Assignment 8

Ans.10

(a) $S_1 = R_1(x)\, R_2(y)\, R_1(z)\, R_2(x)\, R_1(y)$

For 2 operations to be conflicting operations, one of them has to be a write operation. In this part, there are no write operations at all & hence S_1 schedule is conflict serializable

Precedence Graph

T_1 T_2

There are no arcs in its Precedence graph at all

$S_1' = R_1(x)\, R_1(z)\, R_2(y)\, R_2(x)\, R_1(y)$

$S_1' = R_1(x)\, R_1(z)\, R_1(y)\, R_2(y)\, R_2(x)$

Conflict equivalent schedule of S_1. We swapped $R_1(z)$ & $R_2(y)$ since they are 2 read operat-ions & also on 2 different data objects.

(b) $S_2 = R_1(x)\, W_2(y)\, R_1(z)\, R_3(z)\, W_2(x)\, R_1(y)$

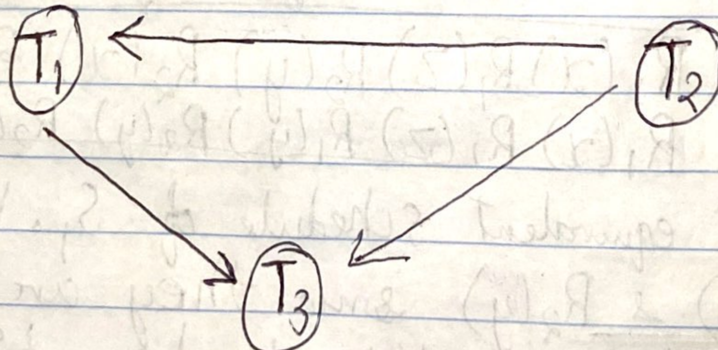
Precedence Graph (S_2)

$T_1 \rightleftarrows T_2$

T_3

S_2 is not Conflict Serializable since its precedence graph is cyclic from T_1 to T_2

(c) $S_3 = R_1(z)\, W_2(x)\, R_2(z)\, R_2(y)\, W_1(x)\, W_3(z)\, W_1(y)\, R_3(x)$

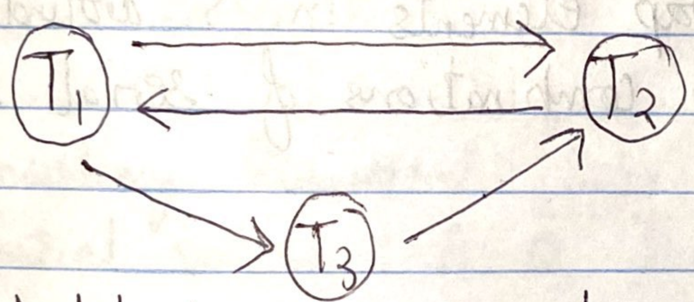


The Precedence Graph for S_3 is acyclic & hence S_3 is conflict serializable

$S_3' = W_2(x)\, R_2(z)\, R_2(y)\, R_1(z)\, W_1(x)\, W_1(y)$
$W_3(z),\ R_3(x)$

S_3' is conflict equivalent with S_3 & it is also serial. We achieve this by # swaps on each of the Transactions operations

Ans-11



Schedule for this precedence graph:

$$\boxed{S = R_1(A)\ W_2(A)\ W_1(B)\ R_3(B)\ W_1(A)\ W_2(B)}$$

T_1: read(A);
read(B);
write(A);

T_2: write(A);
write(B);

T_3: write(B);

Ans.12. T_1 : Read (A), Write (A)
T_2 : Read(B), Write (B)
T_3 : Read (C), Write(C)

Now $S' = R_1(A)\ R_2(B)\ W_1(A)\ W_2(B)\ R_3(C)\ W_3(C)$

S' is a schedule that is conflict equivalent with all serial schedules of T_1, T_2, T_3
We can swap elements in S' around to form all possible combinations of serial schedule.

