UNIVERSITY OF RUHUNA

BACHELOR OF SCIENCE (GENERAL) DEGREE LEVEL I (SEMESTER II) EXAMINATION – NOVEMBER/DECEMBER 2016

SUBJECT : COMPUTER SCIENCE
COURSE UNIT: COM 1213/COM121β (Data Structures & Algorithms) TIME: 2 hours

---

Answer *four (04)* questions *including question one 01*.

1.

a. Calculate the time complexity of the ***Bubble sort*** algorithm for a given array ***A*** of ***n*** elements.

Answer:

In Bubble Sort, n-1 comparisons will be done in 1st pass, n-2 in 2nd pass, n-3 in 3rd pass and so on. So the total number of comparisons will be,
(n-1)+(n-2)+(n-3)+.....+3+2+1
Sum = n(n-1)/2
i.e $O(n^2)$
Hence the complexity of Bubble Sort is $O(n^2)$

b. Write a situation where ***Merge sort*** is the best algorithm to use. Justify your answer.

Answer:

Merge sort is an excellent algorithm for situations where the input cannot all fit into main memory, but must be stored in blocks on an external memory device. Thus, for external memory sorting, the merge sort algorithm tends to minimize the total number of disk reads and writes needed.

c. Write the algorithm for ***Selection sort***.

Answer:

Find the minimum value in the list
Swap it with the value in the first position
Repeat the steps above for the remainder of the list (starting at the second position and advancing each time)

d. Consider the following number set.

47, 20, 54, 7, 32, 18, 59, 27, 61, 2

i. Sort the numbers in ascending order using ***Insertion sort***. Show intermediate steps clearly.

Initial array

47	20	54	7	32	18	59	27	61	2

Current range

1st Pass

20	47	54	7	32	18	59	27	61	2

Current range

2nd Pass

20	47	54	7	32	18	59	27	61	2

Current range

3rd Pass

7	20	47	54	32	18	59	27	61	2

Current range

4th Pass

7	20	32	47	54	18	59	27	61	2

Current range

5th Pass

7	18	20	32	47	54	59	27	61	2

Current range

6th Pass

7	18	20	32	47	54	59	27	61	2

Current range

7th Pass

7	18	20	27	32	47	54	59	61	2

Current range

8th Pass

7	18	20	27	32	47	54	59	61	2

Current range

Final array

2	7	18	20	27	32	47	54	59	61

ii. Search the value 54 using binary search algorithm. Show intermediate steps clearly.

Divide array into two parts.

0	1	2	3	4	5	6	7	8	9
2	7	18	20	27	32	47	54	59	61

Middle value = low+(high-low)/2
=0+(9-0)/2
= 4 (consider only integer value)
Target = 54
Mid = 27
54>27 , hence consider upper part

0	1	2	3	4	5	6	7	8	9
2	7	18	20	27	32	47	54	59	61

Now low= mid+1,
Mid= low+(high-low)/2
= 5+(9-5)/2
=5+4/2
=7
Mid value = 54;
Target value is matched with the mid value. Search terminated.

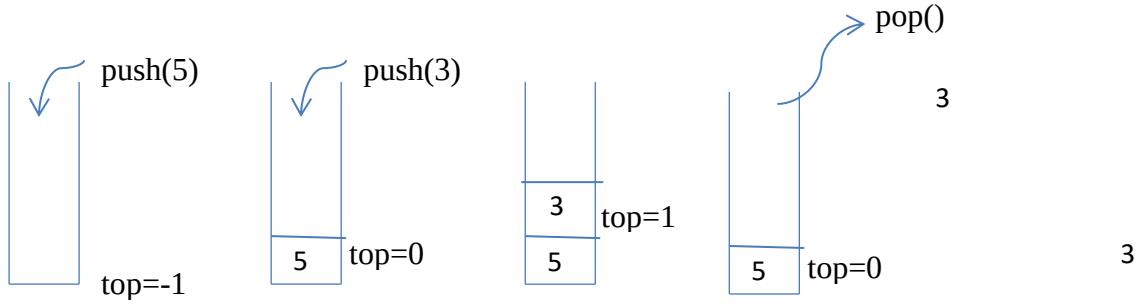
2.

