Name: Omid Azodi

Professor: Dr. Hadaegh

Class: CS443

Due Date: 05/05/2016

**CS443-- Lab 5**

**Question 1:**

A PARTS file with Part# as hash key includes records with the following Part# values: 2369, 3760, 4692, 4871, 5659, 1821, 1074, 7115, 1620, 2428, 3943, 4750, 6975, 4981, 9208. The file uses 8 buckets, numbered 0 to 7. Each bucket is one disk block and holds two records. Load these records into the file in the given order using the hash function h(K)=K mod 8. Calculate the average number of block accesses for a random retrieval on Part#.

**Part # after being hashed:**

Hash Function: h(K) = K mod 8

1. 2369 % 8 = 1
2. 3760 % 8 = 0
3. 4692 % 8 = 4
4. 4871 % 8 = 7
5. 5659 % 8 = 3
6. 1821 % 8 = 5
7. 1074 % 8 = 2
8. 7115 % 8 = 3
9. 1620 % 8 = 4
10. 2428 % 8 = 4 (2428 is overflowed based on the diagrams I made below, Max only 2 per block).
11. 3943 % 8 = 7
12. 4750 % 8 = 6
13. 6975 % 8 = 7 (6975 is overflowed based on the diagrams I made below, Max only 2 per block).
14. 4981 % 8 = 5
15. 9208 % 8 = 0

|  |
| --- |
| **Index: 2** |
| 1074 |
|  |

|  |
| --- |
| **Index: 3** |
| 5659 |
| 7115 |

|  |
| --- |
| **Index: 4** |
| 1620 |
| 4692 |

|  |
| --- |
| **Index: 0** |
| 3760 |
| 9208 |

|  |
| --- |
| **Index: 1** |
| 2369 |
|  |

|  |
| --- |
| **Index: 5** |
| 1821 |
| 4981 |

|  |
| --- |
| **Index: 6** |
| 4750 |
|  |

|  |
| --- |
| **Index: 7** |
| 3943 |
| 4871 |

|  |
| --- |
| **Overflow** |
| 2428 |
| 6975 |

**Calculate the average number of block accesses for a random retrieval on Part#.**

* There are 15 part numbers.
* Hashing is O(1) retrieval for accessing an index.
* Out of the 8 buckets from indexes 0 – 7 having 2 records each, we have 8 \* 2 = maximum of 16 blocks.

From my diagram above only 13 records are inside the total of the buckets.

15 – 13 = 2 remaining, the 2 remaining were the overflow and go inside the overflow buffer if we have one.

O(1) means one access time for an index of a random retrieval of a part #:

So we have a 13/15 records to choose from.

**1 \* (13/15) = (13/15) = .87**

**2 \* (2/15) = [Overflow Area] = .27**

**.87 + .27 = 1.14**

**Question 2:**

Consider the following records of question 1:

**2369, 3760, 4692, 4871, 5659, 1821, 1074, 7115, 1620**

Load the records into expandable hash files based on extendible hashing. You can put two records per block. Show the directory at each step, and the global and local depths. Use the hash function h(k) = K mod 32 to find the bucket number. For example,

***Record# K h(K) bucket number binary h(K)***

***Record 1 2369 1 00001***

***Record 2 3760 16 10000***

***Record 3 4692 20 10100***

***Record 4 4871 7 00111***

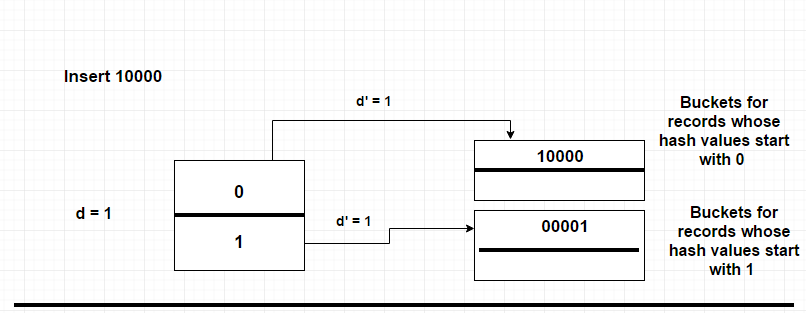
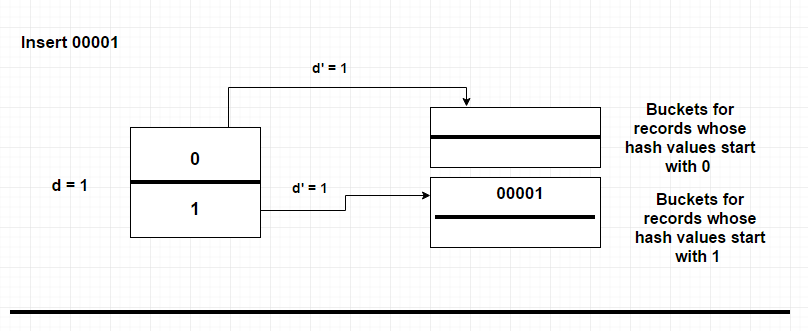
***Record 5 5659 27 11011***