Ans. 13. T_1: read(A);
read(B);
if A = 0 then B := B+1;
write (B);

T_2: read(B);
read (A);
if B = 0 then A := A+1;
write (A);

Consistency Requirement: $\boxed{A = 0 \lor B = 0}$
Initial values: A = 0, B = 0

(a) 2 possible executions: T_1T_2 or T_2T_1
For T_1T_2, After T_1: A = 0, B = 1
After T_2: A = 0, B = 1
$\therefore$ Consistency is met after T_1T_2

For T_2T_1, After T_2: A = 1, B = 0
After T_1: A = 1, B = 0
$\therefore$ Consistency is met after T_2T_1 as well

(b) When we put some of the actions in betwe one another for T_1 & T_2, it will produce a non serializable schedule. For eg:

T_1	T_2
read (A)	
	read(B)
	read(A)
read (B)	
if A=0 then B:= B+1	
	if B=0, then A:=A+1
	write(A)
write(B)	

(c) There is no non serial schedule of T_1 & T_2 which produces a serializable schedule. For a schedule to be serializable, $A = 0 \vee B = 0$. But if we run T_1 & T_2 in a non serial way, say read A from T_1, then complete T_2 scheduling, we will get A=1 & then complete scheduling of T_1, we get B=1. Thus it doesn't satisfy the conditions of serializable schedule.

Scanned by CamScanner

Part 2

Ans. 6. block size = 4096 bytes
block address size = 9 bytes
" access time = 10 ms
record size = 200 bytes
" key size = 12 bytes

Max Branching factor :

$$n \le \frac{\text{blocksize}}{\underset{\text{size}}{\text{blockaddress}} + \underset{\text{size}}{\text{key}}} \le \frac{4096}{9+12} \le 195.047$$

The minimum time (in ms):

$= \log_n N + 1$

$= \log_{195}(10^{14}) + 1$

$= 7$ $\therefore$ Min^m time : 70ms

(b) For Maxtime

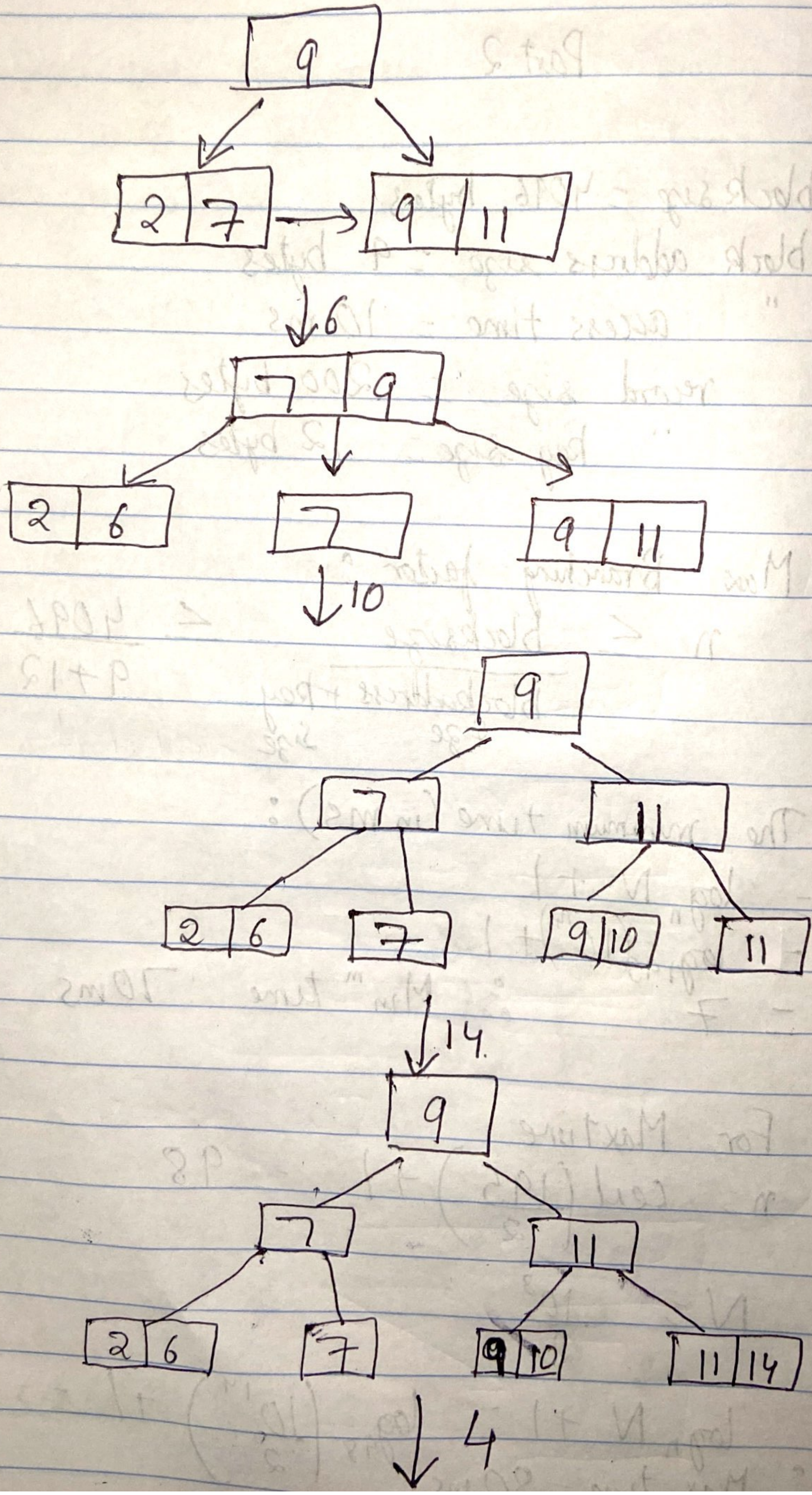
$n = \text{ceil}\left(\frac{195}{2}\right) + 1 = 98$