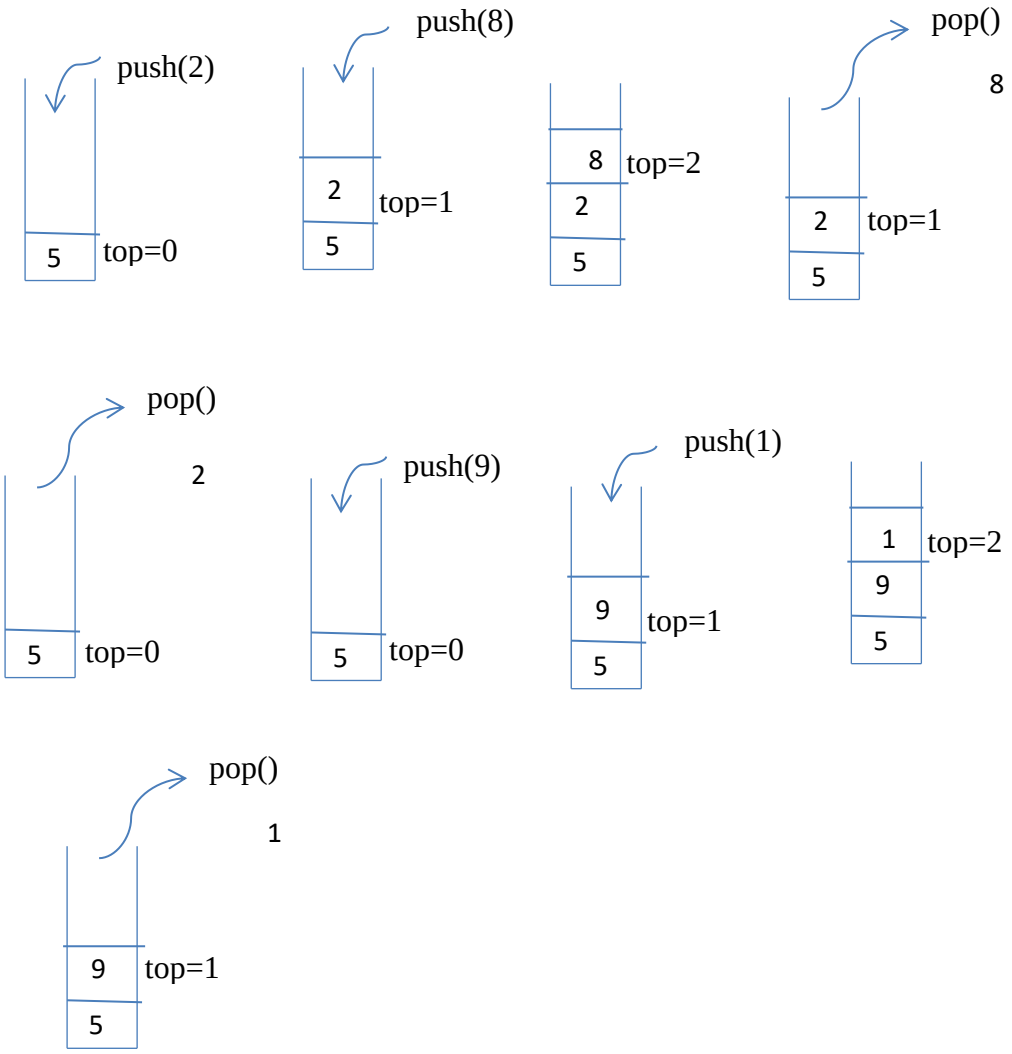
a.

i. Write two (02) applications of stack ADT.
Answer:

- ***Mathematical expression evaluation***
- ***Balanced spell checker –parenthesis balance checker***
- ***Simple calculator***

ii. Describe the output of the following series of stack operations giving the content of the stack after each operation.
push (5), push (3), pop (), push (2), push (8), pop (), pop (), push (9), push (1), pop ()





b. What is the drawback in normal queue ADT which is addressed by circular queue ADT?

