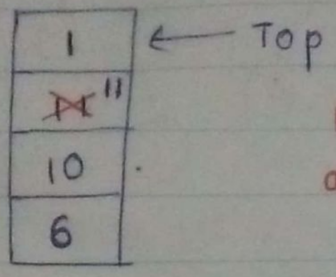
2018 past paper

Question 1

ⓐ



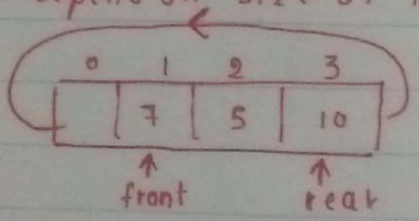
pop → 11
again push → 11 } s1.push(s1.pop())

ⓑ i) Implement peek method using push and pop method.

```
public char peek(){

    char ch = pop();
    push(ch);
    return ch;

}
```

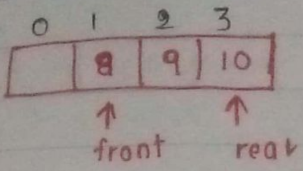
} ← always taking the top value
Constant time.
↑ doesn't depend on the size of the stack.

ii) O(1) ← Because the time is not depend on size of the stack.



ⓒ
```
public void insert (int j)
    if (nItem = = maxSize[])
        System.out.println("Queue is full");
  else {
      if (rear = maxSize[]-1)
        rear = 0;
      else
        rear ++;
  queArray (rear) = j;
  nItem ++;
}
```

(d) rear = maxsize -1 justify answer.

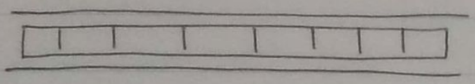


rear = 3

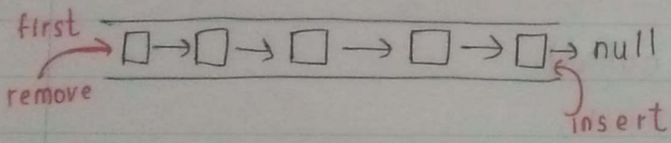
maxSize -1 = 4 -1
= 3

but circular queue ← rear = max size -1
is not full Value can be inserted to index 0
∴ cannot agree.

(e) (i)



using array implement queue



using a link list

remove → Deleting the first link in the link list call the <u>delete first</u>.

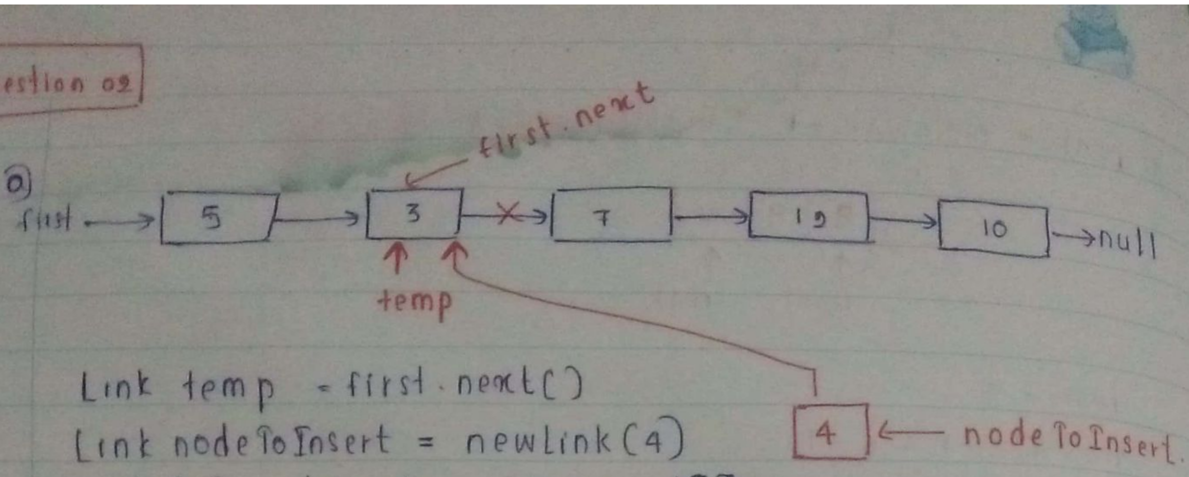
* queue can be implemented using double ended list

