главное аппаратное средство (HARD) обработки информации по заданному алгоритму (программе) (SOFT). Таким образом можно выделить две основные составляющие этого процесса: HARDWARE - аппаратное обеспечение и SOFTWARE - программное обеспечение.

Фактически компьютер реализует методы абстрактного мышления человека. Поэтому для более ясного понимания воспользуемся методом ретроспекции и кратко рассмотрим этапы развития абстрактного мышления в форме образных моделей окружающего мира. Метод ретроспекции в моделировании позволяет на основе данных из предшествующего опыта рассматривать эволюцию сознания человека и связанные с ней формы модельных представлений.

Распознавание и классификация предметов и явлений окружающего мира явилось зарождением математической теории множеств. Отношения элементов множеств породило логику мышления, ставшей формальной основой математики.

На первом этапе появилось понятие истинного и ложного представления действительности исходя из цели выживания. Абстрактные образы способствовавшие выживанию ассоциировались с истиной, а вымиранию с ложью. Образы соотносились с ощущениями, т. е. с сенсорикой (световые, звуковые, силовые, болевые и т. п. ощущения). В памяти формировались классифицированные множества образов и их логические связи.

Решающим элементом в этом процессе стал <знак>, возникший, как отражение существенных конкретных признаков окружающего мира в виде абстрактного обозначения, которое стало в последствии основой кодирования информации. В общем понимании знаком является изображение, жест или действие, как сигнал, под которым подразумевается какое-либо конкретное смысловое значение, например, опасность или отсутствие опасности и предписанное этому значению действие. Простейшим знаком в этом информационном процессе явилась единица (UNIT) "I", интерпретируемая, т. е. физически реализуемая, в виде зарубки, палочки, пальца, камешка и т. п. По-видимому множество определённых выделенных предметов отображалось совокупностью единиц на пальцах. Поэтому близкой человеку явилась скорее всего десятичная система единиц, кратная количеству пальцев на двух руках. Таким образом уже на этом этапе возникло кодирование информации с помощью единиц. В последствии возникло и количественное описание элементов множеств в виде единичного способа счёта путём добавления (суммирования) единиц к текущему коду. Например: IIIII+I=IIIIII, где IIIII - код числа пять, а IIIIII - код числа 6. Поскольку скорее всего для счёта человек использовал пальцы рук, то для упрощения записи и наглядности придумал новые знаки, например вместо IIIII (количество пальцев одной руки) стал использовать V, а IIIII+IIIII=V+V=X и т. д. А новые знаки стали иметь свои коды, например, V имеет в единичной системе код IIIII. Действие (операция сложения) тоже получило своё обозначение в виде знака «+» и обусловило преобразование данных, переводя совокупность двух знаков V в новое значение, обозначаемое знаком X. Возникли римские цифры, как знаки, отображающие количественные отношения

предметов окружающего мира. Далее для упрощения отображения количественных отношений была придумана позиционная система счисления, где каждый знак отражал количество в зависимости от своей позиции в числе. Первой, по-видимому, возникла десятичная система позиционного счисления исходя из количества пальцев на руках. Когда пальцев не хватало то цифру переносили в следующий разряд. Здесь уже возникла определённая последовательность действий при исчислении, т. е. Возник вычислительный алгоритм. Например, число 1375=1×1000+3×100+7×10+5×1.

Таким образом код можно рассматривать, как взаимно однозначное отображение конечного упорядоченного множества символов, принадлежащих некоторому конечному алфавиту, на иное, не обязательно упорядоченное, как правило более обширное множество символов для кодирования передачи, хранения или преобразования информации. Процесс преобразования сообщения в комбинацию символов в соответствии с кодом называется *кодированием*, процесс восстановления сообщения из комбинации символов называется *декодированием*.

Такое отображение чисел уже требовало определённой последовательности операций для их исчисления, т. е. простейшую программу вычисления. Кроме того потребовалось введение нуля "0", как пустого числа (пустого множества). Важным аспектом является то, что позиционное счисление можно распространять на любое количество любых цифр или других знаков, например букв. В простейшем случае это 0 и 1, т. е. 1011=1×IIIIIIII+0×IIII+1×II+1×I=IIIIIIIIIIIII. Такое счисление базируется на двух цифрах 1 и 0 и поэтому называется двоичной системой счисления. Двоичная система счисления оказалась самой простой и удачной для её физической реализации на электронных вычислительных машинах, по-другому называемых компьютерами. Хотя механические вычислительные машины, такие как счёты и арифмометры легко справляются и с десятичной системой. Но электронным машинам лучше всего подходит двоичная система, поэтому остальные важны только для человека, чтобы легче понимать результаты вычислений. С двоичной лучше всего стыкуются четверичная, восьмеричная, шестнадцатиричная и т.д. кратные целым степеням числа 2. Наибольшее распространение получила шестнадцатиричная система цифр 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F. Возникает вопрос: а что здесь делают буквы A,B,C,D,E,F? Они здесь стали недостающими цифрами до 15. Любой знак может иметь любое смысловое значение, которое мы ему присвоим в процессе кодирования! Значит без знания кода, а точнее алгоритма кодирования, нельзя понять смысл знака. А смысл знака или кода это уже семантика языка. Так значит и язык это код! А мы, обычно рассматривая компьютерные технологии, говорим только о цифровизации! Мы отстали от полёта разума! В компьютерных технологиях необходимо разбираться не только с цифрами, а и с языком - главным движителем сознания человека. Компьютер, как и человек не только цифири гонять может! Так что давайте разбираться с буквами, словами, выражениями и языком.

Итак цифры, числа и системы счисления имеют отношение только к количественным отношениям, т. е. к простейшей арифметике. И мы сильно принижаем способности компьютера, называя его цифровым устройством. Цифровизация это вообще из эпохи зарождения человечества. Смысловые, качественные отношения это логика, структура и топология - современные разделы математики, без которых немыслима ни современная наука, ни технологии информации и систем управления. Буквы, алфавит, слова, предложения, язык описывают те отношения окружающего мира, которые не могут описать цифры. На современном уровне научного развития сознания человека мы должны использовать другой уровень структурирования данных без которого нет алгебры, классов, методов, алгоритмов, языков программирования, топологии и многого другого.