Answer:

The normal queue logically moves in the array from left to right.
After several moves, rear reaches the end, leaving no space for adding new elements. But there is a free space before the front index. It is wasting space. And circular queue has solved this issue by using that space for enqueueing new items, i.e. the next entry will be stored at index 0, then 1, until front

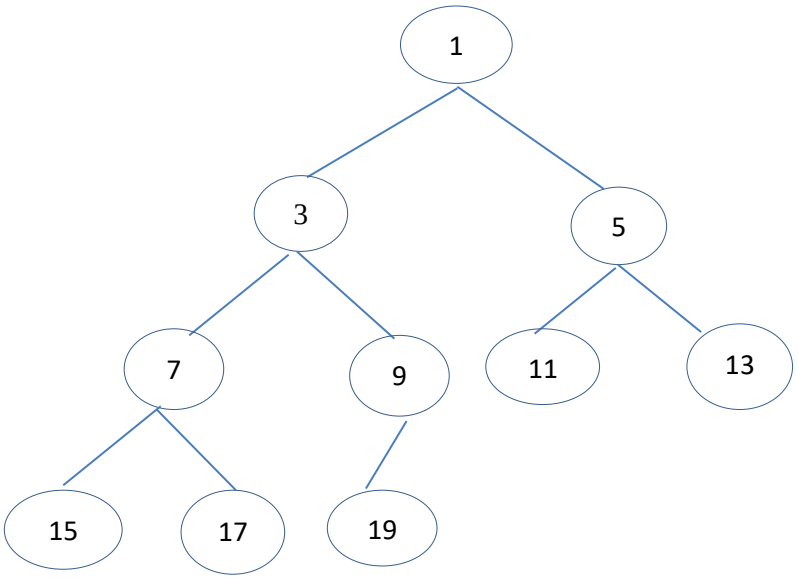
c. Explain the concept of a priority queue and discuss a situation in where this can be useful.

Answer:

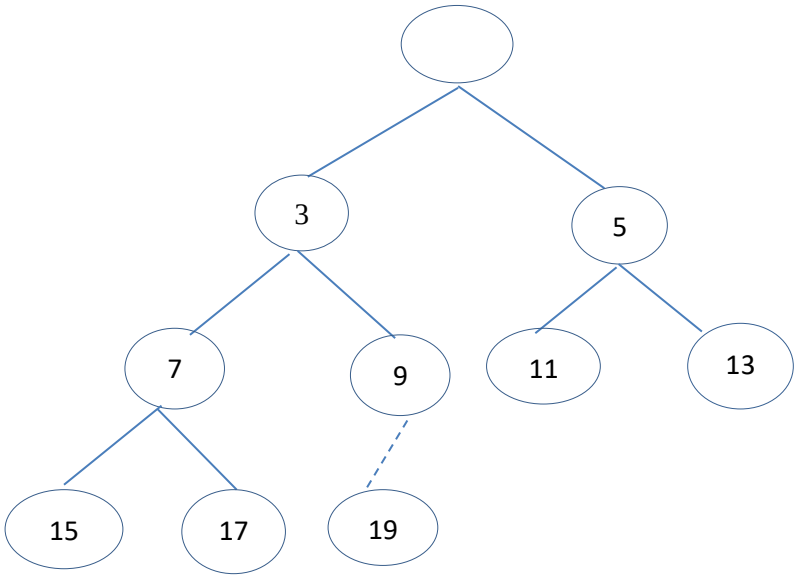
In Priority queue items are ordered by key value so that item with the lowest value of key is at front and item with the highest value of key is at rear or vice versa. A priority queue is different from a "normal" queue, because instead of being a "first-in-first-out" data structure, values come out in order by priority.
Situation-
Handling jobs sent to helpdesk of an organization. Most prioritized job will be done before the least prioritized jobs.