insert → addin a link at the end of the link list call the <u>insert last</u>

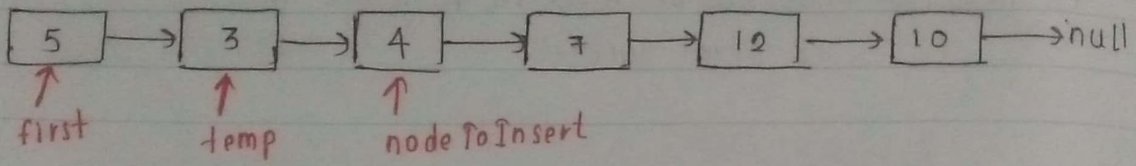
(ii) advantage :-
There is no minimum size when creating Link List but in an array there is limit.

Question 02

a)



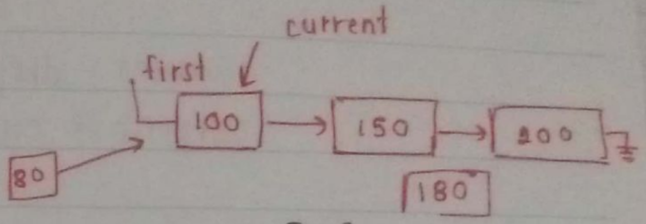
```
Link temp = first.next()
Link nodeToInsert = newLink(4)
nodeToInsert.next = temp.next{};
temp.next = nodeToInsert;
```



b)

i)

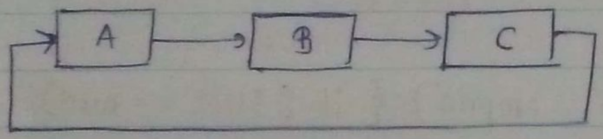
```
class Link{
        public int number;
        public String name;
        public Link next;

    public Link (int no, String name) {
            number = no;
            name = name;
            next = null;
        }
public void display () {
        System.out.println (number);
        System.out.println (name);
}
ii) class Linklist () {
        private Link first;
        public LinkList (){
                first = null;
        }

    }
```



```
iii) public  void SortedInsert (int empNo, String name) {

            Link  newLink  = new  Link (no, name);
            Link  current  = first;
         ~~if(curre~~
            if (no< first.empno) { || (first == null)
                newLink.next  = first
                first  = newLink.next
            }
          else
             {
                Link current  = first;
                while (current.next! = null) && (current.next.
                                               empNo ~~number~~ <no))
                 {
                     current  = current.next;
                     current.next = new Link;


                }
```

iv)
```
public void display Linklist ( ) {

        Link current = first ;

        While ( current ! = null) {

            current . displayLink ( );
            current = current . next ;

        }
```

v) singly circular List ⟶ all the links are connect all togethe

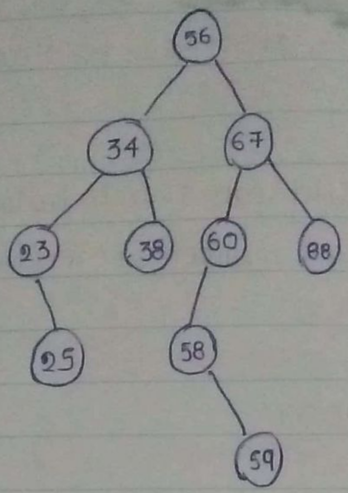


```
public void displayLink ( ) {

     Link current = first ;
     while (current. first != first)
   {
       current. displayLink ( );
       current = current. next;
   }
    current. display Link ( )
```
⟵ To print the current one bast one.

if (current ==
    null)
return;