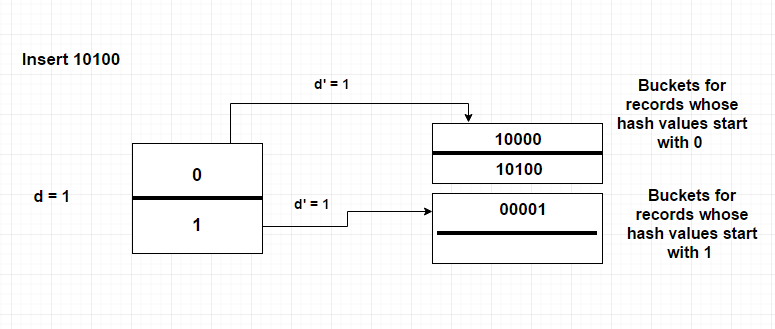
***Record 6 1821 29 11101***

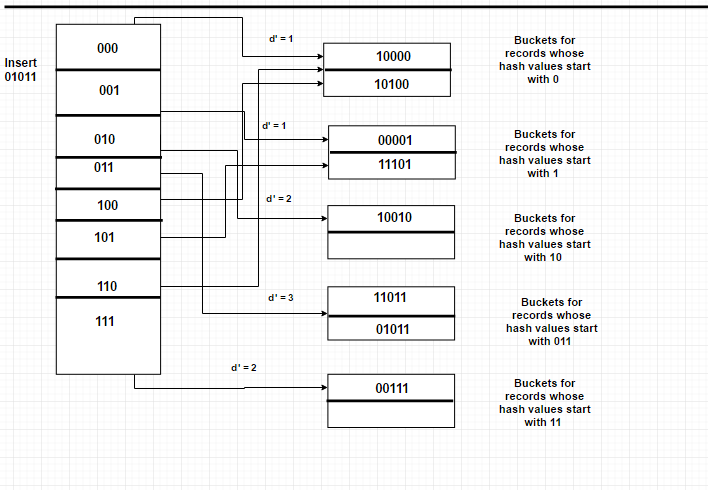
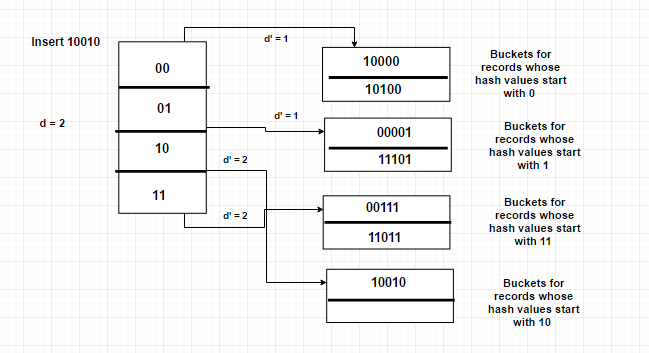
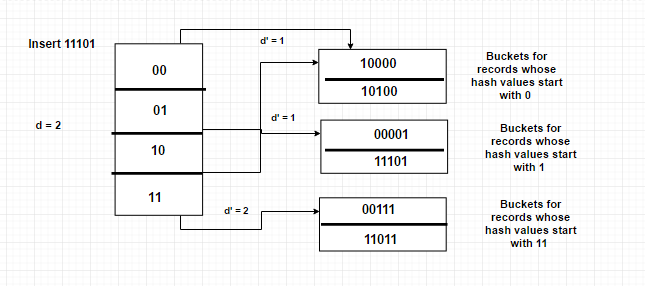
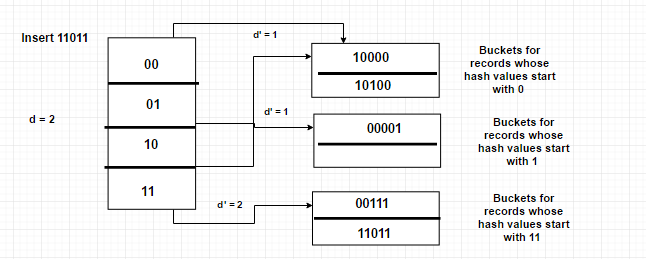
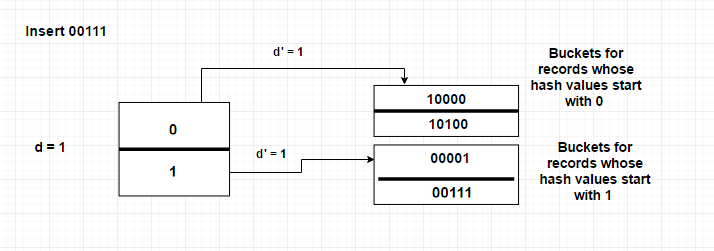
***Record 7 1074 18 10010***