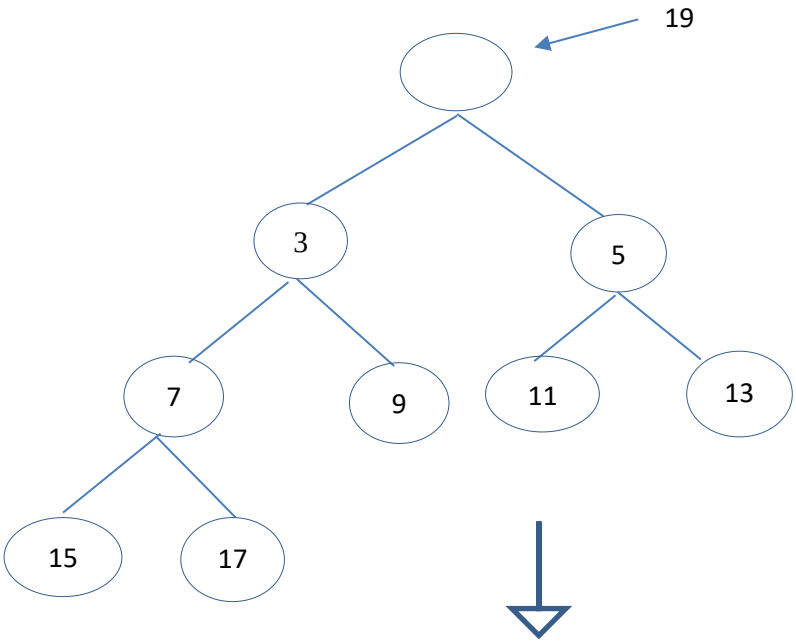
d.

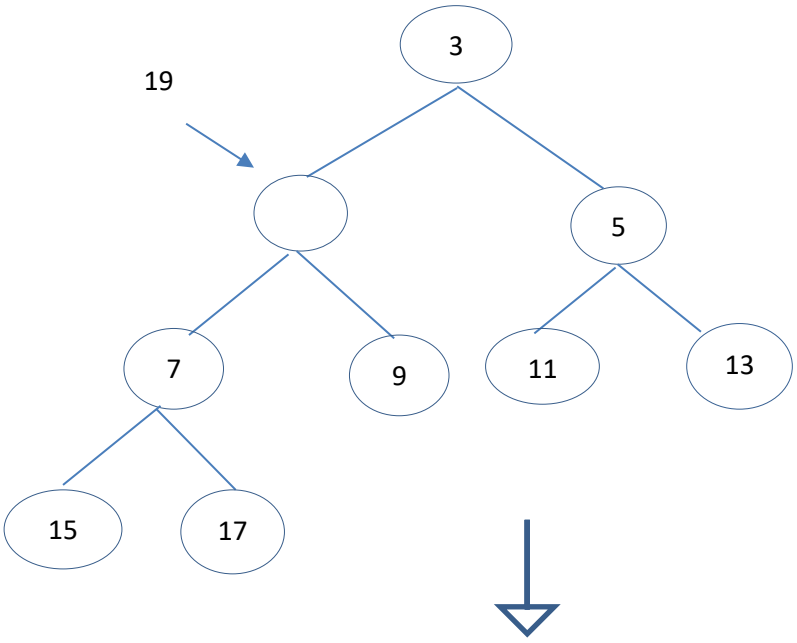
i. Draw a min heap whose keys are all the odd numbers from 1 to 20 (with no repeats)