Цифры являются частным случаем знаков и кодов. Так, например, в двоичной системе счисления цифрами являются два арифметических знака 1 и 0. Из них в соответствии с правилами

системы счисления составляются (формируются) двоичные числа, например: 1101001. Числа могут отражать количественные отношения реальных физических величин и могут меняться с течением времени или пространства, т. е. быть переменными, но отражающими определённую физическую величину, которой можно присвоить имя. Поименованные переменные это уже не цифры, а буквы или слова и здесь мы уже переходим на другой уровень абстрактного мышления от арифметики к алгебре. Этот переход характеризует и процесс обобщения модельных представлений. Именно алгебра рассматривает операции над поименованными количественными данными, называемыми переменными.

Данные могут иметь и логические значения, отображаемые логическими именами "ИСТИНА" (TRUE) и "ЛОЖЬ" (FALSE). Но вместо этих имен для логических значений никто не запрещает использовать знаки "0" и "1", в том числе и для логических сигналов с именами "LOW" и "HIGH". Отношения между логическими переменными, имеющими два значения "ИСТИНА" или "ЛОЖЬ" определяются правилами Булевой алгебры. Операциями Булевой алгебры можно описать любые арифметические операции, поэтому она является более фундаментальной, чем арифметика.

Операции, производимые над данными, тоже обозначаются определёнными знаками или именами, но в отличие от данных они определяют метод и алгоритм обработки данных, т. е. процесс вычисления, преобразования и пересылки данных.

Формальные языки - это математическая абстракция, возникшая как обобщение обычных лингвистических понятий естественных **языков**. Теория **формальных языков** изучает в основном **синтаксис языков** и является фундаментом синтаксически управляемых процессов перевода, к которому можно отнести трансляцию, ассемблирование и компиляцию. **Синтаксис формальных языков** должен обеспечивать возможность **формального** подхода к построению предложений, т. е. он определяет формальные правила составления языковых конструкций.

Семантика формальных языков, сравнительно молодая ветвь теории, занимается способами сопоставления некоего "смысла" словам (цепочкам) **языка**. Необходимость в построении точного математического понятия "смысла" диктуется развитием информационных технологий, прежде всего технологии проектирования компиляторов.

Проведённый ретроспективный обзор развития методов структурирования данных позволяет нарисовать упрощённую схему структурирования данных путём последовательного их кодирования, приведённую на рис. 1.

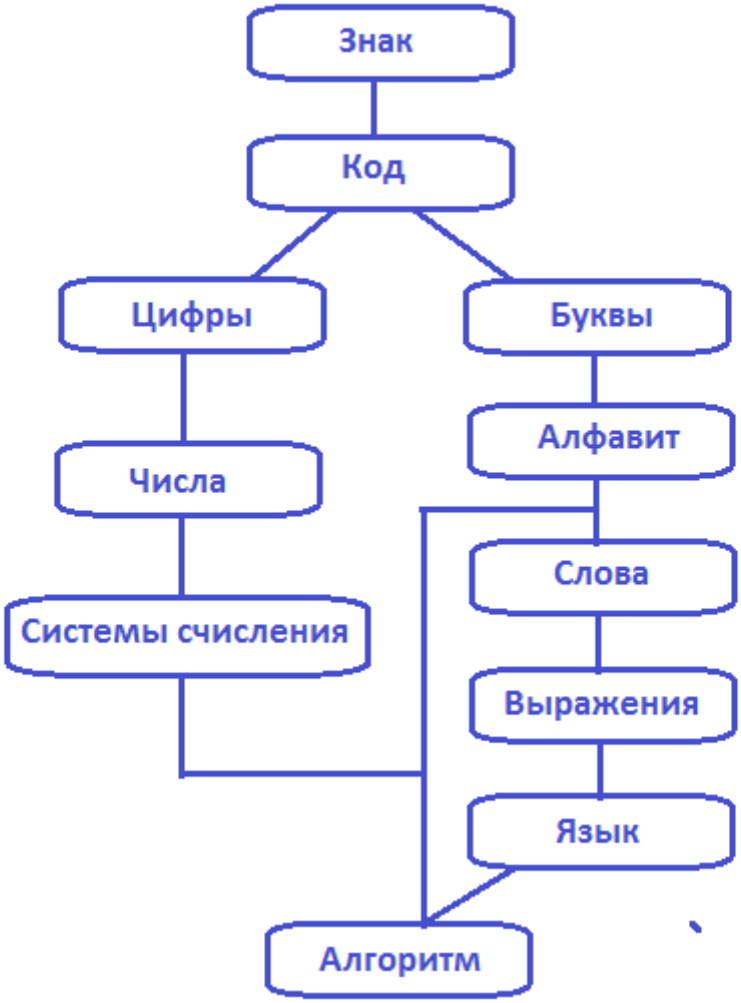


Рис. 1

Следует указать различия в некоторых общепринятых понятиях.

Знаком называют общепринятое вариативное графическое изображение для обозначения какого-либо конкретного смысла. В частности цифры являются знаками для чисел, буквы являются знаками для звуков и, вместе со словами, являются знаками человеческого языка. Примеры знаков приведены на рис. 2:

- Знаки валют обозначают классы денег;
- Буквы это письменные знаки для звуков;
- Иероглифы это письменные знаки образа, понятия, слога или слова в некоторых системах письма. Иероглифы могут означать как отдельные звуки так и слоги;
- Ноты это знаки длительности звука. (А их расположение для обозначения высоты звука — это не знак, а применение знаков в условных обозначениях нотного стана).

