1) It is preferable to use dense index instead of sparse index when file is not sorted on indexed field (such as when index is a secondary index) or when index file is small compared to size of memory.

2) Primary index – Defined on ordered data file
Data file is ordered on key field.
Key field - primary key of relation.

Secondary index – generated from a field which is candidate key, has unique value in every record.

$\text{ceil}\left(\frac{3}{2}\right) = 2$ each node except root max M child.

3. B tree - 2, 3, 5, 7, 11, 17, 19, 23, 29, 31 n = 3 (max keys)

Right - biased used

insert

max keys = M − 1 = 3

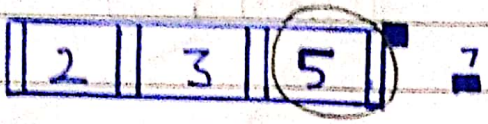
M = 4

max child = 4

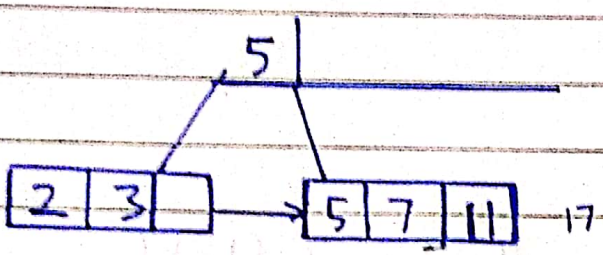
Min child = 2

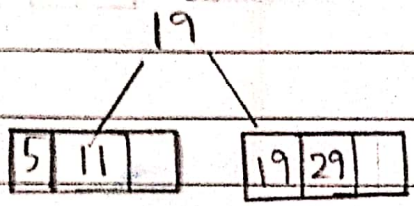
Min keys = $\text{ceil}\left(\frac{4}{2}\right) - 1$

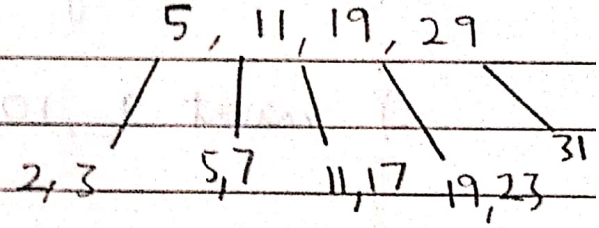
= (1)

