Лабораторная работа № 6. «Поразрядная сортировка»

1.1. Цель работы

Ознакомиться и реализовать алгоритмы поразрядных сортировок для сортировки массивов строк.

1.2. Поразрядная сортировка

На практике сортировка применяется в основном к каким-либо структурам данных и выполняется по определенному ключу. Природа ключей может быть очень сложной. Совсем не обязательно, что на каждом шаге обрабатывается весь ключ.

Рассмотрим пример: известен автор – по трем первым буквам выбирается ящик каталога и в нем ведется поиск. Для того, чтобы сортировки были такими же эффективными будем рассматривать структуру ключей. Т.е. рассмотрим ключи как последовательности –

- Строки - последовательности символов
- Двоичные числа – последовательности битов
- Десятичные числа – последовательности десятичных разрядов.

Каждый элемент такой последовательности имеет строго определенный размер. Сортировки, основанные на обработке за раз одного такого элемента называются поразрядными *(radix).*

Например, сортировка абонентов библиотеки – библиотекарь выставляет карточку клиента в отделение, на котором обозначена одна буква фамилии (всего отделений может быть 29). Если карточек много, то внутри отделения возможна сортировка по второй букве и т.д.. Это пример поразрядной сортировки с основанием 29.

Основа поразрядной сортировки – извлечь *i*- тый объект последовательности.

Существуют два базовых подхода: первый анализирует объекты слева направо (первыми обрабатываются наиболее значащие цифры). Такая сортировка называется *MSD* (*most significant digit*)- сортировкой.

Второй подход анализирует цифры справа налево (первыми обрабатываются меньшие разряды) – *LSD* (*last significant digit*).

1.3. Средства Си для реализации поразрядных сортировок

1.3.1. Выделение двоичного разряда

При выделении *i*- того двоичного разряда можно использовать следующий алгоритм:

1. Выполнить сдвиг вправо на *i.*
2. Наложить на исходное число маску 1 (выполнить логическое умножение).
3. Полученное число вернуть в качестве результата.

Функция, выделяющая разряд может быть записана следующим образом:

```
// x - исходное число, d – номер разряда
int digit(int x, int d)
{ int k = x>>d;
  k = k&1;
  return k;}
```

1.4. MSD - сортировка

1.4.1. Двоичная быстрая сортировка

Предположим, следующие латинские буквы имеют коды:

a 000	**b 001**	**c 010**	**d 011**
e 100	**f 101**	**g 110**	**h 111**

Дана последовательность *e f d e c g h d e e*, необходимо упорядочить ее. Просмотрим последовательность слева направо, и найдем первый ключ, который начинается с бита 1, далее просмотрим последовательность справа налево и найдем ключ, начинающийся с бита 0. Обменяем их местами и будем выполнять этот процесс пока индексы просмотров не пересекутся:

100	011	011
101	**101**	010
011	011	011
100	100	100
010 →	**010** →	101
110	110	110
111	111	111

011	100	100
100	100	100
100	100	100

Первый шаг сортировки закончен. Получено два подмассива: с первой единицей и с первым нулем.
Рекурсивно применим ту же самую процедуру к полученным массивам по следующему разряду:

011
010
011
100
101 Без изменений, все элементы разряда равны 1.
110
111
100
100
100

По третьему разряду:

011	010	
010	011	
011	011	
100	100	
101	101	Сортировка первого подмасссива
110	110	закончена, так как рассмотрен
111	111	последний разряд
100	100	
100	100	
100	100	

Начнем сортировать по второму разряду второй подмассив:

011	010	010
010	011	011
011	011	011
100	100	100
101	101	101
110	100	100
111	**111**	100
100	100	100
100	**100**	111
100	110	110

Сортировка по третьему разряду двух полученных подмассивов:

010	010	010
011	011	011
011	011	011
100	100	100
101	100	100
100	100	100
100	100	100
100	101	101
111	**111**	**110**
110	**110**	**111**

Сортировка закончена, получен отсортированный массив:
c d d e e e ef g h.