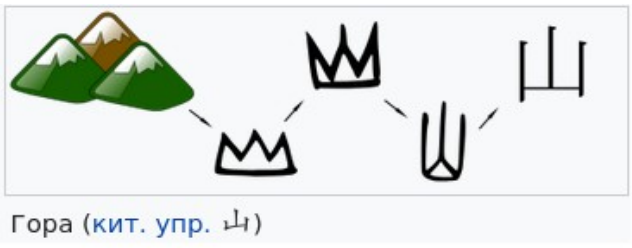
$ € ¥ £ ₽

А Б В Г Д

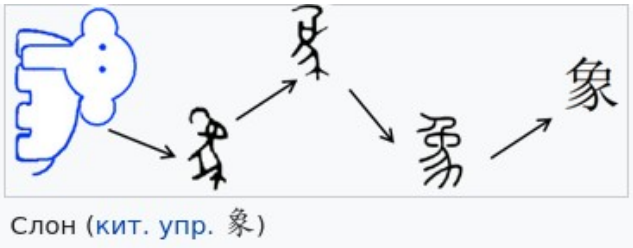
𝅝 𝅗𝅥 𝅘𝅥 𝅘𝅥𝅮

Рис. 2

Иероглифическая письменность считается наиболее ранней - множество первых цивилизаций в Индии, Китае, Центральной Америке и на Ближнем Востоке пользовались подобными логограммами для фиксирования информации. Однако, до нашего времени по сути дожила только китайская ветвь, иероглифы которой несколько тысяч лет эволюционировали от схематичных рисунков к более упрощенным современным формам (рис. 3).



Гора (кит. упр. 山)



Слон (кит. упр. 象)

Рис. 3

Иероглиф не рисунок, а весьма стандартизованная форма записи, где как в конструкторе черты формируют компоненты, компоненты затем объединяются в иероглифы (рис. 4), а уже иероглифы формируют слова.

Иероглиф	Компонент	Черта
字	宀	(1) 丶
		(2) 丶
		(3) 𠂇
	子	(4) 𠂇
		(5) 亅
		(6) 一

Рис. 4

О том как выглядит знак и что он обозначает должна знать определённая группа людей или все люди. Графически один и тот же знак может быть изображён по разному — главное чтобы сохранилась узнаваемая конструкция штрихов.

Символ это общепринятое вариативное изображение для обозначения многозначного образа, понятия, идеи или явления. Проще говоря, символ это знак у которого много смыслов.

Например (Рис. 3):

•Символы религий и философий;
•Астрономические символы;
•Символы зодиака (которые принято называть знаками, но это не знаки)

Рис. 3

Символ также можно изображать по разному, сохраняя узнаваемую топологию.

Компьютерные кодировки знаков и символов

Компьютерное кодирование символов — это система представления букв, цифр и специальных символов с помощью машинных или числовых кодов. В компьютерах кодирование символов играет важную роль, так как позволяет передавать и отображать текст на разных устройствах и платформах.

Самый ранний хорошо известный символьный код с электрической передачей, азбука Морзе, представленный в 1840-х годах, использовал систему из четырех "символов" (короткий сигнал, длинный сигнал, короткий пробел, длинный пробел) для генерации кодов переменной длины.

ASCII. Стандартная кодировка для английского алфавита, содержит 128 символов, а расширенная 256 символов (рис. 4).

Сводная таблица кодов ASCII

ASCII таблица кодов символов Windows (Win-1251)

Dec	Hex	Символ	Dec	Hex	Символ	Dec	Hex	Символ	Dec	Hex	Символ	Dec	Hex	Символ	Dec	Hex	Символ	Dec	Hex	Символ	Dec	Hex	Символ
0	0	спец. NOP	32	20	сцеп. SP (Пробел)	64	40	@	96	60	`	128	80	Ъ	160	A0		192	C0	А	224	E0	а
1	1	спец. SOH	33	21	!	65	41	A	97	61	a	129	81	Ѓ	161	A1	Ў	193	C1	Б	225	E1	б
2	2	спец. STX	34	22	"	66	42	B	98	62	b	130	82	‚	162	A2	ў	194	C2	В	226	E2	в
3	3	спец. ETX	35	23	#	67	43	C	99	63	c	131	83	ѓ	163	A3	J	195	C3	Г	227	E3	г
4	4	спец. EOT	36	24	$	68	44	D	100	64	d	132	84	„	164	A4	¤	196	C4	Д	228	E4	д
5	5	спец. ENQ	37	25	%	69	45	E	101	65	e	133	85	…	165	A5	Ґ	197	C5	Е	229	E5	е
6	6	спец. ACK	38	26	&	70	46	F	102	66	f	134	86	†	166	A6	¦	198	C6	Ж	230	E6	ж
7	7	спец. BEL	39	27	'	71	47	G	103	67	g	135	87	‡	167	A7	§	199	C7	З	231	E7	з
8	8	спец. BS	40	28	(	72	48	H	104	68	h	136	88	€	168	A8	Ё	200	C8	И	232	E8	и
9	9	спец. Tab	41	29	)	73	49	I	105	69	i	137	89	‰	169	A9	©	201	C9	Й	233	E9	й
10	0A	спец. LF	42	2A	*	74	4A	J	106	6A	j	138	8A	Љ	170	AA	Є	202	CA	К	234	EA	к
11	0B	спец. VT	43	2B	+	75	4B	K	107	6B	k	139	8B	‹	171	AB	«	203	CB	Л	235	EB	л
12	0C	спец. FF	44	2C	,	76	4C	L	108	6C	l	140	8C	Њ	172	AC	¬	204	CC	М	236	EC	м
13	0D	спец. CR	45	2D	-	77	4D	M	109	6D	m	141	8D	Ќ	173	AD		205	CD	Н	237	ED	н
14	0E	спец. SO	46	2E	.	78	4E	N	110	6E	n	142	8E	Ћ	174	AE	®	206	CE	О	238	EE	о
15	0F	спец. SI	47	2F	/	79	4F	O	111	6F	o	143	8F	Џ	175	AF	Ї	207	CF	П	239	EF	п
16	10	спец. DLE	48	30	0	80	50	P	112	70	p	144	90	ђ	176	B0	°	208	D0	Р	240	F0	р
17	11	спец. DC1	49	31	1	81	51	Q	113	71	q	145	91	‘	177	B1	±	209	D1	С	241	F1	с
18	12	спец. DC2	50	32	2	82	52	R	114	72	r	146	92	’	178	B2	I	210	D2	Т	242	F2	т
19	13	спец. DC3	51	33	3	83	53	S	115	73	s	147	93	“	179	B3	i	211	D3	У	243	F3	у
20	14	спец. DC4	52	34	4	84	54	T	116	74	t	148	94	”	180	B4	г	212	D4	Ф	244	F4	ф
21	15	спец. NAK	53	35	5	85	55	U	117	75	u	149	95	•	181	B5	µ	213	D5	Х	245	F5	х
22	16	спец. SYN	54	36	6	86	56	V	118	76	v	150	96	–	182	B6	¶	214	D6	Ц	246	F6	ц
23	17	спец. ETB	55	37	7	87	57	W	119	77	w	151	97	—	183	B7	·	215	D7	Ч	247	F7	ч
24	18	спец. CAN	56	38	8	88	58	X	120	78	x	152	98	[illegible]	184	B8	ё	216	D8	Ш	248	F8	ш
25	19	спец. EM	57	39	9	89	59	Y	121	79	y	153	99	™	185	B9	№	217	D9	Щ	249	F9	щ
26	1A	спец. SUB	58	3A	:	90	5A	Z	122	7A	z	154	9A	љ	186	BA	є	218	DA	Ъ	250	FA	ъ
27	1B	спец. ESC	59	3B	;	91	5B	[	123	7B	{	155	9B	›	187	BB	»	219	DB	Ы	251	FB	ы
28	1C	спец. FS	60	3C	<	92	5C	\	124	7C	|	156	9C	њ	188	BC	j	220	DC	Ь	252	FC	ь
29	1D	спец. GS	61	3D	=	93	5D	]	125	7D	}	157	9D	ќ	189	BD	S	221	DD	Э	253	FD	э
30	1E	спец. RS	62	3E	>	94	5E	^	126	7E	~	158	9E	ћ	190	BE	s	222	DE	Ю	254	FE	ю
31	1F	спец. US	63	3F	?	95	5F	_	127	7F		159	9F	џ	191	BF	ї	223	DF	Я	255	FF	я

