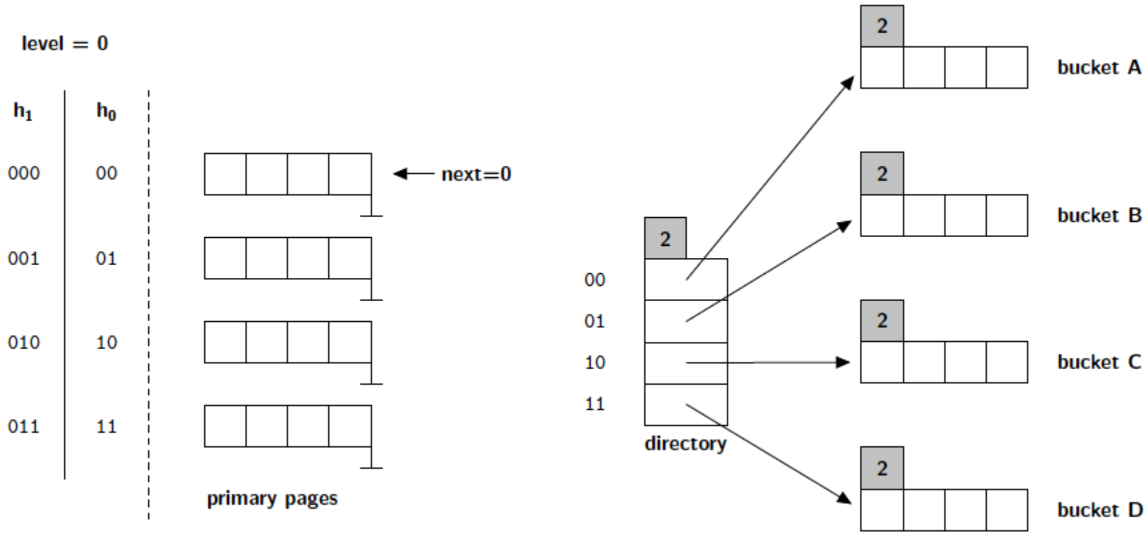
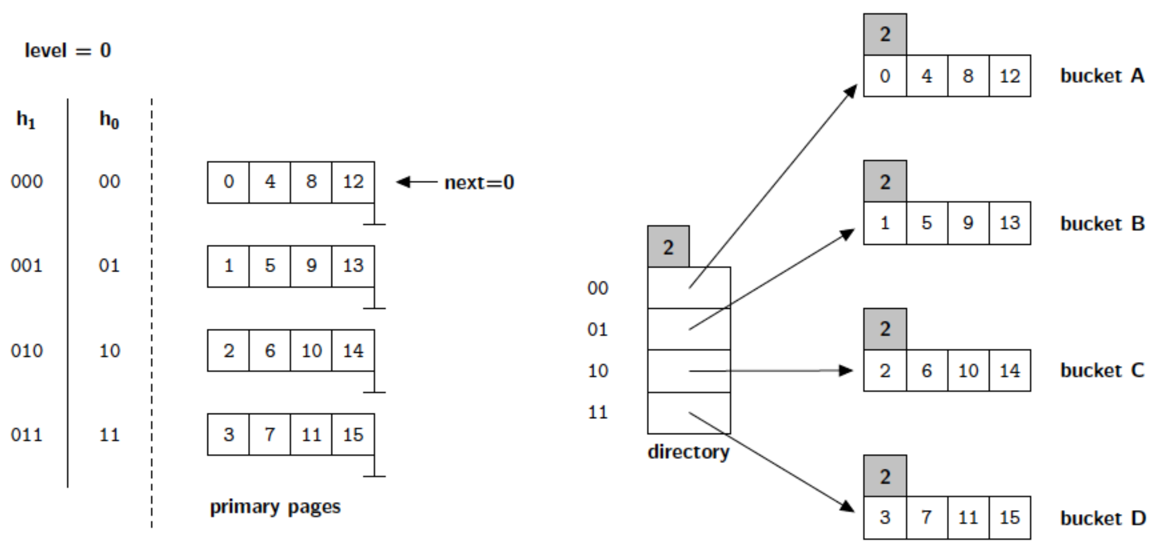
EECS 495
Introduction to Database Systems
Fall 2018
Instructor: Mas-ud Hussain
Solution: Homework Assignment No. 4

Problem No. 1:

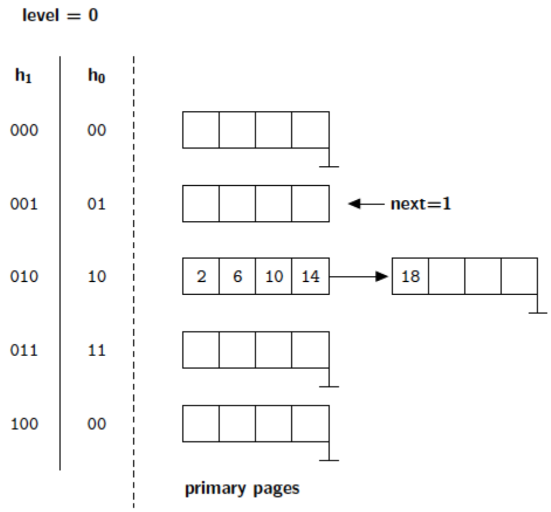
(a) This is a trick question. Even before entering any data, extendible hashing has more pages than linear hashing as extendible hashing needs "bucket address table".

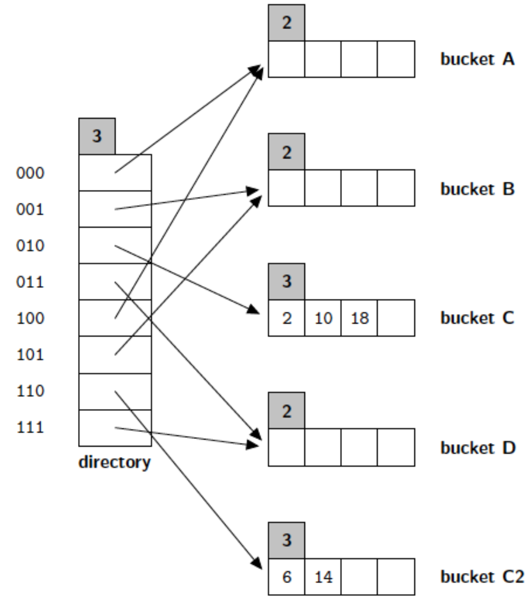


(b)



(c)

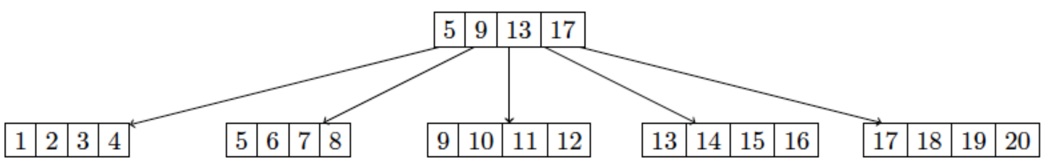




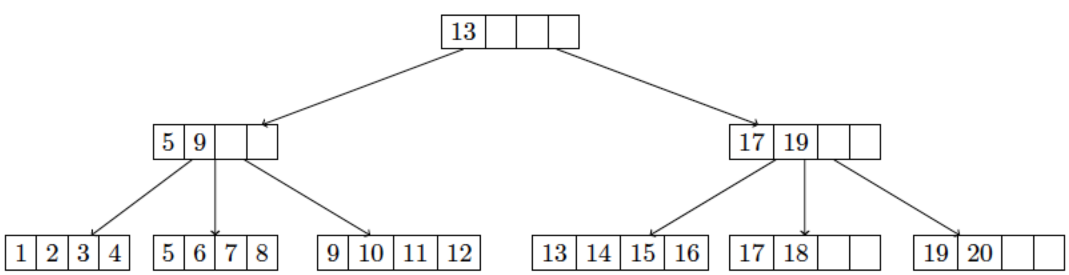
Problem No. 2:

(a)

Insert and delete 21. The tree is full, so the height would have to increase to handle the insertion. Then, when deleting 21, the right-most node would redistribute with its neighbor to maintain the 50 percent occupancy requirement without decreasing the height.

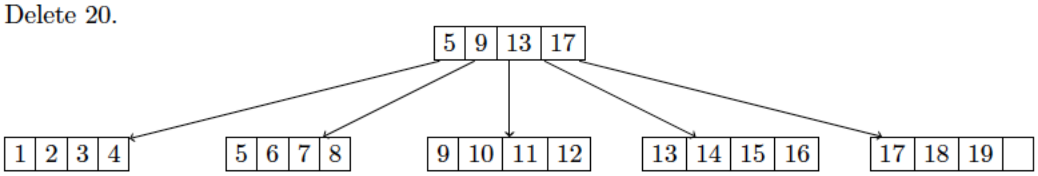


(b)



(c)

Delete 20.



