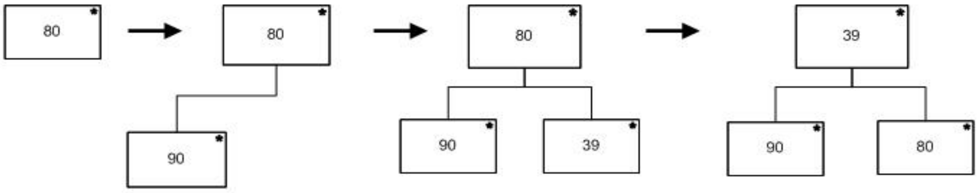
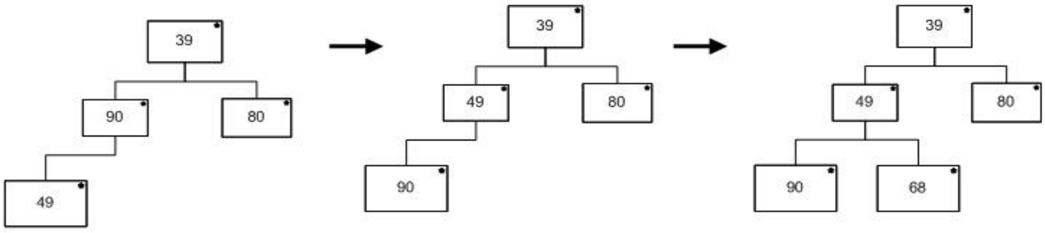
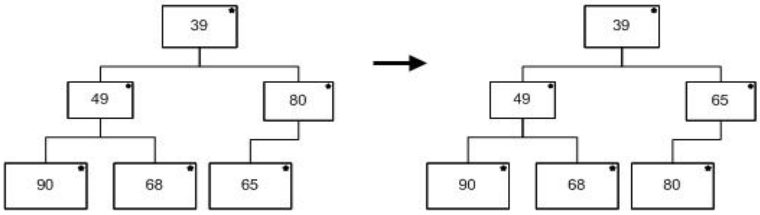
Safa Erişti
Sec03

CS 351 – Homework 3

Answer 1.

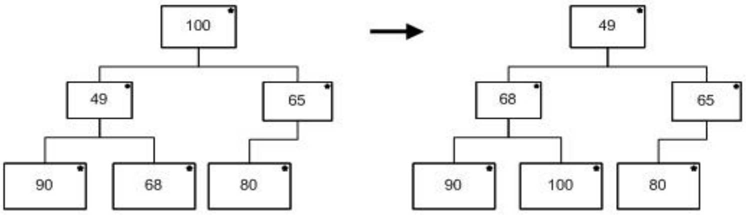






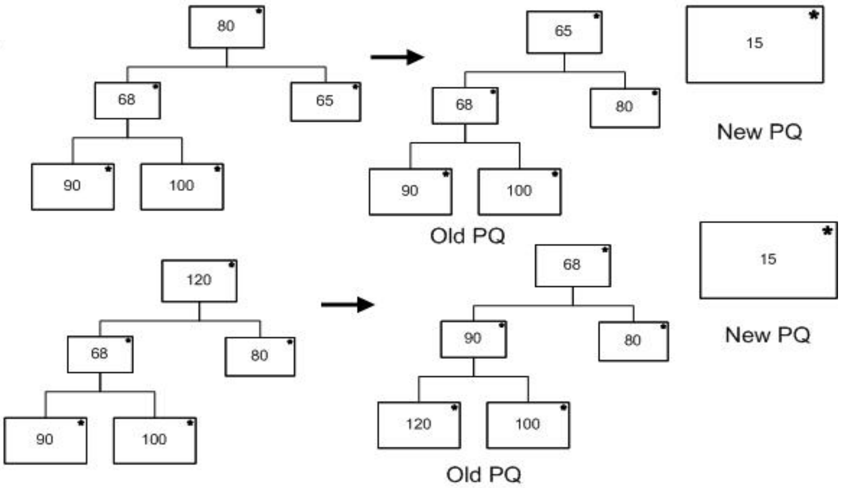
We start to output from now on.