Рис. 4.

Unicode. Стандарт кодирования символов, позволяющий представить знаки практически всех письменных языков. Юникод и его параллельный стандарт ISO/IEC 10646 «Универсальный набор символов», вместе взятые, представляют собой единый стандарт кодирования символов. Вместо того, чтобы напрямую сопоставлять символы с байтами, Юникод отдельно определяет кодированный набор символов, который сопоставляет символы с уникальными натуральными числами (кодовыми точками), как эти кодовые точки сопоставляются с рядом натуральных чисел фиксированного размера (кодовыми единицами), и, наконец, как эти единицы кодируются в виде потока октетов (байтов). Целью такого разделения является создание универсального набора символов, которые могут быть закодированы различными способами.

Компьютерные кодировки цветов

В компьютерах для работы с цветами используются различные цветовые модели и системы. Наиболее распространённые —RGB (Red, Green, Blue) и HEX. Также цвета могут задаваться в формате CSS.

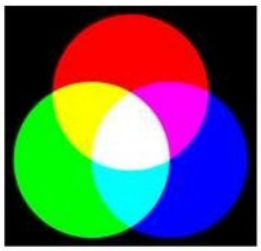
RGB

Основана на смешении трёх основных цветов: красного (R), зелёного (G) и синего (B). Каждый цвет представлен числовым значением от 0 до 255, что позволяет получить более 16 миллионов различных оттенков.

Особенности:

- Нужный оттенок создаётся за счёт разной интенсивности базовых цветов.
- Когда все три цвета смешиваются в максимальной интенсивности (255), получается белый цвет.
- Недостаток RGB — цвета, отображаемые на экране, могут незначительно отличаться на разных устройствах из-за особенностей их цветопередачи.

RGB



Кодировка цветов при глубине цвета 24 бита

Цвет	Двоичный и десятичный коды интенсивности базовых цветов					
	Красный		Зеленый		Синий	
Черный	00000000	0	00000000	0	00000000	0
Красный	11111111	255	00000000	0	00000000	0
Зеленый	00000000	0	11111111	255	00000000	0
Синий	00000000	0	00000000	0	11111111	255
Голубой	00000000	0	11111111	255	11111111	255
Пурпурный	11111111	255	00000000	0	11111111	255
Желтый	11111111	255	11111111	255	00000000	0
Белый	11111111	255	11111111	255	11111111	255

Рис. 5.

HEX

Позволяет точно определить любой оттенок через комбинацию из 6 символов. Каждый HEX-код начинается с решётки (#) и содержит значения для красного, зелёного и синего цветов в диапазоне от 00 до FF.

Особенности:

- Первые два символа отвечают за интенсивность красного цвета, вторая пара — за зелёный, третья — за синий.
- Цифры могут варьироваться от 0 до 9, а буквы — от A до F.
- Например, #FF0000 задаёт чистый красный цвет, а #000000 — абсолютно чёрный.

	A	B	C	D	E	F	G	H	I	J	K
1	Color										
2	HEX value	#000000	#434343	#666666	#999999	#b7b7b7	#cccccc	#d9d9d9	#efefef	#f3f3f3	#ffffff
3	Color										
4	HEX value	#980000	#ff0000	#ff9900	#ffff00	#00ff00	#00ffff	#4a86e8	#0000ff	#9900ff	#ff00ff
5	Color										
6	HEX value	#e6b8af	#f4cccc	#fce5cd	#fff2cc	#d9ead3	#d0e0e3	#c9daf8	#cfe2f3	#d9d2e9	#ead1dc
7	Color										
8	HEX value	#dd7e6b	#ea9999	#f9cb9c	#ffe599	#b6d7a8	#a2c4c9	#a4c2f4	#9fc5e8	#b4a7d6	#d5a6bd
9	Color										
10	HEX value	#cc4125	#e06666	#f6b26b	#ffd966	#93c47d	#76a5af	#6d9eeb	#6fa8dc	#8e7cc3	#c27ba0
11	Color										
12	HEX value	#a61c00	#cc0000	#e69138	#f1c232	#6aa84f	#45818e	#3c78d8	#3d85c6	#674ea7	#a64d79
13	Color										
14	HEX value	#85200c	#990000	#b45f06	#bf9000	#38761d	#134f5c	#1155cc	#0b5394	#351c75	#741b47
15	Color										
16	HEX value	#5b0f00	#660000	#783f04	#7f6000	#274e13	#0c343d	#1c4587	#073763	#20124d	#4c1130

Рис. 6.

