Homework 4.1

Ans ⇒ 1. Yes, we can achieve O(1) time complexity for searching by doing a ~~trade off~~ tradeoff. ~~but~~
we can do so by declaring an array of size = size of the universe.

→ Space complexity O(R)
→ Time complexity: O(1)

---

2. Yes, we can do that using linked list. No extra space is required.
searching would then take O(n) time.

But since prof. Naveen already mentioned Binary search tree, we can also use that.
Space complexity: O(N)
Time complexity: O(log N)
(searching

Note: Insertion and deletion won't be O(1)

Homework 4.2

Ans⇒ Recursive procedure (Assuming 0-based indexing)

S.No	low	mid	high
1.	0	7	15
2.	0	3	6
3.	4	5	6

Iterative procedure (Assuming 1 based indexing because in algorithm, 1 based indexing is followed)

S.No	low	mid	high
1.	1	8	16
2	1	4	7
3.	5	6	7

Homework 4.3

Ans⇒ Iterative procedure

Every time the loop executes, the size of problem reduces to half of its initial value.

- After 0 iterations N
  After 1 iterations N/2
  After 2 iterations N/4
  ⋮
  After K iterations 1

$2^K = N$ (roughly)

$K = \log_2 N$

Page No.
Date

Also, no extra space is required for iterative Binary search.

Therefore, for iterative binary search.

Space Complexity: O(1)
Time Complexity: O($\log_2 N$)

Recursive Approach

Here also, after every recursion call, size of array in consideration reduces to half of its previous value.

Since we know that during any recursive call, a temporary memory is allocated to variables from stack memory. So, memory needs to be allocated for variable "mid" in each recursive call.

Therefore, for recursive binary search

Space complexity: O($\log_2 N$)
Time Complexity: O($\log_2 N$)

Page No.
Date

Homework 4.4

- We will modify the parameters and use them in hash function as follows.

$$H(person) = (Term_{age} + Term_{name} + Term_{gender}) \% 5$$

where

- $Term_{age} = age/5$
- $Term_{gender}$ = 1 if gender is female and 0 otherwise
- $Term_{name}$ = number of letters in the name.

	Name	Age	Gender	Hash
1.	Rahul Garg	40	M	(8+0+9) % 5 = 2
2.	Rajirekha Sen	35	F	(7+1+12) % 5 = 0
3.	Yogish Subhanwal	40	M	(8+0+15) % 5 = 3
4.	Spongebob Squarepants	10	M	(2+0+19) % 5 = 1