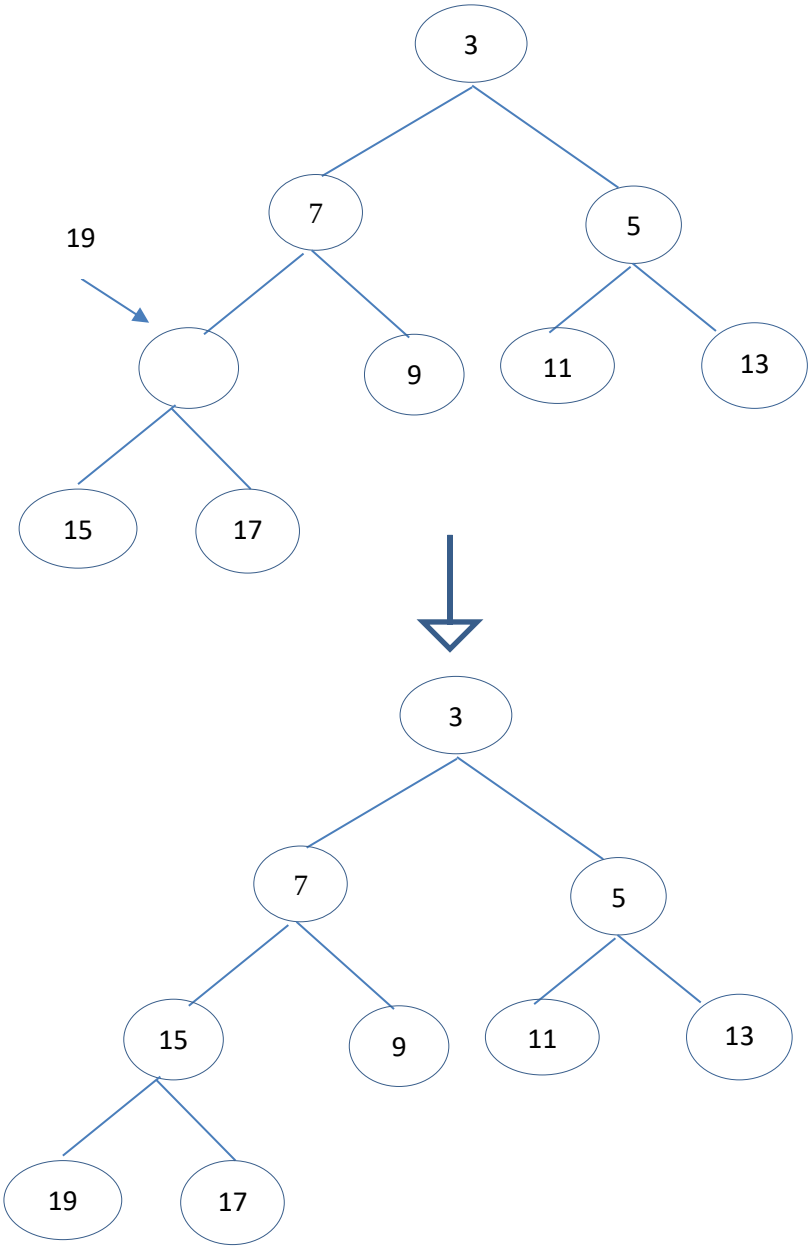


ii. Redraw the above binary heap after deleting the minimum value. Show intermediate steps clearly.




19
3
5
7
9
11
13
15
17


3
19
5
7
9
11
13
15
17



3.

a.

i. Write two (02) advantage of using a linked list over an array in implement Stack ADT.

Answer:

- ***Arrays are not very adaptable. For instance, we should fix the size n of an array in advance, which makes resizing an array difficult. But when using linked list to implement stack ADT, we can allocate memory dynamically***.

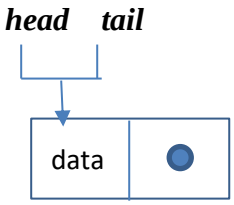
- ***Inserting a new element into an array is expensive, because room should be created for the new element and to create room, existing elements must shift, but in linked list insertion elements are easy.***

b. Explain following operations in singly linked list by providing suitable illustrations.

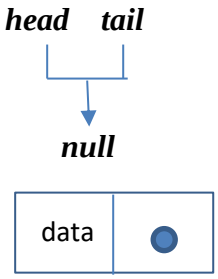
i. Delete a node from a list where it is having only one node.