$N = \frac{10^8}{2}$

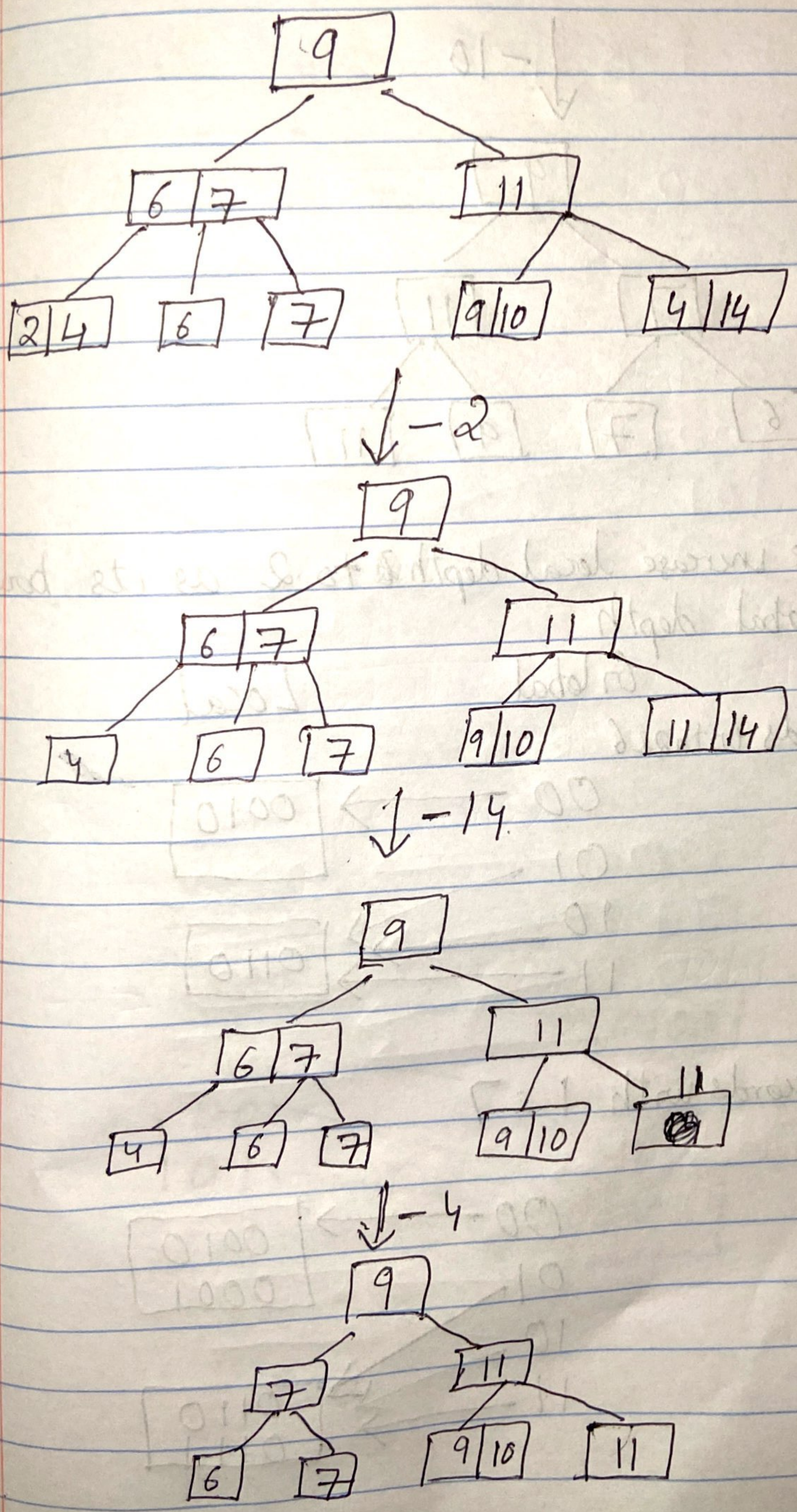
$\log_n N + 1 = \log_{98}\left(\frac{10^{14}}{2}\right) + 1 = 8$