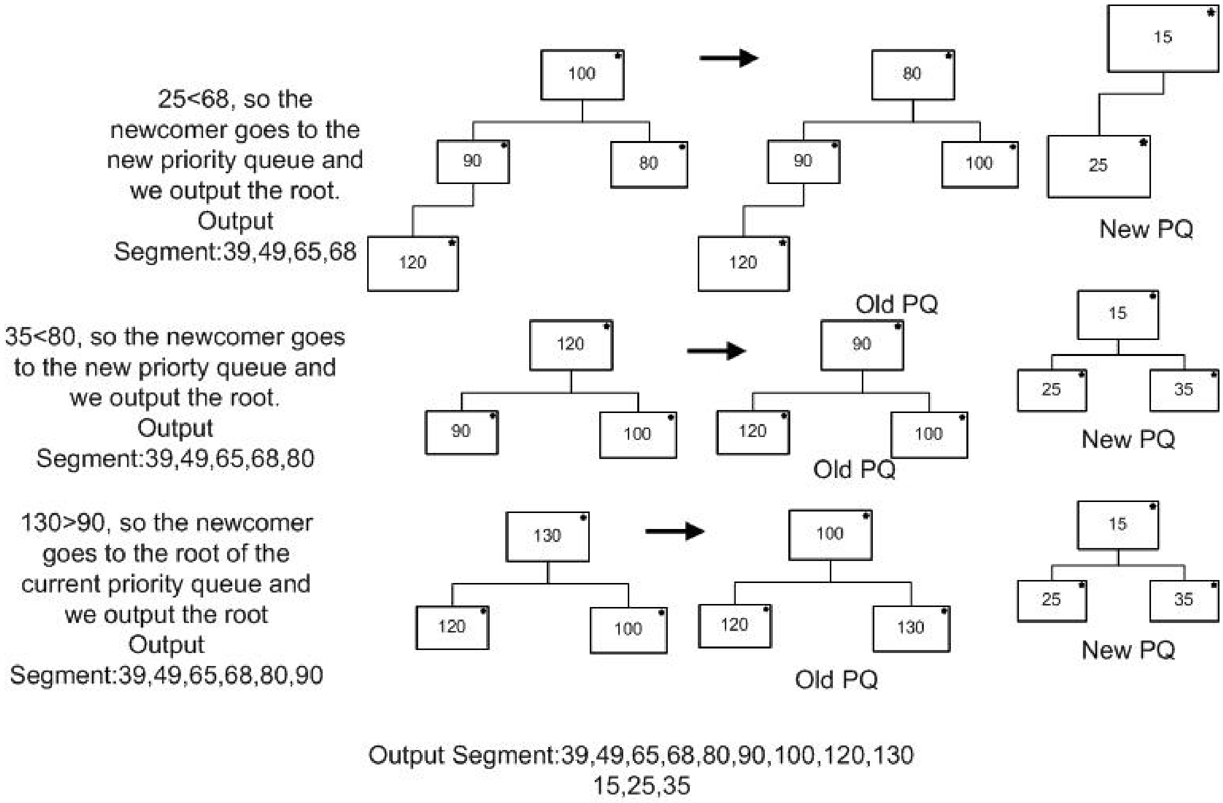
100>39, so the newcomer goes to the root of the current queue and we output the current root
Output Segment :39



15<49, so the newcomer goes to the new priority queue and we output the current root.
Output Segment: 39,49

120>65, so the newcomer goes to the current priority queue and we output the root.
Output Segment:39,49,65





25<68, so the newcomer goes to the new priority queue and we output the root.
Output Segment:39,49,65,68

35<80, so the newcomer goes to the new priorty queue and we output the root.
Output Segment:39,49,65,68,80

130>90, so the newcomer goes to the root of the current priority queue and we output the root
Output Segment:39,49,65,68,80,90

Output Segment:39,49,65,68,80,90,100,120,130
15,25,35

Answer 2.

HeapSort, unlike other sorting algorithms, can be executed overlapping the input and output. Only I/O time needs to be counted. So the answes is:

2 * b * ebt = 2 * 0.84 * [(400 * 10000000)/2400]

2 * 0.84 *1666666.7 = 2800000ms = 2800seconds.

Answer 3.

In the previous question we calculated the heap sort, now we need to calculate the merge times and add them together.

a)For 2-way merge time needed for one pass is = 2 * p * nsg * (r+s) + 2 * b *ebt

= 2 * 2 * 400 * (24.3) + 2 * 1666666.7 * 0.84 = 2838880 ms

We have ceiling[$\log_2$(400)] = 9, we have 9 passes, so

9 * 2838880 = 25549920ms and we need to add time on answer 2.