Answer:

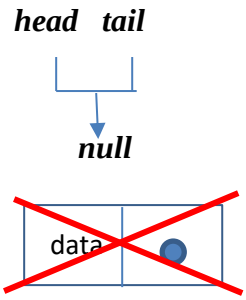
In a list with one node, head points to the same node as the tail.



Set both head and tail to NULL.



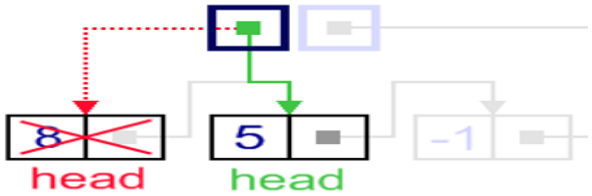
Delete the node.



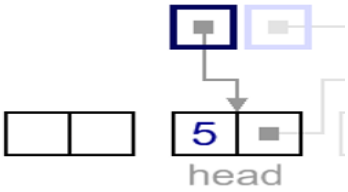
ii. Remove the first node from the linked list.

Answer:

Update head pointer to the next node of the current head node



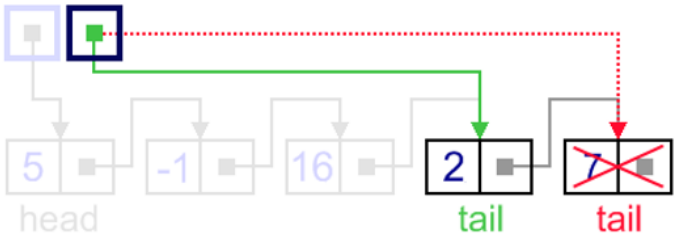
Delete the current head node.



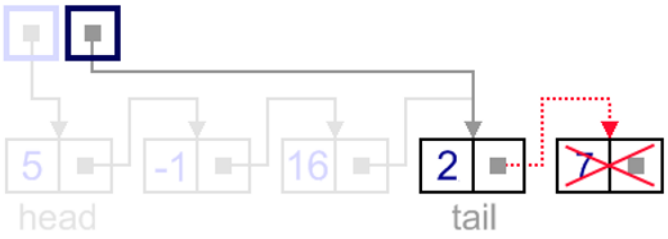
iii. Remove the last node from the linked list.

Answer:

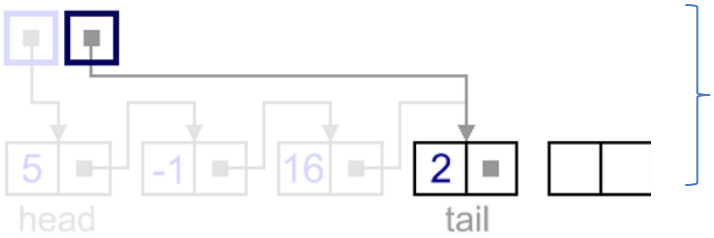
Update tail link to point to the node, before the tail. To find it, list should be traversed first, beginning from the head.



Set next link of the new tail to NULL.



Delete the node.



c.

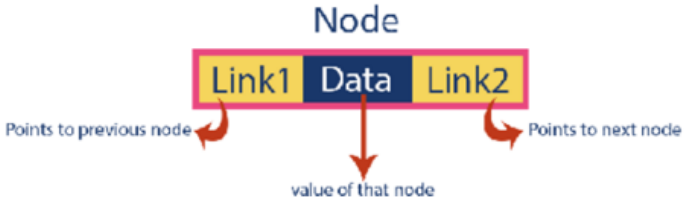
i. What is the drawback of a singly linked list which is addressed by a doubly linked list?

Answer:

In a single linked list, every node has link to its next node in the sequence. So, we can traverse from one node to other node only in one direction and we cannot traverse back.

ii. Draw the structure of doubly link list node.

