1. Consider the following hash function for String
$(s[0]*R^{(n-1)} + s[1]*R^{(n-2)} + ... + s[n-1])\%M$, Where R = 256 and M=255
Show that is a bad hash function because any permutation of letters within a string hashes to the same value

Answer:

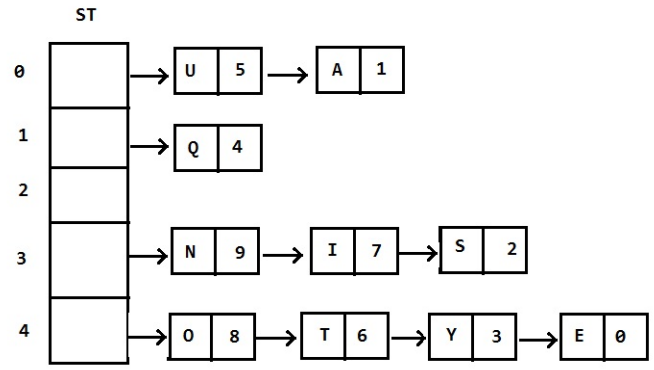
Because $R^K \% M = (255+1)^K \% 255 = 1$, where k>=0

Thus $(s[0]*R^{(n-1)} + s[1]*R^{(n-2)} + ... + s[n-1])\%M = (s[0] + s[1] + ... + s[n-1])\%M$, which shows the order of letters in the string does not matter in this hash function

2. Insert the keys E A S Y Q U T I O N (with associated values 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, respectively) in that order into an initially empty table of M = 5 lists, using separate chaining. Use the following hash function char.hashCode() % M. For search miss, insert the key value pair at the beginning of the linked list.

Answer:

Key	Value	Key.hashCode()	Array Index
E	0	69	4
A	1	65	0
S	2	83	3
Y	3	89	4
Q	4	81	1
U	5	85	0
T	6	84	4
I	7	73	3
O	8	79	4
N	9	78	3



3.Describe the effect of using the following method, public int hashCode() { return 17; }

Answer:

The implementation does follow this rule: equal objects have the same hash value. However, the implementation is a very bad one, because it does not distribute keys at all. If this implementation is used in the previous exercise, all the key value pairs will end up in the same linked list at array index 2, which can result in worst case (linear) running time for search and insertion.

4. Insert the keys E A S Y Q U T I O N (with associated values as 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, respectively) in that order into an initially empty table of size M =16 using linear probing without resizing. Use the following hash function char.hashCode() % M

Answer:

Key	Value	Key.hashCode()	Initial Array Index
E	0	69	5
A	1	65	1
S	2	83	3
Y	3	89	9
Q	4	81	1
U	5	85	5
T	6	84	4
I	7	73	9
O	8	79	15
N	9	78	14

The final result after the insertions is:

index	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
keys		A	Q	S	T	E	U			Y	I				N	O
vals		1	4	2	6	0	5			3	7				9	8

5. Which of the following scenarios leads to expected linear running time for a random search hit in a linear-probing hash table?

a) All keys hash to the same index.
b) All keys hash to different indices.
c) All keys hash to an even-numbered index.
d) All keys hash to different even-numbered indices

Answer:

a and c