Total time is =7.09 hours + 0.78hours = 7.87 hours

b) For 4-way merge time needed for one pass is=2 *4 *400 *(24.3)+2 * 1666666.7 * 0.84 = 2877760ms

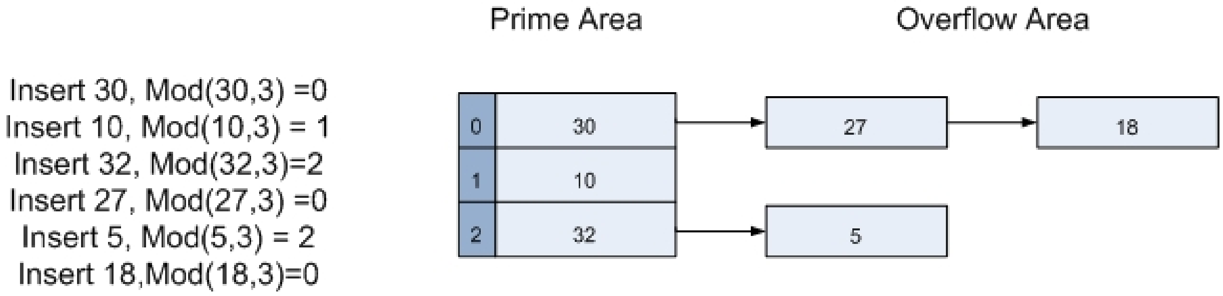
We have ceiling[$\log_4$40] = 5, we have 5 passes, so 5 * 2877760 = 3.99 hours

We need to add the time on answer two, so the total time is 3.99 + 0.78 = 4.77hours

c)For this question P is 400, we can use a specialized formula since only one pass is needed.

4 * ebt * b + 2 * (nsg)^2 * (r+s) = 4 * 0.84 * 1666666.7 + 2 * (400)^2 * 24.3 = 3.71 hours

Answer 4.



a) To find 18, 3 disk accesses is required because first access will be to prime area, second access for first entry in the overflow area and on the third access 18 is retrieved.

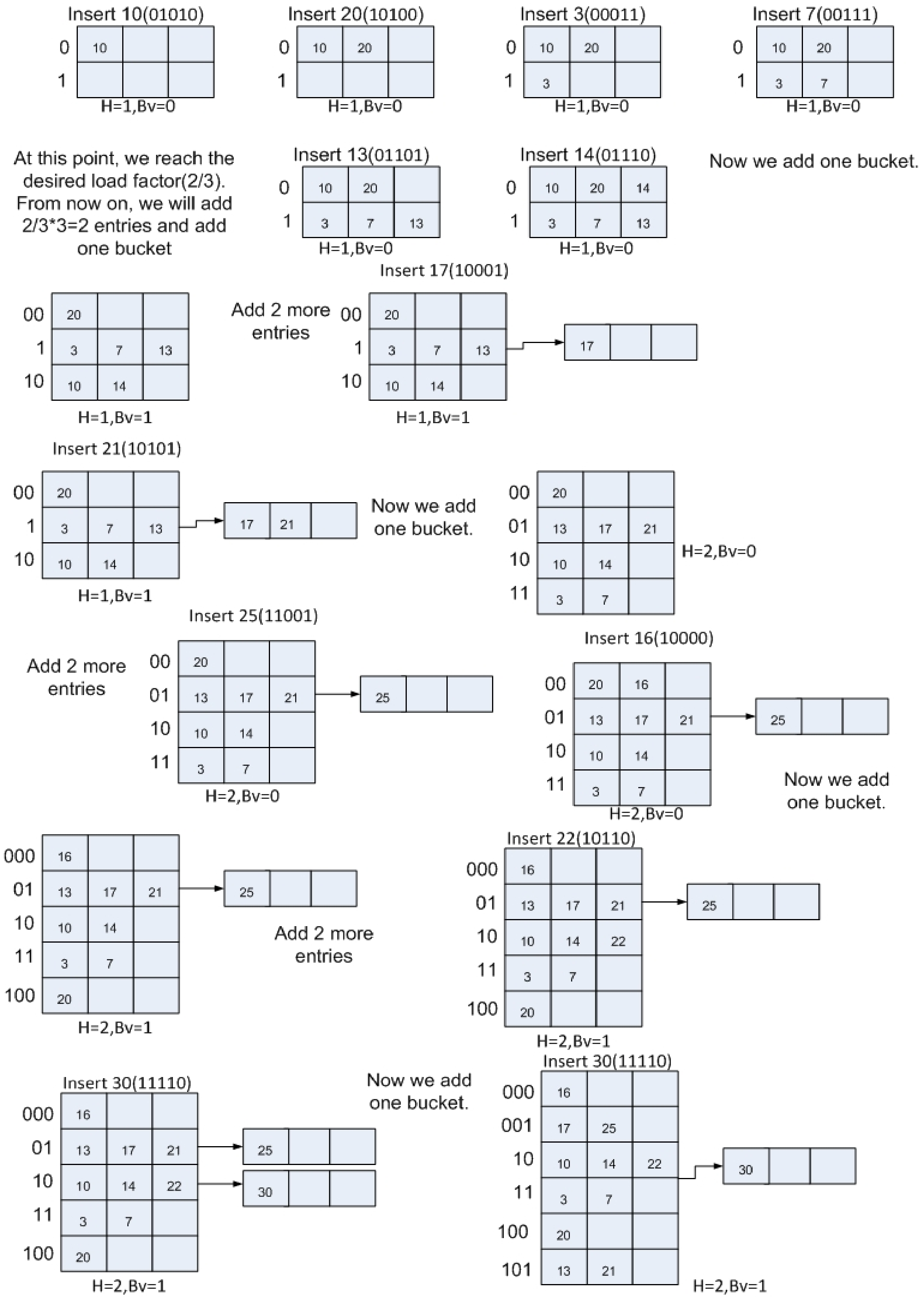
b) For 30, 10 and 32 one disk access is requires, for 27 and 5, two disk access is required and for 18, three disk accesses is required and we have 6 entries. The average number of accesses for successfull search is = 1+1+1+2+2+3/6 = 1.666

