1. **Suppose that you are sorting an array containing the following 8 keys (the subscript is not part of the key; its purpose is to uniquely identify each of the equal keys).**

$$B_0 \quad A_0 \quad B_1 \quad A_1 \quad B_2 \quad A_2 \quad B_3 \quad A_3$$

Give the process of sorting the above 8 keys between 'A' and 'B' by using *Key-indexed counting*.

		count frequencies	compute cumulates	move items	copy back
i	**a[i]**	**count[r]**	**count[r]**	**aux[i]**	**a[i]**
0	B_0				
1	A_0				
2	B_1				
3	A_1				
4	B_2				
5	A_2				
6	B_3				
7	A_3				

2. **Consider the *first call* to key-indexed counting when running LSD string on the input array a[ ] of 13 strings. Recall that key-indexed counting is comprised of four loops. Give the contents of the integer array count[ ] after each of the first three loops; then, give the contents of the string array after the fourth loop.**

			count frequencies	Compute cumulates	move items	copy back
i	**a[i]**	**r**	**count[r]**	**count[r]**	**aux[i]**	**a[i]**
0	now	0				
1	for	1				
2	tip	2				
3	ilk	3				
4	dim	4				
5	tag	5				
6	jot	6				
7	sob	7				
8	nob	8				
9	sky	9				
10	hut	10				
11	ace					
12	bet					

3. **The column on the left is the original input of 24 strings to be sorted; the column on the right are the strings in sorted order; the other 5 columns are the contents at some intermediate step during one of the 3 radix-sorting algorithms listed below. Match up each algorithm by writing its number under the corresponding column. You may use a number more than once.**

0	null	byte	cost	byte	java	byte	byte
1	tree	cost	lifo	cost	load	cost	cost
2	lifo	edge	list	edge	find	miss	edge
3	list	find	miss	flip	tree	hash	find
4	miss	flip	hash	find	byte	java	flip
5	hash	hash	java	hash	edge	load	hash
6	java	java	load	java	trie	leaf	java
7	next	lifo	leaf	lifo	type	flip	lazy
8	load	list	flip	list	leaf	link	leaf
9	leaf	load	link	load	hash	list	left
10	flip	leaf	byte	leaf	path	edge	lifo
11	path	lazy	edge	lazy	sink	lazy	link
12	byte	left	lazy	left	link	left	list
13	edge	link	left	link	rank	find	load
14	lazy	miss	find	miss	null	lifo	miss
15	trie	null	next	null	lifo	next	next
16	find	next	null	next	flip	null	null
17	left	path	type	path	swap	type	path
18	type	rank	sink	rank	miss	sink	rank
19	sink	sink	trie	sink	list	trie	sink
20	link	swap	swap	swap	next	swap	swap
21	swap	tree	path	tree	left	path	tree
22	cost	trie	rank	trie	cost	rank	trie
23	rank	type	tree	type	lazy	tree	type
	A						E

A. Original input
B. LSD radix sort
C. MSD radix sort
D. 3-way string quicksort (no shuffle)
E. Sorted