Problem No. 3:

(a)

Sorted Runs of A:

A1 = (8,19,20,20)

A2 = (3,13,18,20)
A3 = (4,5,5,6)

Sorted Runs of B:
B1 = (1,3,13,15)
B2 = (6,10,13,19)
B3 = (2,8,15,16)

(b)
Output: (3,3) (6,6) (8,8) (13,13) (13,13) (19,19) *(output in sorted order)*
Step-1:
$P_{A1} = 8$, $P_{A2} = 3$, $P_{A3} = 4$; Lowest A = 3
$P_{B1} = 1$, $P_{B2} = 6$, $P_{B3} = 2$; Lowest B = 1
As Lowest B < Lowest A; move P_{B1} to next.

Step-2:
$P_{A1} = 8$, $P_{A2} = 3$, $P_{A3} = 4$; Lowest A = 3
$P_{B1} = 3$, $P_{B2} = 6$, $P_{B3} = 2$; Lowest B = 2
As Lowest B < Lowest A; move P_{B2} to next.

Step-3:
$P_{A1} = 8$, $P_{A2} = 3$, $P_{A3} = 4$; Lowest A = 3
$P_{B1} = 3$, $P_{B2} = 6$, $P_{B3} = 8$; Lowest B = 3
As Lowest B = Lowest A; **output (3,3).** Move both P_{A2} and P_{B1}.

Step-4:
$P_{A1} = 8$, $P_{A2} = 13$, $P_{A3} = 4$; Lowest A = 4
$P_{B1} = 13$, $P_{B2} = 6$, $P_{B3} = 8$; Lowest B = 6
As Lowest A < Lowest B; move P_{A3} to next.

Step-5:
$P_{A1} = 8$, $P_{A2} = 13$, $P_{A3} = 5$; Lowest A = 5
$P_{B1} = 13$, $P_{B2} = 6$, $P_{B3} = 8$; Lowest B = 6
As Lowest A < Lowest B; move P_{A3} to next.

Step-6:
$P_{A1} = 8$, $P_{A2} = 13$, $P_{A3} = 5$; Lowest A = 5
$P_{B1} = 13$, $P_{B2} = 6$, $P_{B3} = 8$; Lowest B = 6
As Lowest A < Lowest B; move P_{A3} to next.

Step-7:
$P_{A1} = 8$, $P_{A2} = 13$, $P_{A3} = 5$; Lowest A = 6
$P_{B1} = 13$, $P_{B2} = 6$, $P_{B3} = 8$; Lowest B = 6
As Lowest B = Lowest A; **output (6,6).** Move both P_{B2} and P_{A3} to next. Run A3 is finished processing.

Step-8:
$P_{A1} = 8$, $P_{A2} = 13$; Lowest A = 8
$P_{B1} = 13$, $P_{B2} = 10$, $P_{B3} = 8$; Lowest B = 8
As Lowest B = Lowest A; **output (8,8).** Move both P_{B3} and P_{A1}.

Step-9:
$P_{A1} = 19$, $P_{A2} = 13$; Lowest A = 13
$P_{B1} = 13$, $P_{B2} = 10$, $P_{B3} = 15$; Lowest B = 10
As Lowest B < Lowest A; move P_{B2} to next.

Step-10:
$P_{A1} = 19$, $P_{A2} = 13$; Lowest A = 13
$P_{B1} = 13$, $P_{B2} = 13$, $P_{B3} = 15$; Lowest B = 13
As Lowest B = Lowest A for both (P_{A2}, P_{B2}) and (P_{A2}, P_{B1}) pairs; **output (13,13) and (13, 13).** Move all of P_{B1}, P_{B2} and P_{A2} to next.

Step-11:
$P_{A1} = 19$, $P_{A2} = 18$; Lowest A = 18
$P_{B1} = 15$, $P_{B2} = 19$, $P_{B3} = 15$; Lowest B = 15
As Lowest B < Lowest A**;** move both P_{B1} and P_{B3} to next. Run B1 is finished processing.

