COMP3506/COMP7505—Algorithms and Data Structures

School of Information Technology and Electrical Engineering

Week 7 Tutorial Sample Solutions

Question 1.
Show the result of the following operations on a Map that has been implemented using a hash-table with a list-based map in each bucket-array position. Fill in the table and **also** give a representation of the hash-table.

a) Show the result of the following operations assuming a hash code of 2i + 7 and using a mod compression function. The size of the bucket array is 9.

operation	hash code	compression fn	bucket index
put(12, e)	2 * 12 + 7 = 31	31 mod 9 = 4	4
put(44, e)	2 * 44 + 7 = 95	95 mod 9 = 5	5
put(13, e)	2 * 13 + 7 = 33	33 mod 9 = 6	6
put(89, e)	2 * 89 + 7 = 185	185 mod 9 = 5	5
remove(44)	95	5	5
remove(89)	185	5	5

After put(12, e)

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]
				12-e				

After put(44, e)

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]
				12-e	44-e			

After put(13, e)

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]
				12-e	44-e	13-e		

After put(89, e)

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]
				12-e	44-e	13-e		
					89-e			

After remove(44)

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]
				12-e	89-e	13-e		

After remove(89)

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]
				12-e		13-e		

b) Show the result if each entry of the bucket array stored a single entry and collisions were handled by linear probing.

operation	hash code	compression fn	bucket index
put(12, e)	2 * 12 + 7 = 31	31 mod 9 = 4	4
put(44, e)	2 * 44 + 7 = 95	95 mod 9 = 5	5
put(13, e)	2 * 13 + 7 = 33	33 mod 9 = 6	6
put(89, e)	2 * 89 + 7 = 185	185 mod 9 = 5	5, 6, 7
remove(44)	95	5	5
remove(89)	185	5	5, 6, 7

After put(12, e)

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]
				12-e				

After put(44, e)

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]
				12-e	44-e			

After put(13, e)

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]
				12-e	44-e	13-e		

After put(89, e)

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]
				12-e	44-e	13-e	89-e	

After remove(44)

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]
				12-e	AVAL	13-e	89-e	

After remove(89)

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]
				12-e	AVAL	13-e	AVAL	

c) Show the result if each entry of the bucket array stored a single entry and collisions were handled by double hashing using the secondary hash code of 7 - (k mod 7) with the same compression function used before.

operation	hash code (*i*)	*d(k)*	compression fn	bucket index
put(12, e)	2 * 12 + 7 = 31	7 – (12 mod 7) = 2	31 mod 9 = 4	4
put(44, e)	2 * 44 + 7 = 95	7 – (44 mod 7) = 5	95 mod 9 = 5	5
put(13, e)	2 * 13 + 7 = 33	7 – (33 mod 7) = 2	33 mod 9 = 6	6
put(89, e)	2 * 89 + 7 = 185	7 – (89 mod 7) = 2	? mod 9 = 5	5, 7
remove(44)	95	5		5
remove(89)	185	2		5, 7

Handles collisions by placing an item in the first available cell of the series:
($i + j.d(k)$) mod N, *for* $j=0, 1, .., N-1$.

($i + j.d(k)$) mod N = (31 + 0.2) mod 9 = 4 ($j=0$)

After put(12, e)

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]
				12-e				

($i + j.d(k)$) mod N = (95 + 0.5) mod 9 = 5 ($j=0$)

After put(44, e)

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]
				12-e	44-e			

($i + j.d(k)$) mod N = (33 + 0.2) mod 9 = 6 ($j=0$)

After put(13, e)

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]
				12-e	44-e	13-e		

($i + j.d(k)$) mod N = (185 + 0.2) mod 9 = 5 ($j=0$) (collision)
($i + j.d(k)$) mod N = (185 + 1.2) mod 9 = 7 ($j=0$)

After put(89, e)

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]
				12-e	44-e	13-e	89-e	

($i + j.d(k)$) mod N = (95 + 0.5) mod 9 = 5 ($j=0$)

After remove(44)

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]
				12-e	AVAL	13-e	89-e	

($i + j.d(k)$) mod N = (185 + 0.2) mod 9 = 5 ($j=0$) (found AVALIABLE)
($i + j.d(k)$) mod N = (33 + 1.2) mod 9 = 7 ($j=0$)

After remove(89)

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]
				12-e	AVAL	13-e	AVAL	

Question 2

When the load factor ($\alpha = n / N$) of a hash table passes a critical point the hash table is increased in size - increasing N, the size of the array, decreases the load factor.

What needs to happen when the table is resized?

Each element in the hash table needs to be rehashed so as to use the now modified compression function, which is dependent on *N*.

Question 3.

Starting with an initially empty skip list, and by flipping a real coin to obtain, i, show the result of each insertion of the following keys into a skip list: 10, 12, 14, 15, 23, 32, and 35.

Indicate the value of i, and show the resulting skip list after each insertion.