1.4.2. Поразрядная MSD сортировка

Если в быстрой двоичной сортировке разделение происходит на два подмассива (0 и 1), то *MSD* сортировка обобщает понятие поразрядной сортировки по произвольному основанию *R* – происходит разделение всего массива на *R* подмассивов.

Таким образом, в функции будет выполнено *R* рекурсивных вызовов.

Для больших значений *R* можно использовать следующую схему передвижения элементов к своему подмассиву. Создаются два вспомогательных массива – массив счетчиков для каждого из значений *R* и временный массив для хранения передвинутых элементов. Просматривается исходный массив первый раз и подсчитывается сколько раз встретилась буква “*а”* среди первых букв слов, буква *“б”* и т.д.. При втором проходе элементы из исходного массива записываются во

временный, используя те точки разделения, которые получены при первом проходе. Для вычисления точек разделения можно использовать следующий алгоритм:

3 4 2 7 9 → 3 7 9 16 25

Каждый элемент массива сложим с предыдущим. Далее просмотрим основной массив – если первая буква *"a"*, то ставим ее на *2 (3-1)* место и первый элемент массива разделений уменьшаем: *2 7 9 16 25* и т.д..

Основная часть работы происходит на первом же этапе разделения. Можно улучшить алгоритм *MSD*-сортировки, если для сортировки подмассивов маленькой размерности использовать алгоритм простой сортировки.

1.4.3. Поразрядная LSD сортировка

Сортировка работает только в случае устойчивого способа перестановки элементов. К устойчивым сортировкам относится сортировка подсчетом (при условии, что элементы массива на этапе распределения по своим местам просматриваются с конца массива), поэтому можно применить его. Рассмотрим на примере:

01001	10010	10100	00000	00000	00000	0
10010	10110	00000	10000	10000	00011	3
11011	10100	10000	01001	10010	01001	9
10110	00000	01001	10010	00011	10000	16
00011	10000	10010	11011	10100	10010	18
10100	01001	10110	00011	10110	10100	20
00000	11011	11011	10100	01001	10110	22
10000	00011	00011	10110	11011	11011	27

Сортировка не рекурсивная.

На основании *R* выполняется аналогичным образом и остается такой же не рекурсивной.

1.5. Порядок выполнения

Если Ваш вариант – реализация **поразрядной быстрой сортировки**:

1. Сортируемые данные – массив целых беззнаковых чисел (unsigned int). Элементы массива задаются случайным образом, диапазон значений массива специально не оговаривается.

2. Сортировка выполняется для массивов размерности 100 - 1000

3. Время выполнения реализованной сортировки сравнивается с временем выполнения сортировки, реализованной в **четвертой** лабораторной работе.

Если Ваш вариант – реализация **MSD/LSD** сортировки:

1. Сортируемые данные – массив строк. Размер массива и длина строки задаются с клавиатуры. Все строки в массиве имеют одинаковую длину. Сами строки генерируются случайным образом. Для генерации строк случайным образом используйте коды ASCII-символов (Приложение А).

2. Внутри разряда используется сортировка подсчетом.

Приложение А. Коды ASCII-символов

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	спец. SP (Пробел)	64	40	@	96	60	`	128	80	Ђ	160	A0		192	C0	А	224	E0	а
1	1	спец. SOH	33	21	!	65	41	A	97	61	a	129	81	Г	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	Ј	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. Табуляция	41	29	)	73	49	I	105	69	i	137	89	‰	169	A9	©	201	C9	Й	233	E9	й
10	0A	спец. LF (Возвр. каретки)	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 (Новая строка)	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	І	210	D2	Т	242	F2	т
19	13	спец. DC3	51	33	3	83	53	S	115	73	s	147	93	“	179	B3	і	211	D3	У	243	F3	у
20	14	спец. DC4	52	34	4	84	54	T	116	74	t	148	94	”	180	B4	ґ	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	ј	220	DC	Ь	252	FC	ь
29	1D	спец. GS	61	3D	=	93	5D	]	125	7D	}	157	9D	ќ	189	BD	Ѕ	221	DD	Э	253	FD	э
30	1E	спец. RS	62	3E	>	94	5E	^	126	7E	~	158	9E	ћ	190	BE	ѕ	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