***Record 8 7115 11 01011***

***Record 9 1620 20 10100***



****



**Question 3:**

Insert the following into B-tree of order **4**. Show your work step by step with proper illustration of pointers as shown in pages 20-29 in multi-way trees lecture

**8, 2, 80, 25, 26, 27, 28, 55, 71, 15, 51, 20, 21, 22, 23, 24, 63, 90, 35**

**\*Hard copy submitted \***

**Question 4:**

**Insert the following into B+ tree of order 3. Show your work step by step with proper illustration of pointers as shown in pages 47-54 in multi-way trees lecture**

**90, 22, 27, 24, 28, 20, 51, 63, 8, 80, 15, 71, 35, 55**

**Hard copy submitted \***

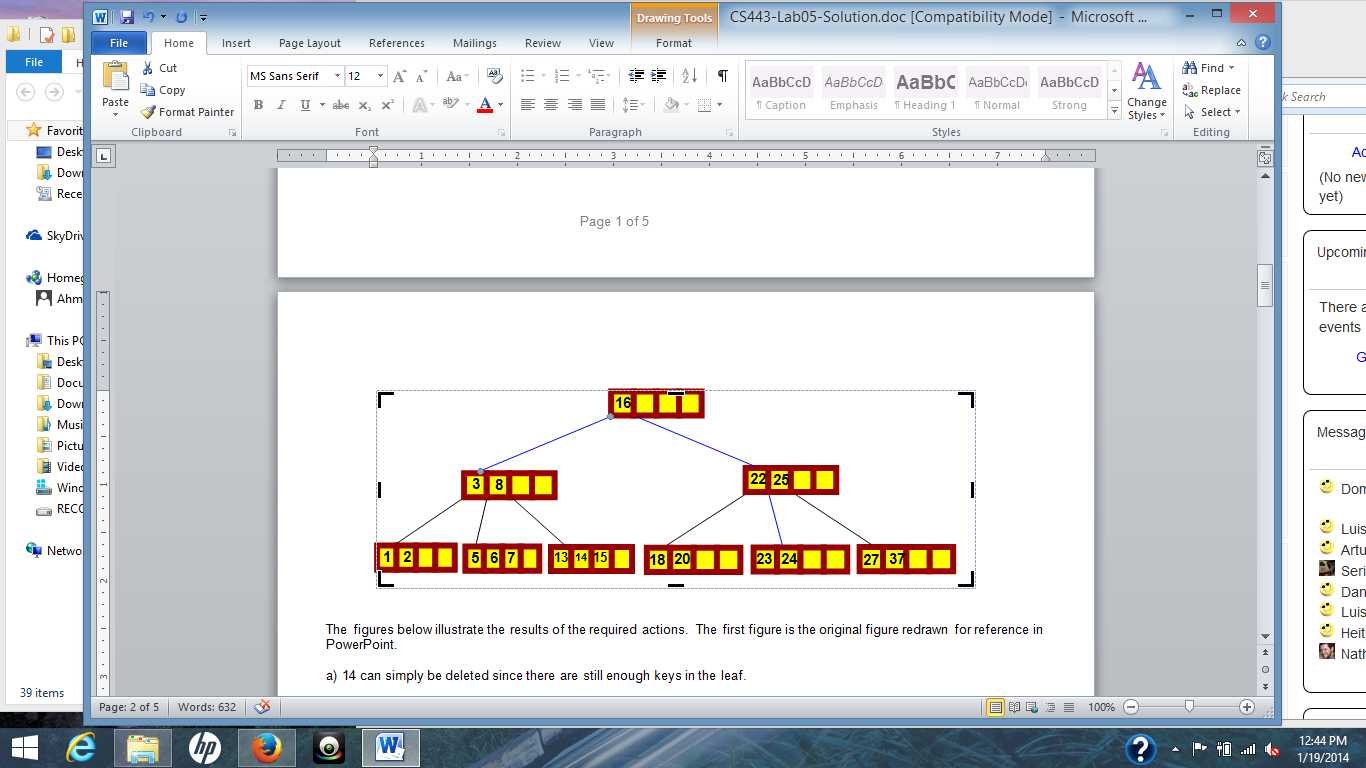
**Question 5:**

Consider the following B-tree.

a) Redraw the tree after deleting 14.

b) Again, redraw the tree after deleting 15.