Answer:



d. Consider the following scenario and propose an appropriate linked list to implement the given scenario.

Suppose there are multiple applications running on a PC. Therefore, the operating system will put the running applications on a list and the operating system gives a fixed time slot to all for running. The operating system keeps on iterating over the list until all the applications are completed.

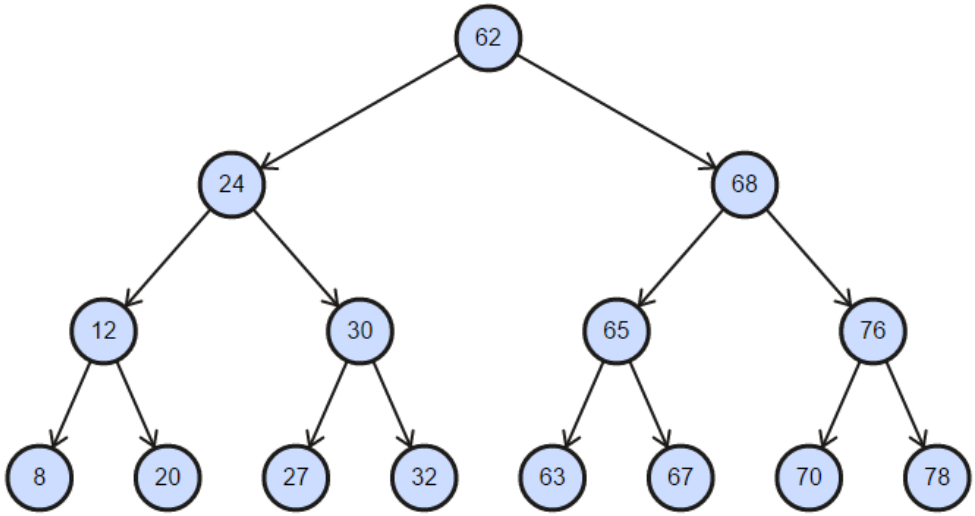
Answer:
Circular linked list

4.

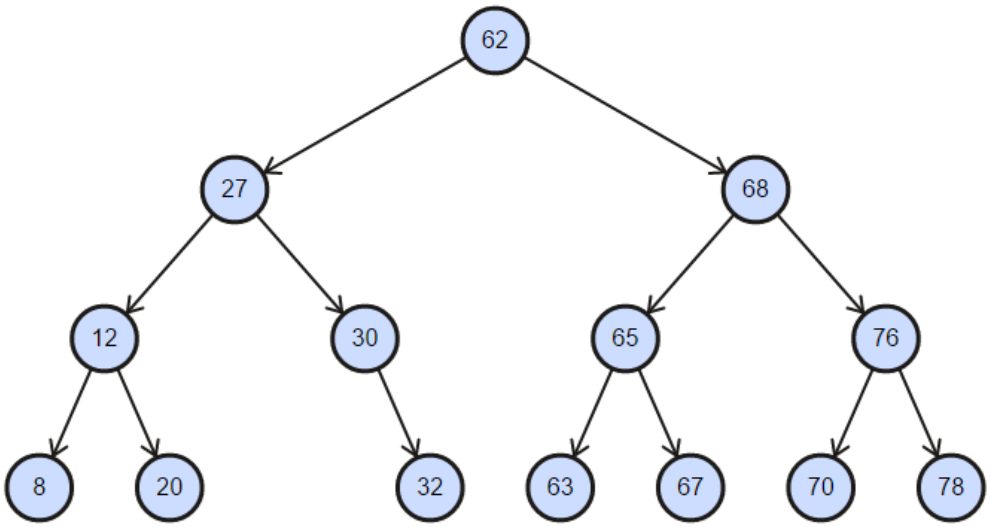
a. What is the main properties of a strictly binary tree?
Answer:
Every non-leaf node in a strictly binary tree has non-empty left and right sub-trees.