CSS

Цвета в CSS можно задавать несколькими способами: с помощью ключевых слов (например, red, blue), шестнадцатеричных значений (например, #ff0000), RGB и RGBA значений (например, rgb(255, 0, 0), rgba(255, 0, 0, 0.5)).

Некоторые свойства цвета в CSS:

•`background-color`
— цвет фона блока или текста;

•`color`
— цвет текста и вариантов его форматирования, например перечёркивания;

•`text-shadow`
— оттенок тени;

•`text-decoration-color`
— цвет зачёркивания и подчёркивания, например для выделения ссылок.

Именованные цвета в Python-tkinter

Tkinter использует именованные цвета. Поэтому для задания кода цвета можно задать rgb код цвета или мы можем написать название. Так вместо '#C0C0C0' можем написать 'silver', вместо '#800000' — 'maroon' и т. д.

Основные положения теории кодирования

Слово код происходит от франц. code или от лат. codex — свод законов. В информационных и компьютерных технологиях (ИКТ) кодом называется система условных знаков (символов) для передачи, обработки и хранения (запоминания) различной информации.

Более строгое математическое определение: код это взаимно однозначное отображение конечного упорядоченного множества символов, принадлежащих некоторому конечному алфавиту, на иное, не обязательно упорядоченное, как правило более обширное множество символов для кодирования передачи, хранения или преобразования информации.

Например, код Морзе, в котором любая буква или символ кодируются последовательностями точек и тире. Иной пример — кодирование букв, чисел и символов последовательностями логических нулей и единиц в вычислительных машинах (компьютерах).

Конечная последовательность кодовых знаков называется словом. Число различных символов, которые используются в словах данного кода, называется его основанием; например, код с основанием 2 называется двоичным. Если все слова имеют одинаковую длину, или количество элементов, — n, то это равномерный n-значный код, например телеграфный код. Если слова имеют переменную длину, то код называется неравномерным например код Морзе. Код называется полным, когда к нему без нарушения его различимости нельзя добавить ни одной новой кодовой комбинации. Полный равномерный n-значный код содержит $т^n$ слов, где т — основание кода. Код, содержащий кодовые комбинации, служащие для отделения одного сообщения от другого, называется кодом с разделительными знаками; код, в котором все без исключения кодовые комбинации символов служат лишь для обозначения элементов сообщения, является кодом без разделительных знаков. Кодовые комбинации, являющиеся разделительными знаками, могут конструироваться либо из специальных кодовых символов, либо из тех же кодовых символов, которые образуют кодовые комбинации, соответствующие определенным элементам сообщения. Иногда бывает удобно разбить элементы сообщения на несколько групп и для каждой из этих групп построить свой код; сигнал о переходе от одного кода к другому подается специальными кодовыми комбинациями (адресами). Совокупность кодов для каждой из групп элементов сообщения вместе с адресными кодовыми комбинациями называется многоадресным, или многопрограммным кодом. Для записи кода чаще всего используют либо цифры и числа (0, 1, 2,... 57, 9276 и т.п.), либо знаки, например + (плюс), — (минус), • (точка), — (тире) и т. д.

В вычислительных машинах (ВМ), иначе называемых компьютерами, каждый код использует знаки своего алфавита. Для большинства кодов алфавиты двухсимвольные либо состоят из букв двухсимвольного алфавита. Основные символы, используемые в ВМ, 0 и 1. Обычно в ВМ используются:

- символьный код (цифро-буквенный) для представления текстовой информации и программ, записанных на алгоритмических языках;
- код команд для представления программ на машинном языке ;
- код чисел для представления числовой информации.

Схема кода, в которой указаны все его основные части и количества двоичных знаков, входящих в каждую из частей, называется форматом кода.

Символьный код — последовательность групп, состоящих из одинакового количества двоичных знаков, обычно состоящим из 8 знаков, называемых байтом, или кратным 8 знакам (байтам). Каждая группа обозначает один символ (букву, условный знак, цифру). Число групп в коде зависит от длины закодированного текста.

Код команды в основной части содержит так называемые коды операций, определяющие действия ВМ по данной команде, и структуру остальной части команды, куда могут входить коды адресов (операндов) и искомых результатов, иногда коды самих операндов и коды др. частей команды. Коды команд определяются установленной системой команд.

Код чисел зависит от формы представления чисел в ВМ. Число в форме с фиксированной запятой представляется с помощью одного из трёх кодов: прямого, обратного и дополнительного.

Код числа, представленного в форме с плавающей запятой , записывается в виде упорядоченной пары кода мантиссы и кода порядка; при этом как мантисса, так и порядок могут быть представлены в одном из указанных трёх кодов. Прямой код обычно используется при хранении чисел в запоминающем устройстве , а обратный и дополнительный коды — при выполнении над числами арифметических и некоторых др. операций. При пересылках из запоминающего устройства в арифметико-логическое (АЛУ) и обратно числа перекодируются. Все три кода состоят из кода знака (число отведённых разрядов l), кода целой части (m) и кода дробной части (n) числа. Сумма d =l+m+n называется длиной кода. Как правило, в ЦВМ или в её устройствах I, m и n фиксированы. В случае целых чисел n=0, для правильных дробей обычно m=0, когда все числа одного знака, l=0. Для положительных чисел код знака обозначается последовательностью нулей, для отрицательных — последовательностью единиц. Для положительных чисел прямой, обратный и дополнительный коды совпадают. В прямом коде отрицательных чисел меняется только код знака; в обратном коде цифры числа заменяются их дополнениями до 1 (т. е. 0 заменяется на 1, а 1 на 0). Дополнительный код отрицательного числа отличается от обратного кода тем, что после замены цифр производится сложение результата с d-разрядным числом, все разряды которого, кроме младшего, содержат нули, причём перенос из старшего разряда при сложении не выполняется. Например, число в двоичной системе счисления равно +11,01. Пусть задано l=2, m=3, n=4; дополняя целую и дробную части нулями, запишем число в виде +011,0100. Прямой, обратный и дополнительный коды заданного числа одинаковы — 00 011 0100. Для отрицательного числа —11,01 прямой код имеет вид 11 011 0100, обратный код— 11 100 1011 и дополнительный — 11 100 1100. Выбор между обратным и дополнительным кодом обусловливается конструкцией и логикой ЦВМ.