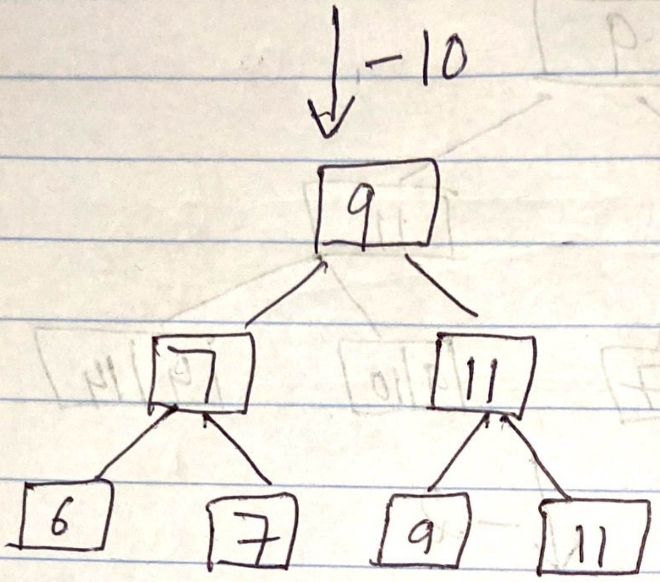
$\therefore$ Max time = 80 ms

Ans. 7 (a)



(b)

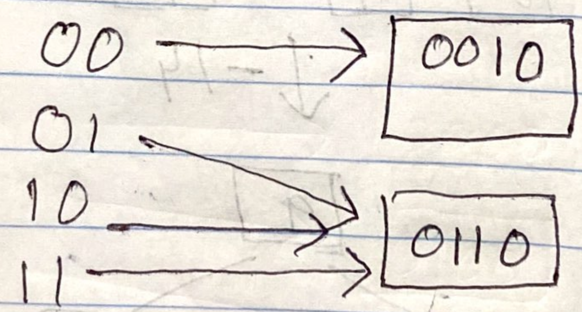




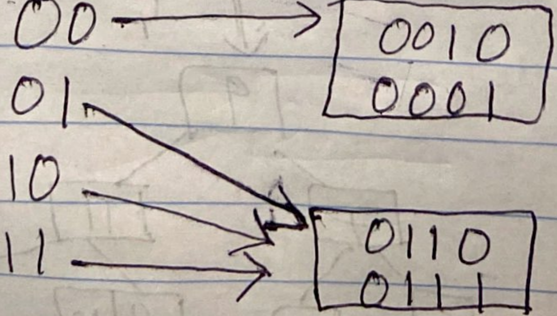
Ans. 8a) We increase local depth to 2 as its bounded by global depth