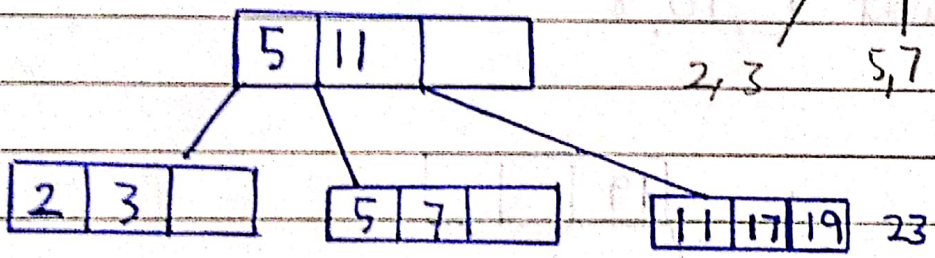


insert 7

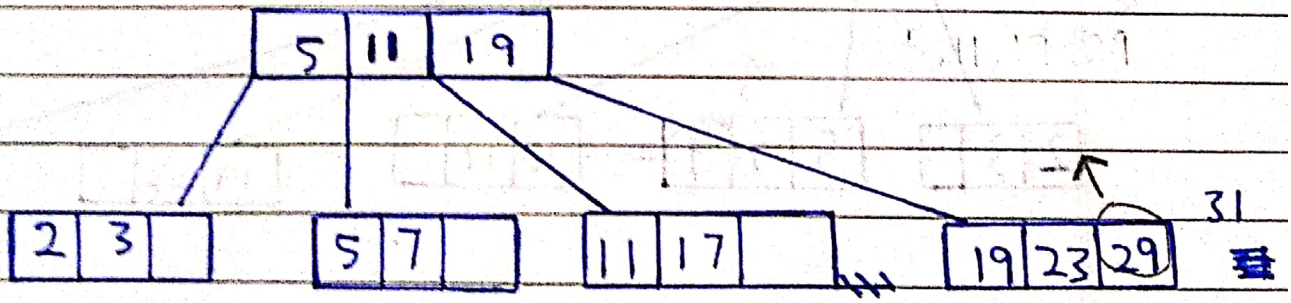






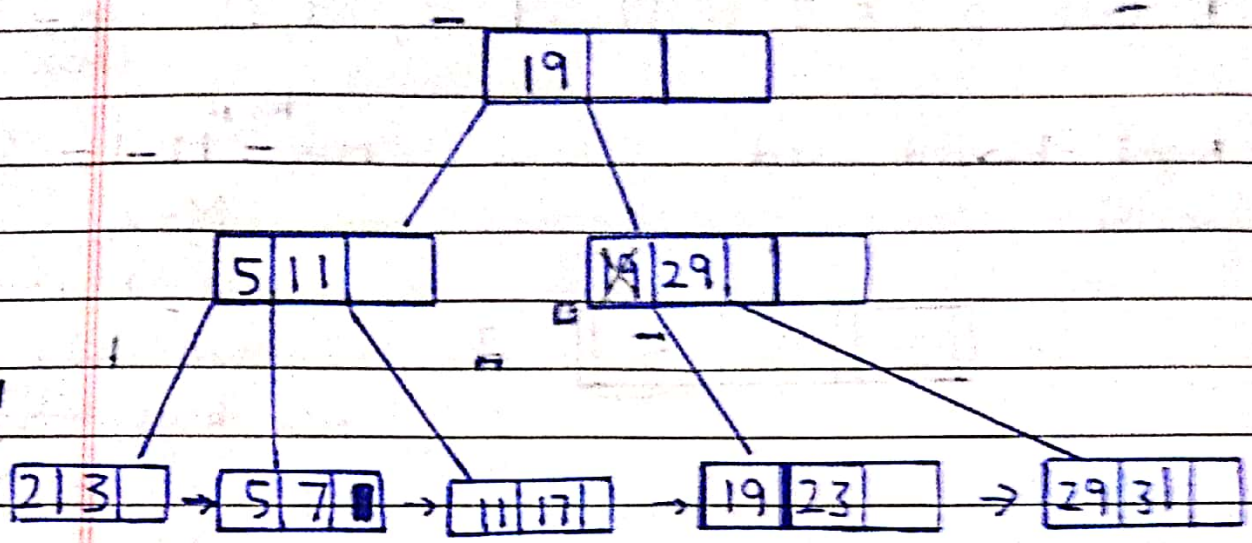


insert 23



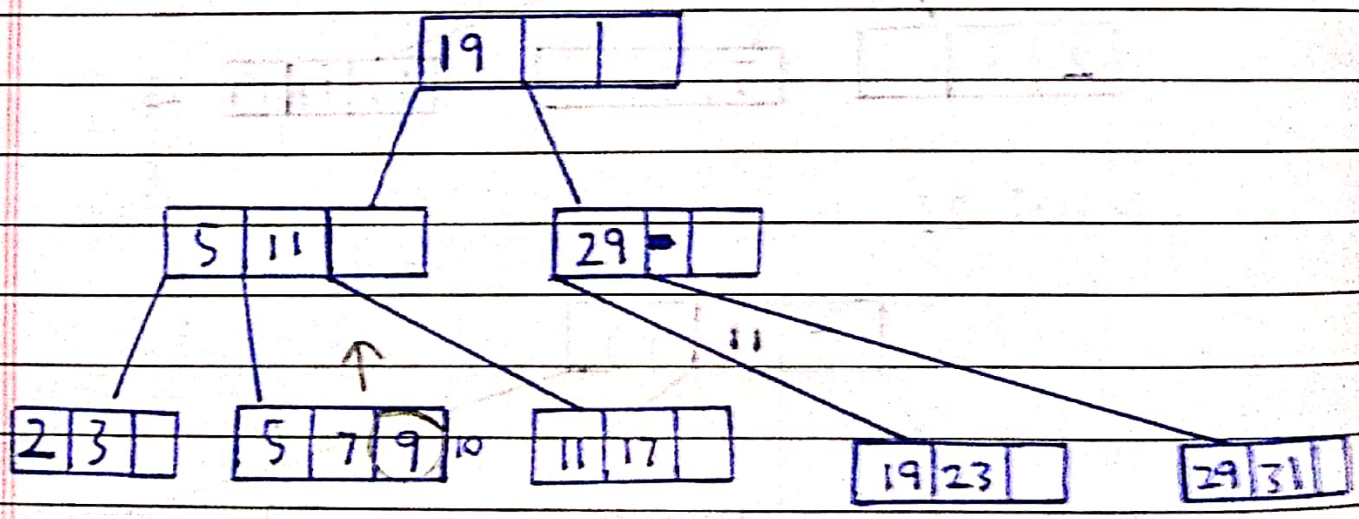
insert 31