Физическая интерпретация кодов, сигналы.

В технике каждый кодовый знак является условным обозначением некоторого элементарного сигнала, обладающего какими-либо физическими параметрами (сигнальными признаками), которые могут принимать несколько различных значений. В ИКТ это называется интерпретацией, т. е. каждый кодовый знак реализуется некоторой установленной физической величиной. Для электрических сигналов такими признаками могут служить амплитуда тока или напряжения, полярность или длительность электрических импульсов (посылок), периодичность их следования и др. Коды, применяемые в телемеханике, в системах связи и автоматического управления, в вычислительной технике, обычно представляют собой набор комбинаций из электрических импульсов и пауз между ними, что эквивалентно изображению значений кодируемой величины в виде двоичных чисел — наборов, состоящих из 0 и 1. Количество импульсов в комбинации или разрядов в эквивалентном двоичном числе определяет значность кода. Последовательность элементарных закодированных символов принято называть кодовым сообщением или кодовой посылкой. Иногда последовательность закодированных символов известной длины называют кодовым словом, или кодовым кадром. Выбор кода определяется условиями передачи, обработки или хранения информации и связан главным образом с наиболее эффективным использованием каналов связи, обеспечением необходимой помехоустойчивости передачи и т.п. С целью улучшения помехоустойчивости коды усложняются: к так называемым информационным знакам добавляются дополнительные — контрольные (проверочные). По такому принципу строятся коды обнаружения и исправления ошибок. В телемеханике (интерфейсах) для представления и передачи отдельных элементов кода используются сигналы с различными признаками по амплитуде, частоте, полярности, фазе, длительности и др. Так, в двоичном коде при полярных признаках элемент «0» кодируется импульсом отрицательной, а «1» — положительной полярности; широтные признаки означают

различие в длительности импульсов либо в паузах между ними и так далее. Если для передачи сообщений используются не все возможные комбинации элементов кода, то применяют специальные методы, позволяющие при приеме обнаруживать и исправлять искажения (ошибки) в переданных элементах кода, что повышает достоверность передачи информации. Выбор системы кодирования сообщения, способа его передачи и методов повышения достоверности передаваемой информации определяется конкретными условиями работы телемеханической системы или интерфейса, важностью объектов, свойствами каналов связи, применяемой аппаратурой и др.

Физическая форма кода зависит от характера используемого носителя информации и даже для одной ВМ может допускать несколько вариантов. Например, на письменных документах код представляется в виде цифр и (или) букв русского либо латинского алфавита, на перфокартах — сочетанием пробитых и непробитых участков, на магнитных лентах, барабанах, дисках и карточках— в виде конфигураций из намагниченных участков, в ячейках оперативной и флэшпамяти — в виде групп элементарных ячеек электрического заряда, каждая из которых находится в одном из двух возможных для неё состояний (заряжена или разряжена).

Алгоритм

В настоящее время существует много вариантов значения слова "алгоритм", но в поисках подсказок, о том как можно работать с алгоритмами суть раскрывает только теория моделирования.

Так, например, формулировка "конечная совокупность точно заданных правил решения произвольного класса задач" говорит что есть возможность как-то "точно задать правила" из них собрать "совокупность" и этой совокупностью "решить" некоторый "класс задач".
К этому определению сразу возникает масса вопросов.

- •Что такое правило?
- •Как, кому и для кого это правило можно задать?
- •Что есть объединение совокупностью?
- •Каким образом правила соотносятся с задачей?
- •Что формирует класс задачи?
- •Определяется ли способ формирования совокупности правилами и задачами?

и т. д.

Другая формулировка "набор инструкций, описывающих порядок действий исполнителя для решения некоторой задачи" говорит что есть "исполнитель", который может выполнять некоторые "действия", и при некотором "порядке" выполнения этих "действий" "решается задача". Вопросов не стало меньше.

Поставим себе меньшую, но очень важную с точки зрения моделирования задачу. Давайте попробуем сформулировать, что делает алгоритм способом решения задач, и какие процессы являются для него "действиями". Рассмотрим "Действие" и его признаки, которые опущены в указанных выше определениях алгоритма, но являются очень важными для понимания сути алгоритма.

Рассмотрим определение алгоритма, говорящее, что он — приводящая к решению задачи последовательность действий. Любой программный код состоит из частей. Такими частями являются функции, классы, модули. Когда программист пишет текст функции — он занимается написанием алгоритма.

Раньше алгоритм создавали в виде блок схем и полуавтоматически компилировали их в машинные коды. Сейчас нет необходимости быть художником и компилятором для написания программы. Текст функции — это запись алгоритма в текстовом виде — его текстовая блок-схема. Здесь можно вспомнить Scratch, где используется визуальное создание блок-схемы алгоритма без

написания текста. Scratch - это визуальная среда программирования для обучения школьников младших, средних и старших классов, построенная на интуитивно понятных ребенку принципах, где результаты действий визуализированы, что делает работу с программой понятной, интересной и увлекательной для школьников. Способ записи алгоритма сейчас не так важен.

Важно, что в написании алгоритма функции можно использовать вызовы других функций, которые вы или другой программист уже написал до этого момента. Вспоминая фразу "последовательность действий, приводящая к решению задачи", можно отметить, что функции, написанные ранее, являются вашими "действиями". То есть "действия" могут быть функциями. Если обобщать, то "действия" могут быть алгоритмами.

Если "действие = алгоритм", то определение можно попробовать переписать рекурсивно "алгоритм — это приводящая к решению задачи последовательность использования существующих алгоритмов". Рекурсивные определение не самое простое, что можно записать в словаре обычного человека. Но для программиста и математика эта форма знакома. Мы умеем с ней работать, и это даёт нам преимущество в рассмотрении разных задач, разбиваемых на подобные себе подзадачи. Так давайте воспользуемся этим преимуществом.

Чтобы разрешить рекурсию нам необходимо найти:

- терминальное условие выхода из рекурсии — минимальное неделимое "действие" (атомарный алгоритм), которое можно использовать в разработке алгоритма;
- способ перехода от текущего уровня рекурсии (набора "действий") к следующему уровню (алгоритму).