(i) Global Local

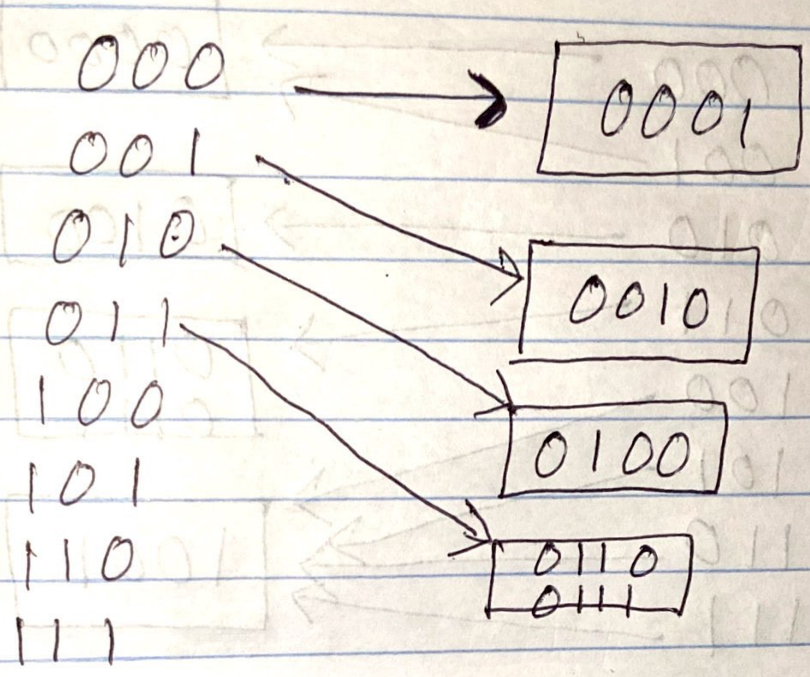
records with 2&6



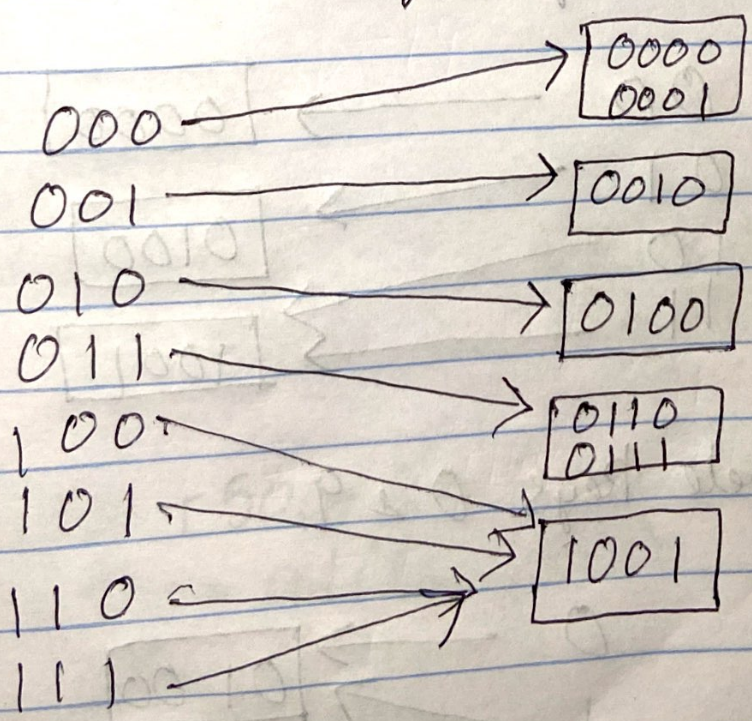
(ii) records with 1 & 7



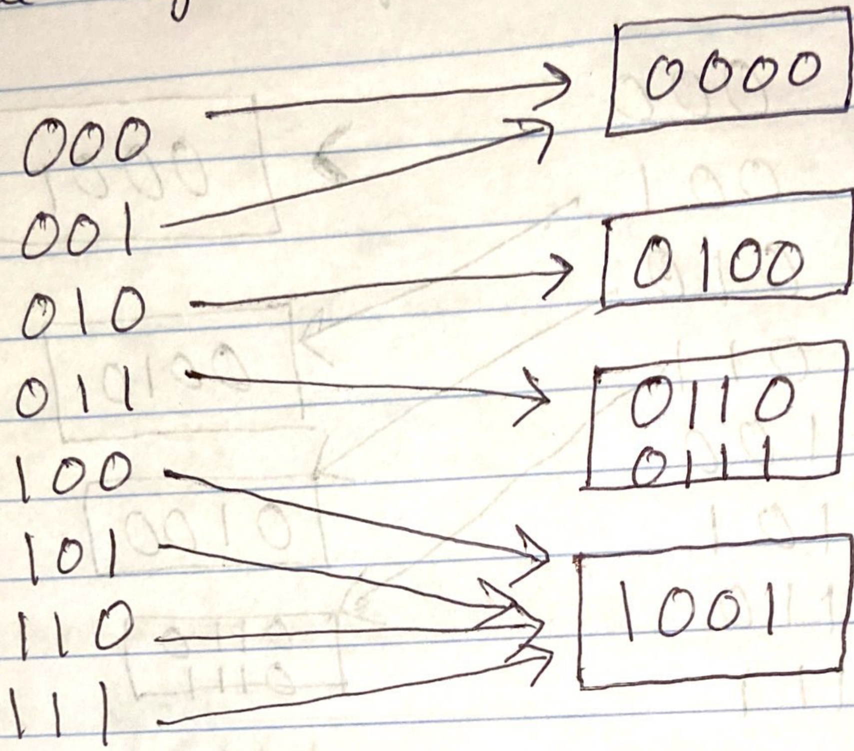
(iii) records with keys 4 & 8



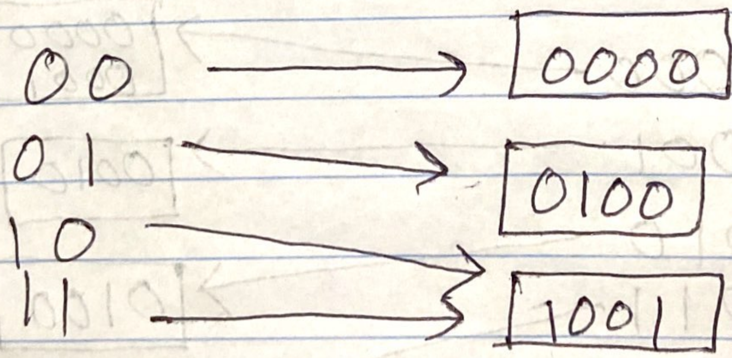
(iv) records with keys 0 & 9.



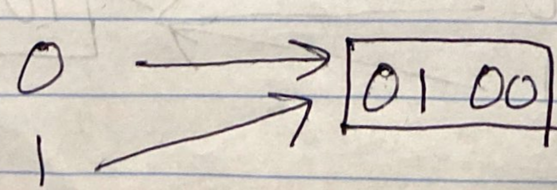
(iv) Delete keys 1 & 2



(v) Delete keys 6 & 7



(vi) Delete keys 0 & 9.



Ans. 9. (a) We will use S as the outer relation

$$IO = b(S) + b(S) * \frac{b(R)}{M}$$

$$= \frac{5000}{10} + \frac{5000}{10} * \left(\frac{1{,}500{,}000}{30}\right) / 101$$

$$= \cancel{1000 + (50{,}000)/101}$$

$$= 500 + 500 * \frac{50000}{101}$$

$$IO = 248{,}024.752$$

(b) Total IO

$= b(R) + b(S) + 2 * b(R) * \text{ceil}(\log_M(b(R))) + 2 * b(S) + \text{ceil}(\log_M(b(S)))$

$= 50000 + 500 + 2 * 50000 * \text{ceil}(\log_{101}(50000)) + 2 * 500 * \text{ceil}(\log_{101}(500))$

$= \boxed{352{,}500}$

(d) $IO = 3 * (b(R) + b(S)) = 3(50000 + 500) = \boxed{151{,}500}$