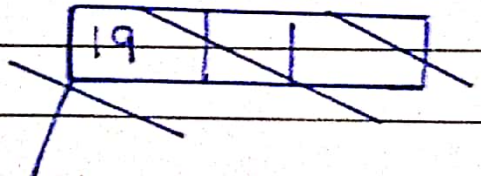
Data don't repeat in internal nodes

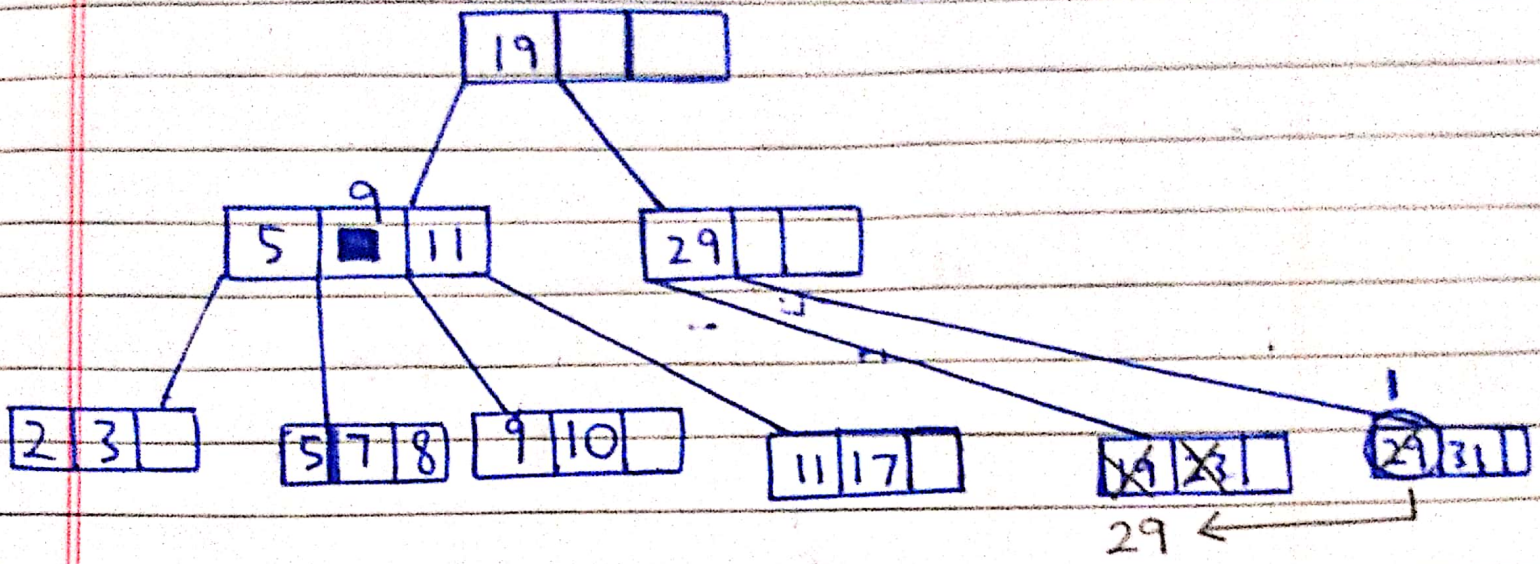


b) search key betwe. 7, 17 — 7, 11, 17

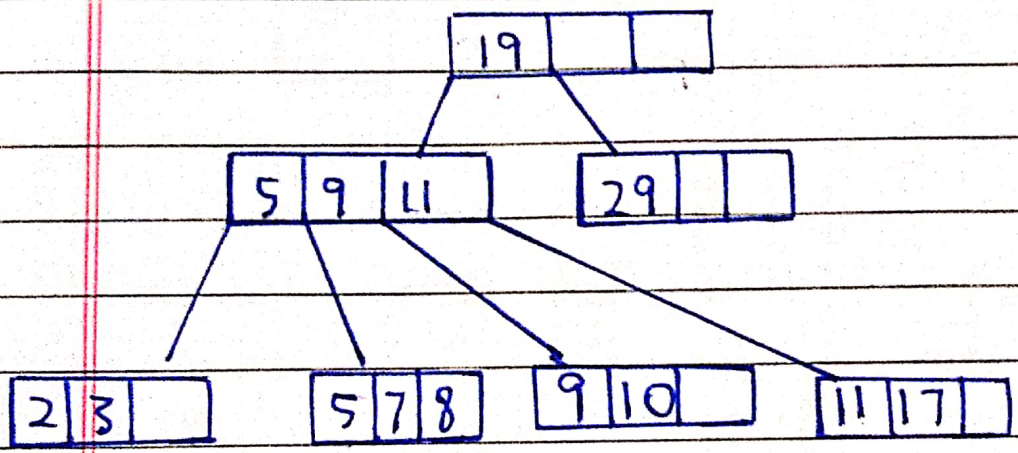
c) insert 9, 10, 8







delete 23, 19