c) For entires which have Mod(entry,3) = 0, three disk accesses is required for unsuccessfull search, for entries which have Mod(entry,3)=1, 1 disk access is required and for entries which have Mod(entry,3)=2, 2 disk accesses is required. So (3 + 1 + 2)/3 = 2

Answer 5.

Pseudo Key	Key	Pseudo Key In Binary
10	10	01010
20	20	10100
3	3	00011
7	7	00111
13	13	01101
14	14	01110
17	17	10001
21	21	10101
25	25	11001
16	16	10000
22	22	10110
30	30	11110

We start from h=1, Bv=0, Insertions illustrated below.



At this point, we reach the desired load factor(2/3). From now on, we will add 2/3*3=2 entries and add one bucket

Now we add one bucket.

Add 2 more entries

Now we add one bucket.

Add 2 more entries

Now we add one bucket.

Add 2 more entries

Now we add one bucket.

Answer 6.

For bv=5, h=5;

In linear hashing, blocks hashed at level h+1 are: [0,bv-1] and [2^h,n-1] n being the number of blocks. There are 5 blocks in [0, bv-1] interval. The other part always has the same quantity. So the blocks hashed at level h+1 is 10.

Blocks hashed at level h are in the interval [bv,2^h-1] which is 27.

The binary address of the last bucket of the file is 100100.

The binary address of the last bucket hashed at level 5 is 11111.

For bv=0, h=5;

In linear hashing, blocks hashed at level h+1 are: [0,bv-1] and [2^h,n-1] n being the number of blocks. There are 0 blocks in [0, bv-1] interval. The other part always has the same quantity. So the blocks hashed at level h+1 is 0.

Blocks hashed at level h are in the interval [bv,2^h-1] which is 32.

The binary address of the last bucket of the file is obviously 11111.

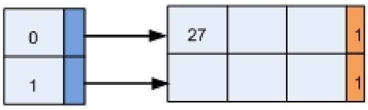
The binary address of the last bucket hashed at level 5 is 11111.

Answer 7.

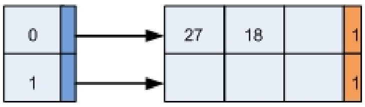
Key Value	**Pseudo Key**	**Binary Pseudo Key**
27	5	0101
18	7	0111
29	7	0111
28	6	0110
42	9	1001
13	2	0010
16	5	0101

Safa Erişti
Sec03

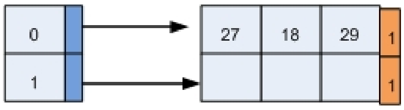
Insert 27 (0101)



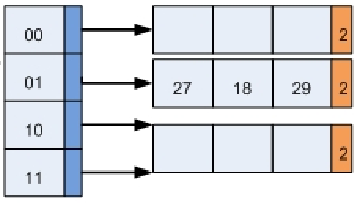
Insert 18 (0111)



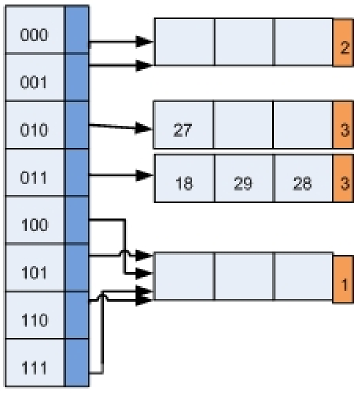
Insert 29 (0111)