Действие

Для начала рассмотрим "действие" и попробуем найти причину, обеспечивающую возможность использования существующего "действия" для создания нового алгоритма. Этой причиной является возможность повторного использования "действия" с получением тождественного результата. Только тогда разработанный с использованием этого "действия" алгоритм решения некоторой задачи будет одинаково решать эту задачу снова и снова. Мы нащупали важные законы нашего мира, в котором:

- существуют "действия", главным свойством которых является одинаковость результатов их исполнения в разные моменты времени и в разных местах,
- существует возможность создать такую схему использования нескольких "действий", которая сформирует из них новое "действие", которое мы назвали алгоритмом.

Какие признаки "действия" кроме повторимости делают возможным его использование в создании алгоритма? Что является терминальным неделимым "действием"? Чтобы ответить на этот вопрос стоит рассмотреть разные примеры "действий" из практического опыта. Все встречали их много раз. Это и сложение, и умножение, и установка цвета пикселя на экране. Но мы знакомы с ними и вне программирования. Вся наука основывается на повторяемых явлениях, т.е. на обобщённых закономерностях окружающего мира определяемых методами теории размерности в моделировании.

Закон гравитации, описывающий повторяющееся явление падения яблока, тоже может стать действием. Ведь любое яблоко будет падать на землю? Значит этот процесс можно использовать в качестве "действия"!

Рассмотрим, что происходит при выполнении "действия". Например, во время падения яблока с ветки яблони на землю. В этом процессе происходит несколько изменений. Если вспомнить школьную физику и рассмотреть ситуацию в системе отсчета, привязанной к Земле, то сила гравитации вызывает изменение скорости яблока, разгоняя его. При этом в процессе отмечается еще одно важное изменение — уменьшается расстояние между яблоком и Землей.

В рамках примера процесса "Земля-Яблоко" можно отметить у "действия" следующие признаки:

•наличие процесса изменения (расстояния, как одного из параметров движения объектов);

•наличие объектов, взаимодействие которых вызывает такие изменения;

•наличие локальности процесса, то есть существование значения расстояния между объектами, с превышением которого их взаимодействие может практически перестать вызывать процесс изменения, что делает несущественным использование "действия" (например можно считать, что Земля и гипотетическое яблоко, далеко от Солнца и других планет солнечной системы).

Рассмотрим с этими признаками разные области и процессы, выделяя в них примеры "действий" и контролируя особенности указанных признаков в описании структуры "действия".

Физические процессы

Для физических систем, процессы которых мы наблюдаем в нашем мире, характерные объекты и изменения опираются на фундаментальные взаимодействия и потому их достаточно просто выделить по аналогии с гравитационным взаимодействием. Например, для системы из протона и электрона или системы двух протонов.

Отдельно от этих простых взаимодействий двух объектов стоят многокомпонентные процессы, например, ядерные реакции (рис. 7), по структуре "действия" близкие к химическим процессам, рассматриваемым далее.

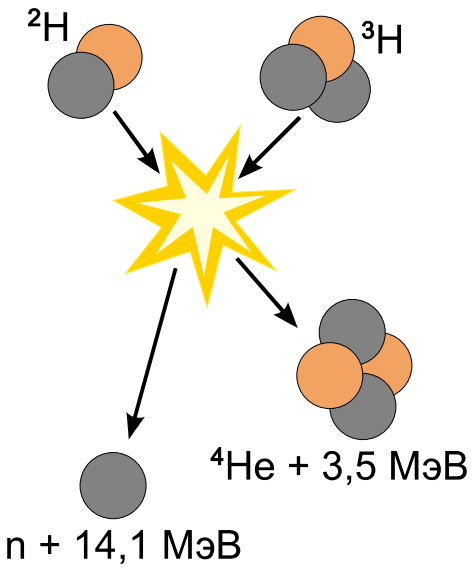


Рис. 7.

Сложны и процессы описываемые суммарным взаимодействием большого числа элементов, например, "идеальный газ". Пока отложим их рассмотрение и сосредоточимся на самых простых примерах.

Химические процессы

Перейдем к следующей большой области — химическим процессам. Химические реакции, например, NaOH+HCl>NaCl+H_2O по признаку своей повторимости так же являются "действиями". Объектами в них являются атомы и молекулы.

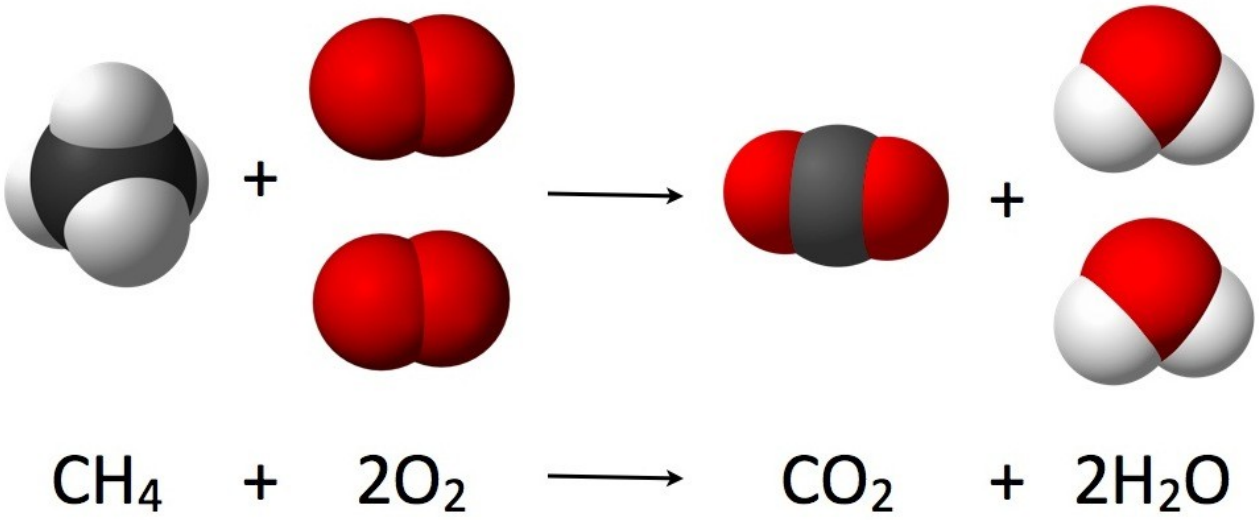


Рис. 8

Для описания происходящих изменений необходимо немного преобразовать "физические" изменения. Так изменения параметров движения в совокупности дают нам изменение температуры в ходе химической реакции. А среди изменений расстояний между молекулами мы, игнорируя броуновское движение, можем выделить фиксацию расстояния в виде повторимого формирования и разрушения связей между частями взаимодействующих молекул. Локальность для химической реакции тоже существует — это отсутствие реакции при нахождении гидроксида натрия и соляной кислоты в разных пробирках и наличие реакции при соприкосновении веществ. Конечно, в "химической" области "действий" есть интегративные (эмерджентные) свойства, не сводящиеся к молекулам, например, фотохимические реакции, где к рассматриваемым объектам необходимо добавить фотоны.