Date
Page

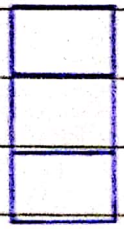
8 | 2 | 0
-0
2

4) extendable hashing

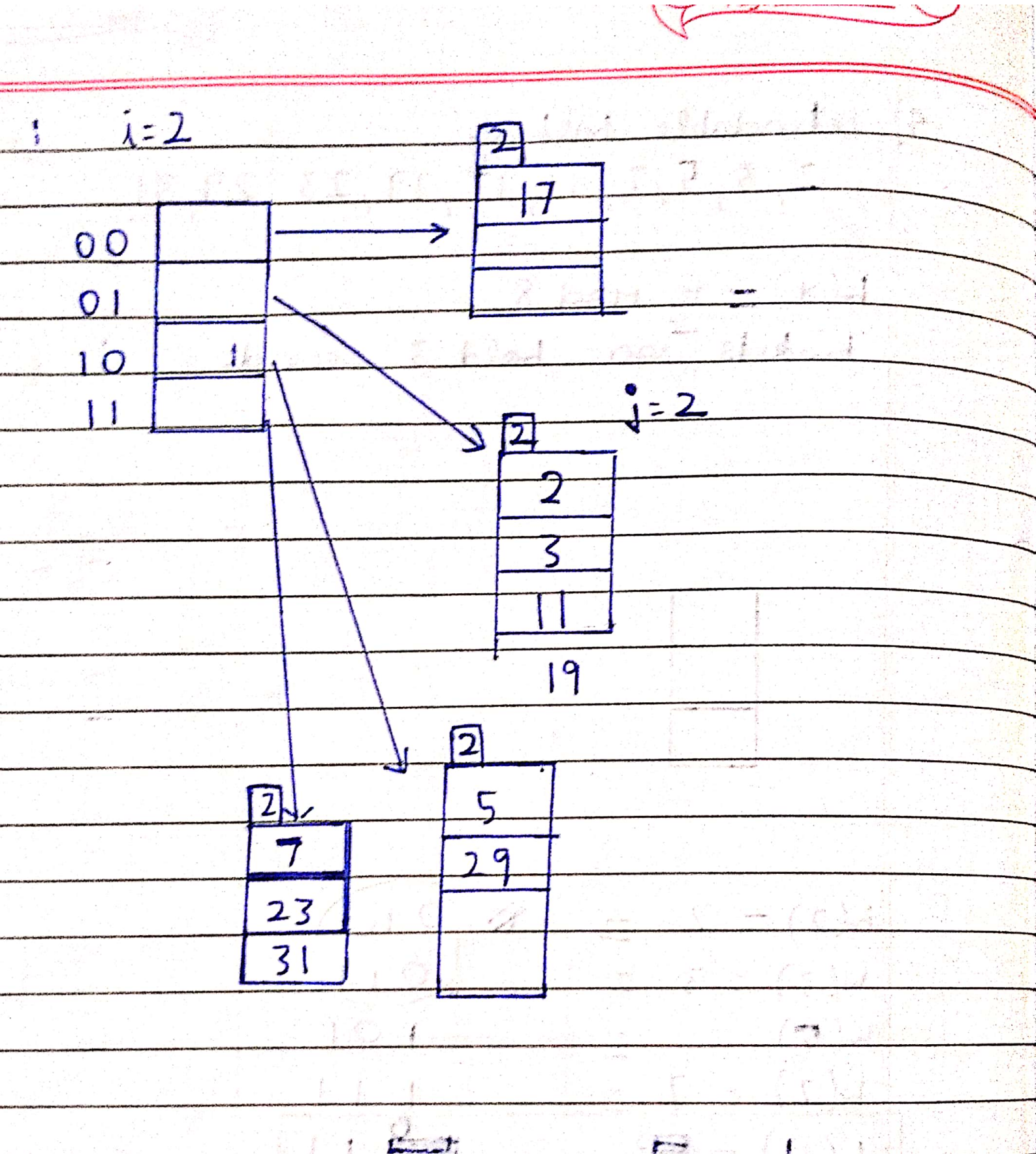
2, 3, 5, 7, 11, 17, 19, 23, 29, 31

$h(x) = x \bmod 8$

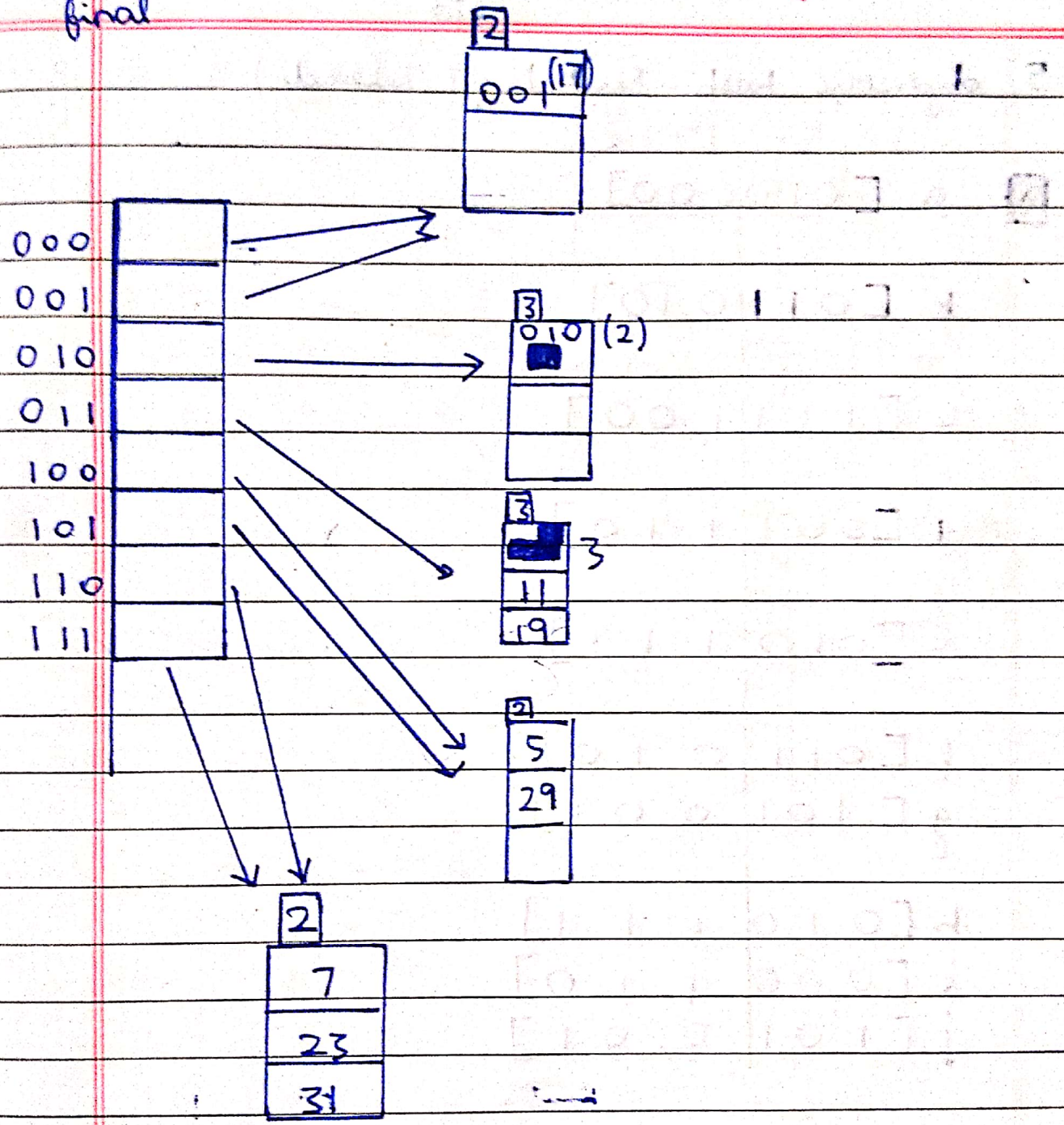
buckets can hold 3 records



$h(2) = 2$	=	0 1 0
$h(3) = 3$	=	0 1 1
$h(5) = 5$	=	1 0 1
$h(7) = 7$	=	1 1 1
$h(11) = 3$	=	0 1 1
$h(17) = 1$	=	0 0 1
$h(19) = 3$		0 1 1
$h(23) = 7$		1 1 1
$h(29) = 5$		1 0 1
$h(31) = 7$		1 1 1



final



upto

5) dynamic hash - buckets (3 records)

a [0 1 0 0 0 0]

b [0 1 1 0 1 0]

c [1 1 1 1 0 0]

d [0 0 1 1 1 0]

e [0 1 0 1 1 1]

f [0 1 1 0 1 0]

g [1 0 1 0 0 1]

h [0 1 0 1 1 1]

i [0 0 0 1 1 0]

j [1 0 1 0 0 1]