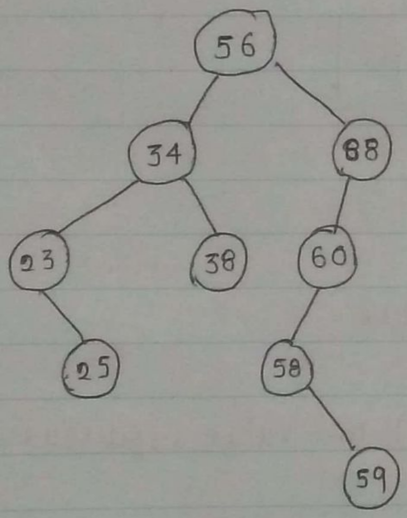
c)

i)



ont

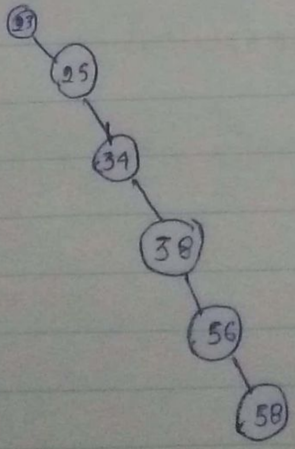
ii)



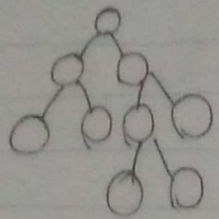
after delete $\boxed{67}$ node

iii) Every node has only one child

Quick sort → 23, 25, 34, 38, 56, 58, 59, 67, 88



iv) disadvantage → if searching it take lot of time something
In here minimum number of steps to find out a number



← easily can search balanced branch minimum number of levels.

Q4) Text = 6750100502007
pattein(p) = 50
q = 10

$$\boxed{p/q \rightarrow \quad 50 \% 10 = 0}$$

67 % 10 → 7

75 % 10 → 5

50 % 10 → $\boxed{0}$ ← value

01 % 10 → 1

10 % 10 → $\boxed{0}$ ← Hit not value sopurious

00 % 10 → $\boxed{0}$

05 % 10 → 5

50 % 10 → $\boxed{0}$ ← value

02 % 10 → 2

20 % 10 → $\boxed{0}$ ← not value spurious

00 % 10 → $\boxed{0}$

07 % 10 → 7

Hits □
not the pattein

A suprious hits and two value hints

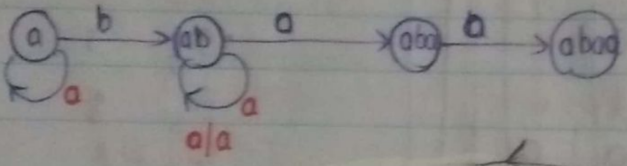
ii) by increasing the q value.

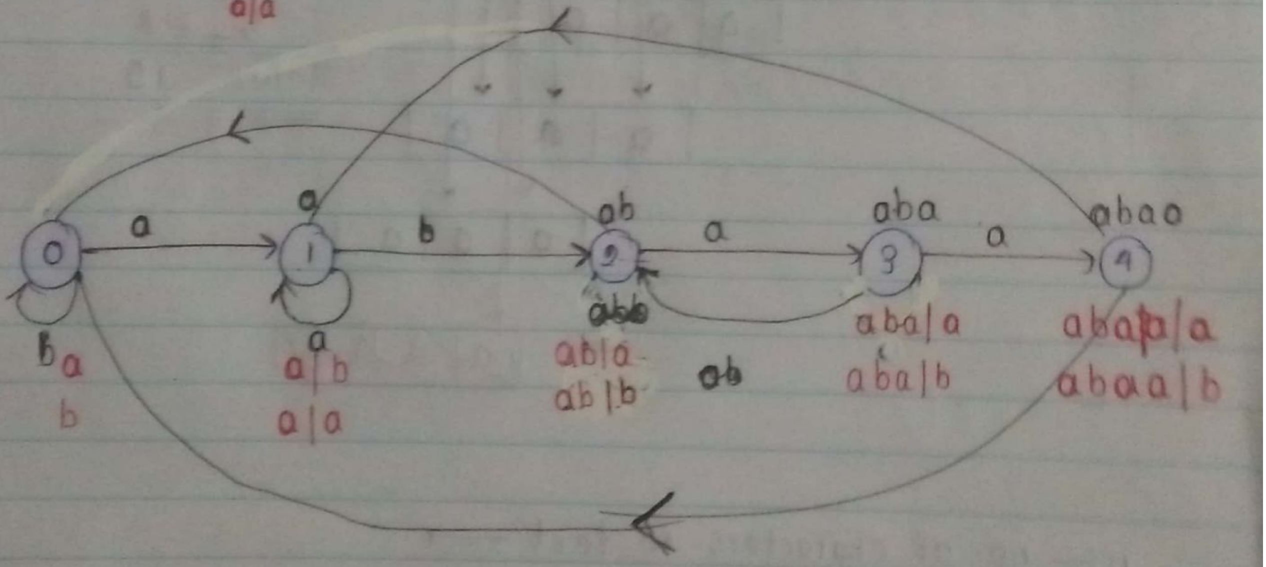
iii) Best case → if we don't get hits like above so we don't wont to futher check it again