Step-12:
$P_{A1} = 19$, $P_{A2} = 18$; Lowest A = 18
$P_{B2} = 19$, $P_{B3} = 16$; Lowest B = 16
As Lowest B < Lowest A**;** move P_{B3} to next. Run B3 is finished processing.

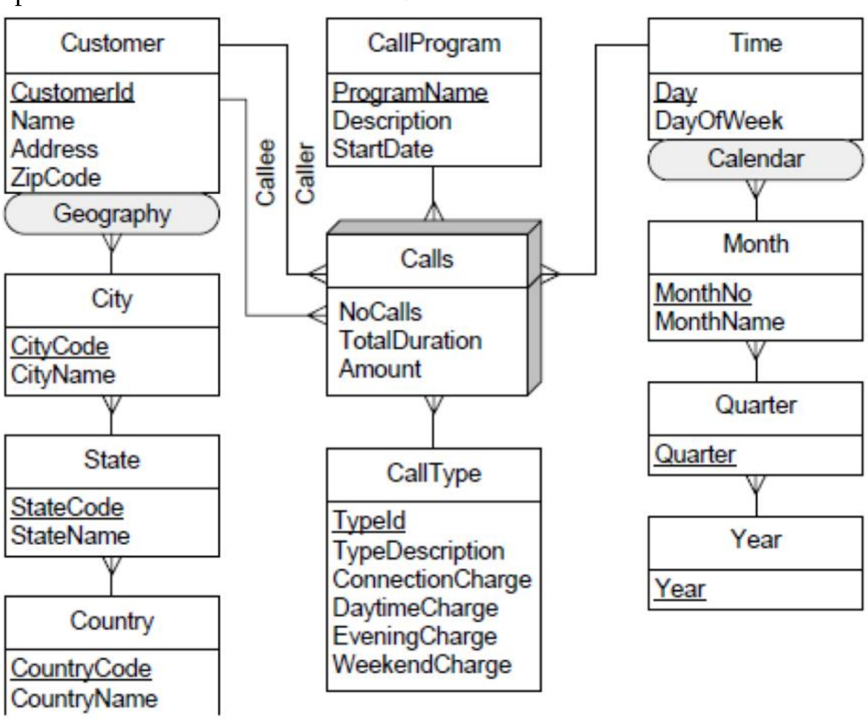
Step-13:
$P_{A1} = 19$, $P_{A2} = 18$; Lowest A = 18
$P_{B2} = 19$; Lowest B = 19
As Lowest A < Lowest B**;** move P_{A2} to next.

Step-14:
$P_{A1} = 19$, $P_{A2} = 20$; Lowest A = 19
$P_{B2} = 19$; Lowest B = 19
As Lowest B = Lowest A; **output (19,19).** Move both P_{B2} and P_{A1}. Run B2 is finished processing.

As all runs of B is finished processing, there cannot be any more matches. So, the algorithm also terminates.

Problem No. 4:

(a)
A possible solution is shown below:



(b)

1.

Calls1 ← _DICE(Calls, Caller.City = 'Brussels')
Calls2 ← _DICE(Calls1, Time.Year = 2016)
Result ← _ROLLUP*(Calls3, SUM(TotalDuration))

2.

Calls1 ← _DICE(Calls, Time.Year = 2016)
Calls2 ← _ROLLUP(Calls1, Time →Year, SUM(Amount))
Calls3 ← _ROLLUP(Calls2, Caller → _All, SUM(Amount))
Calls4 ← _ROLLUP(Calls3, Callee → _All, SUM(Amount))
Result ← _ROLLUP(Calls4, CallType → _All, SUM(Amount))

3.

Calls1 DICE(Calls, Time.DayOfWeek = 'Saturday' OR
Time.DayOfWeek = 'Sunday')
Calls2 ← _DICE(Calls1, Caller.City = 'Brussels')
Calls3 ← _DICE(Calls2, Callee.City = 'Antwerp')
Calls4← _DICE(Calls3, Time.Year = 2012)
Result← _ROLLUP*(Calls4, SUM(NoCalls))

4.

Calls1 ← _DICE(Calls, Caller.Country = 'Belgium')
Calls2 ← _DICE(Calls1, Callee.Country <> 'Belgium')
Calls3 ← _DICE(Calls2, Time.Year = 2012)
Result ← _ROLLUP*(Calls3, SUM(TotalDuration))