Математические процессы в абстрактных объектах

Следующей областью выберем "действия" из известных нам абстрактных объектов. Самые яркие их представители — математические процессы. Для простоты рассмотрим в качестве "действия" достаточно элементарную операцию — сложение. А примером этого "действия" выберем сложение с помощью управляющего устройства машины Тьюринга (рис. 9).

Историческое значение машины Тьюринга в том, что это первая математическая модель универсальных вычислений. Другими словами, всё, что можно вычислить, описывается как машина Тьюринга. Это не единственная модель вычислений, но, пожалуй, самая известная и популярная. И ещё это описание работы компьютера, которое Тьюринг дал до появления компьютеров. Его машина осуществила связь между абстрактными символами и физическим миром. Также новаторство

Тьюринга было в том, что его машина использовала двоичную систему во времена, когда преобладала десятичная. Тьюринг пошёл не по привычному пути формул и доказательств, а начал с описания машины, которая совершает несколько простых операций.

Рис. 9.

Машина Тьюринга — это абстрактная вычислительная машина, мысленный эксперимент для решения проблемы математической логики. Она состоит из трёх элементов:

- бесконечной ленты с ячейками;
- автомата или головки для чтения и записи;
- программы.

Машина работает следующим образом: головка может прочитать содержимое конкретной ячейки, стереть, перезаписать и/или передвинуться на ячейку влево или вправо и повторить считывание/запись.

Действия головки определяются программой. Она записывается в виде таблицы, где есть внешний и внутренний алфавиты и таблица переходов между ними. Внешний алфавит — это конечное множество, элементы которого называются буквами или символами. Внутренний алфавит — это конечное множество состояний головки. Таблица переходов описывает поведение головки. Команда, которую должна выполнить головка, определяется пересечением внешнего и внутреннего алфавитов в таблице.

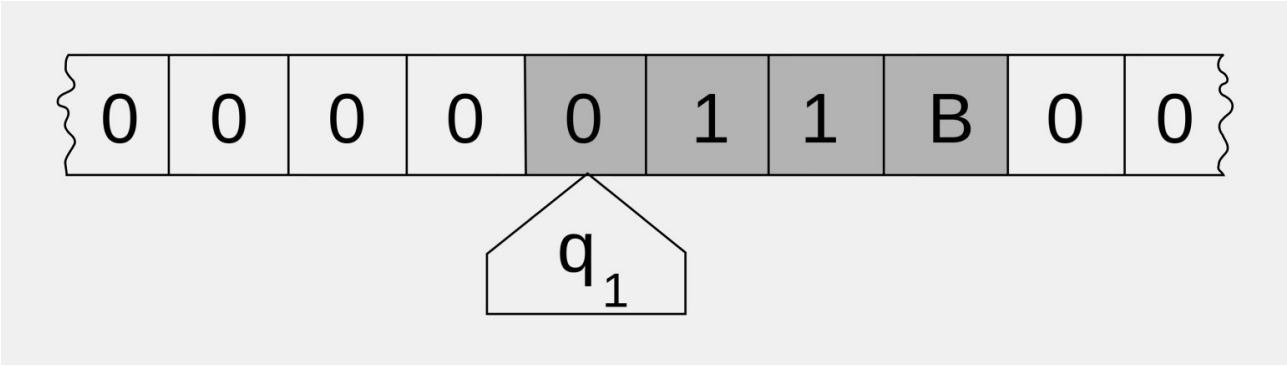


Рис. 10.

На рис. 10 изображена лента, входная информация (11B) расположена так, что в каждой клетке по одному символу. До и после неё — нулевые ячейки. Исходное состояние головки — q1. Это состояние запускает работу машины. Головка может сдвинуться влево и вместо 0 записать цифру или букву. Также она может сдвинуться вправо и перезаписать 1 другой цифрой или буквой. Или передвинуться ещё на ячейку влево/вправо без перезаписи.
Головка может также остаться на месте и ничего не делать. Выполнение операций прекращается после того, как головка считывает пассивное состояние — q0.

При сложении "ячейки ленты" машины Тьюринга будут содержать слагаемые. При этом остаётся и признак локальности как возможность взаимодействия управляющего устройства только с текущей ячейкой ленты, и признак изменения параметров объектов, который можно описать как изменение состояния ячеек.

Подробное описание исходных и результирующих состояний объектов, а так же "действий" производящих эти изменения для сложения, исполняемого машиной Тьюринга, оставим за рамками нашего рассмотрения. Но упомянем, что перейдя к **машине** мы передаём "действия" абстрактных процессов от человека машине , что является главным способом для создания формальных методов работы с алгоритмом. Можно поставить себе целью упрощение каждой составляющей алгоритма до состояния, когда её выполнение можно будет полностью поручить компьютеру. Тогда в определении алгоритма не останется тёмных мест, и многочисленные вопросы к понятию алгоритмов найдут свои ответы. Пока формализован только исполнитель. Скажем спасибо за это Тьюрингу и вспомним про "действие", формализацией которого и занимается информационные и компьютерные технологии.

Литература:

1. Габидулин Э. М., Пилипчук Н. И. Лекции по теории информации — МФТИ, 2007. - 214 с. — ISBN 978-5-7417-0197-3
2. Цымбал В. П. Теория информации и кодирование. — Киев: Выща Школа, 1977. — 288 с.