→ No of valid hits and no of suprious hits should be zero

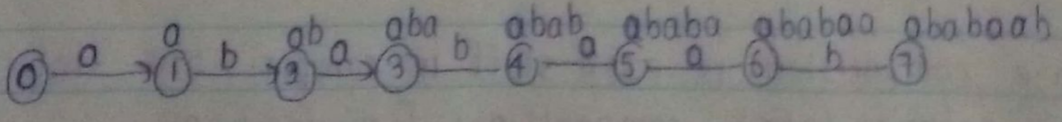
Whatever the conclusion get minimum no of steps

(b) P → abaa





Text → ababaabaaaba

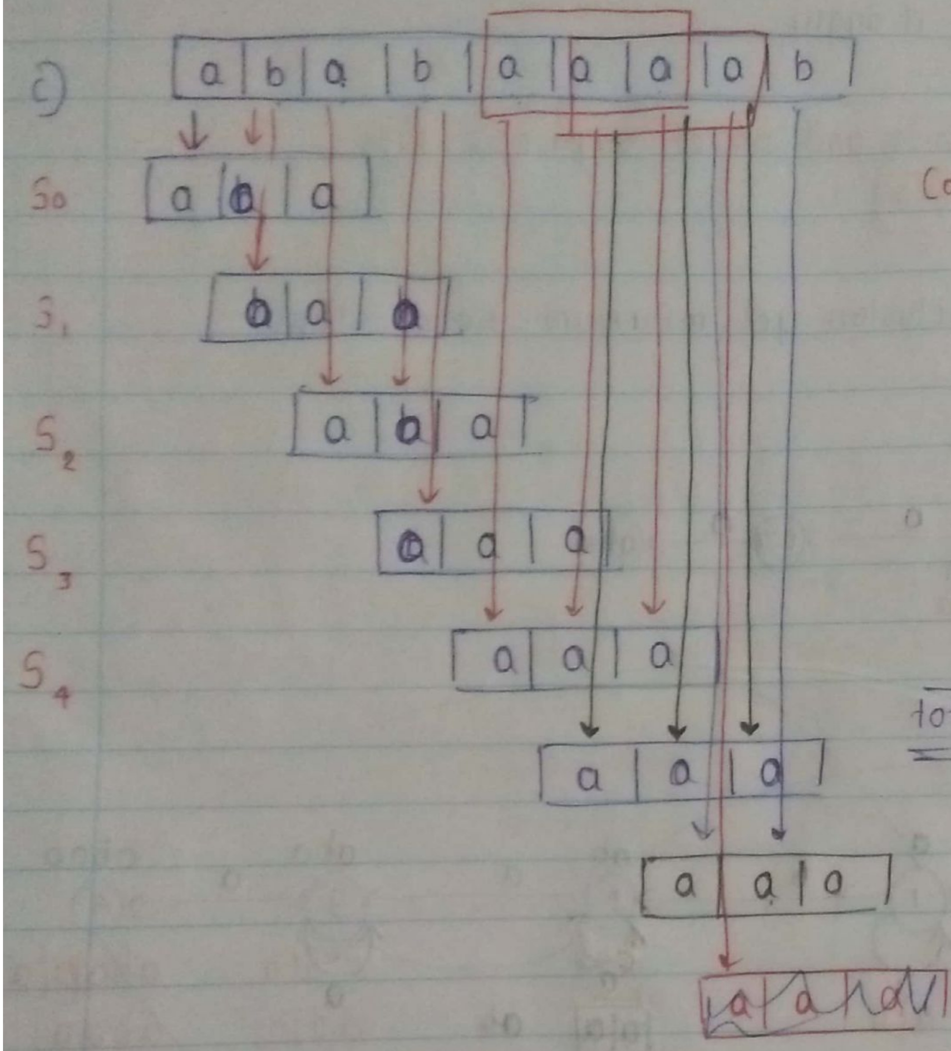


abaa — Pattern

Text : ababaabaaaba

1 2 3 1 2 3 1 2 3 1 2 3

c)



Pattern

a | a | a

Comparision	
S_0	2
S_1	1
S_2	2
S_3	1
S_4	3
S_5	3
S_6	3
~~S_7~~	~~2~~
total	15

n ← no of characters in text ⟶

m ← no of character in Pattern ⟶ 3

no of Shifts ⟶ n - m

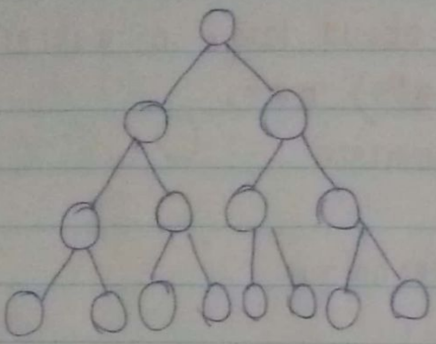
Warst Case comparision is each shift m comparisions should occur Therefore total comparisions $(n-m+1)m$

$\therefore O(m(n-m+1))$

Tute 08

full binary → both side should be balance.