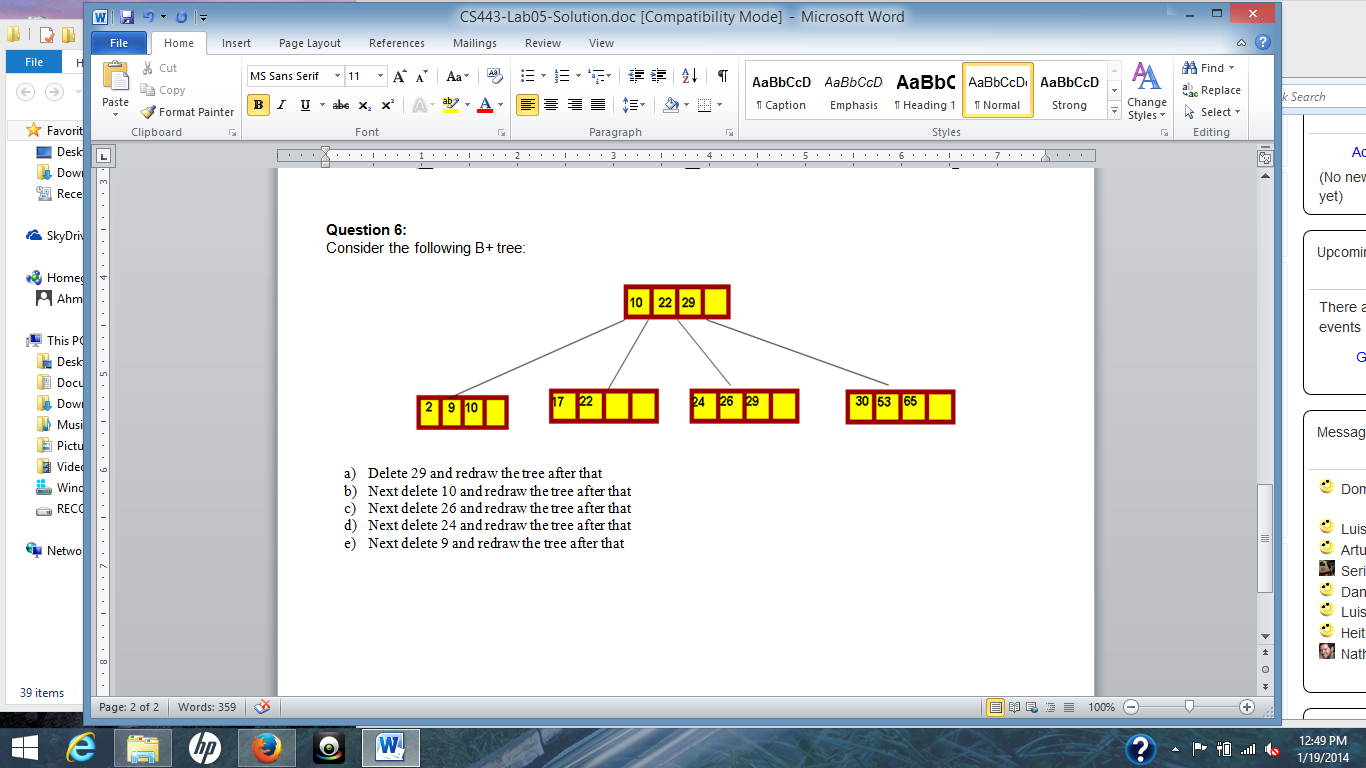
c) Again, redraw the tree after deleting 25.



**Hard copy submitted \***

**Question 6:**

Consider the following B+ tree:



1. Delete 29 and redraw the tree after that
2. Next delete 10 and redraw the tree after that
3. Next delete 26 and redraw the tree after that
4. Next delete 24 and redraw the tree after that
5. Next delete 9 and redraw the tree after that

**Hard copy submitted \***