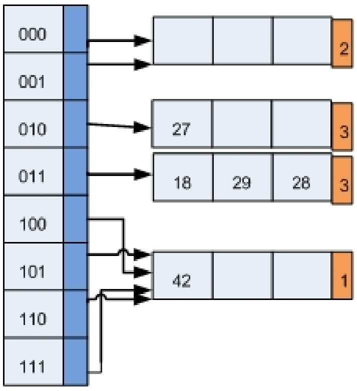
Insert 28 (0110)
Overflow in bucket 0,
table doubles



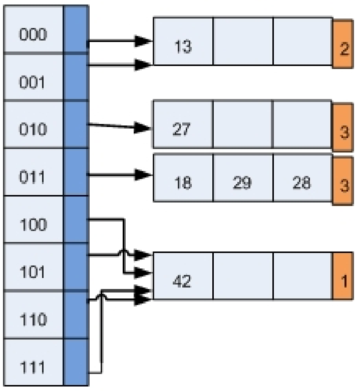
Still can not add
28,Overflow in bucket
01, table doubles



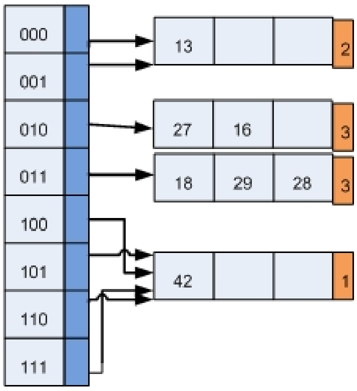
Insert 42(1001)



Insert 13(0010)



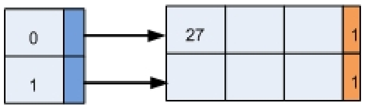
Insert 16(0101)



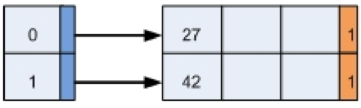
Answer 8.

Key Value	Pseudo Key	Binary Pseudo Key
27	5	0101
42	9	1001
18	7	0111
29	7	0111
28	6	0110
13	2	0010
16	5	0101

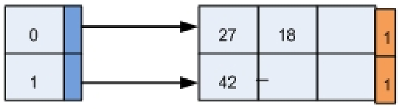
Insert 27 (0101)



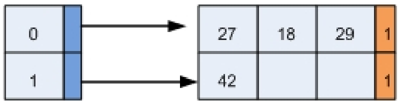
Insert 42 (1001)



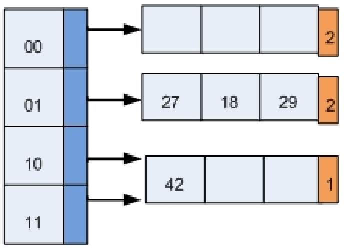
Insert 18 (0111)



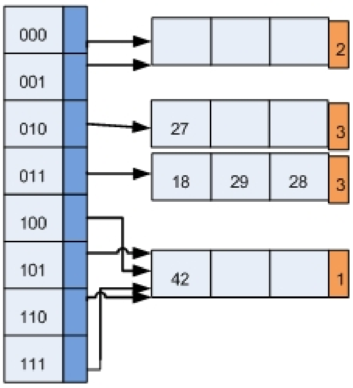
Insert 29 (0111)



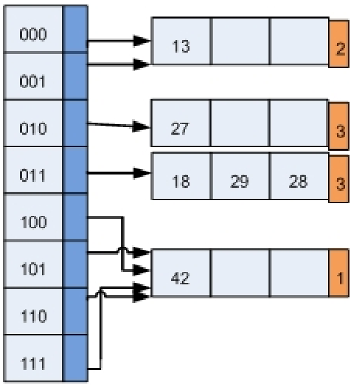
Insert 28(0110), Overflow in bucket 0, table doubles



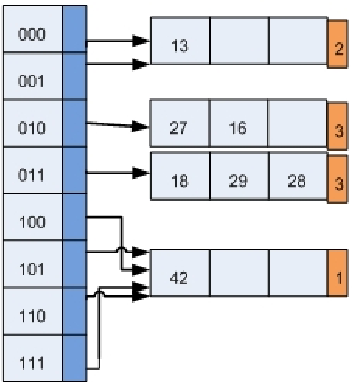
Still can not insert 28(0110), Overflow in bucket 01, table doubles



Insert 13(0010)



Insert 16(0101)



Directory size does not change, it is 8 for questions 7 and 8. Number of data pages,which is 4, is the same also, because no matter how we put the numbers, entries will go to the same place and table will increase accordingly.