b) $n - 2^{h+1}$



$n = 15$

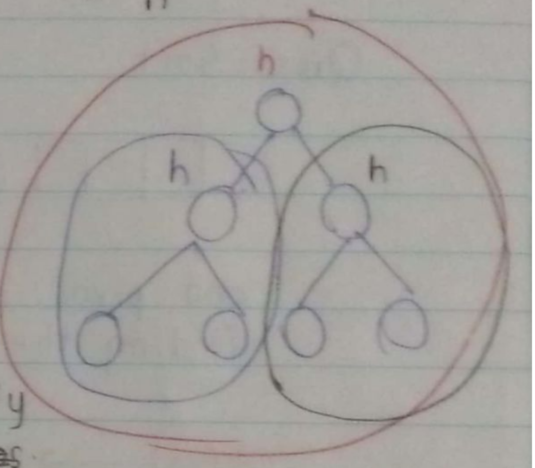
$h = 3$

depth $= 3$

$2^{h+1} - 1 = 2^{3+1} - 1$

$= 15$

$= n$



buid max heap

$$T(n) = \sum_{n=0}^{\log n} \underbrace{\left[\frac{n}{2^{h+1}}\right]} \underbrace{O(h)}$$

maximum number of nodes given ~~each~~ ~~level of h~~ height

big O value of max heapify for given ~~chees~~ subtree of height $(h) = h$

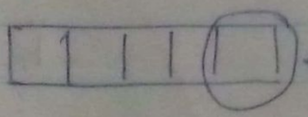
Q3)

a) i) O(nlogn)

ii) $O(2^n)$ ← taking about large n values so $2^n > n8$

b) both sorting Algorithm
both use devider and conqure technique.

Quick Sort



← last element Pivot

and pivot
finding the correct place.

Pivot → coming to first
Best → pivot coming to middle

c) line no 4
if $A[j] \geq x$