b. Consider following elements.
62,24,30,12,68,76,78,65,70,63,67,8,20,27,32

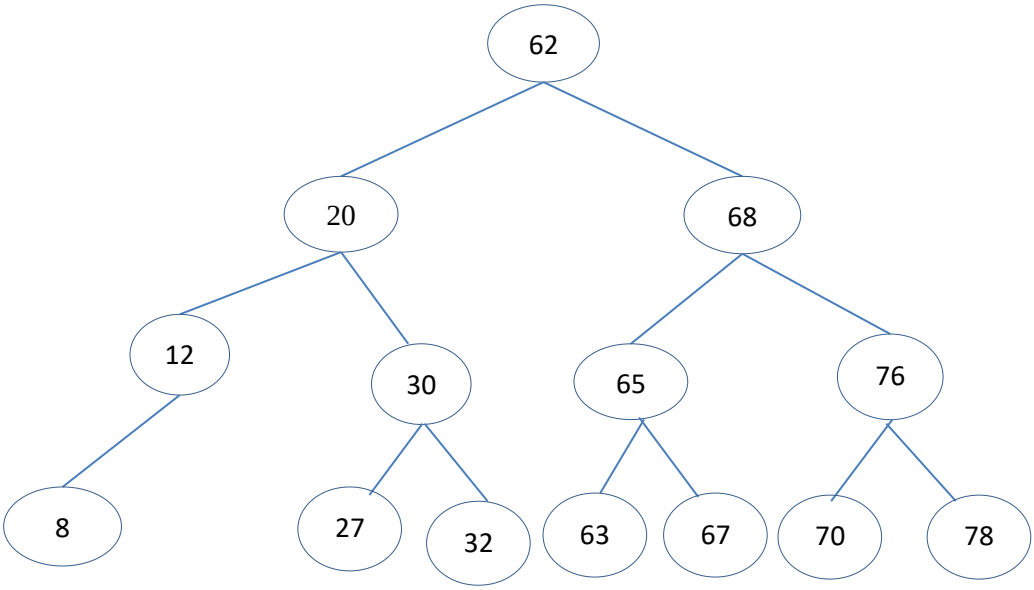
i. Construct a binary search tree.



ii. Delete 24 from the tree and give two separate resulting trees that can be obtained after deleting the node.
Answer:
Tree 1



Tree 2



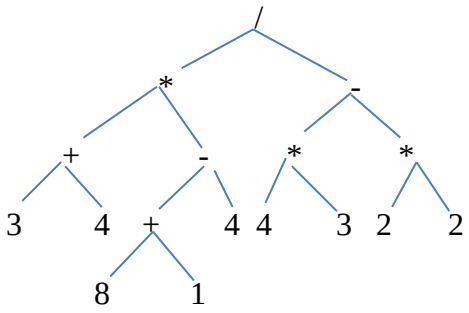
iii. Write post order traversal for one of the above resulting trees given in (ii).
Answer:
Postorder-
Tree 1
8,20,12,32,30,27,63,67,65,70,78,76,68,62

Tree 2 –
8,12,27,32,30,20,63,67,65,70,78,76,68,62

c. Consider the following expression.

((3 + 4) * (8 + 1 - 4))/ ((4 * 3) - (2 * 2))

i. Draw an expression tree for the above expression.



ii. Write the equivalent prefix and postfix expressions.
Answer:
Prefix:
/ * + 3 4 - + 8 1 4 - * 4 3 * 2 2
Postfix
3 4 + 8 1 + 4 - * 4 3 * 2 2 * - /

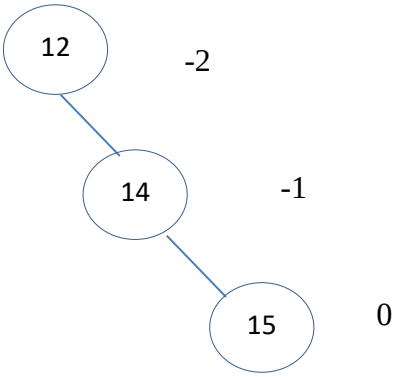
d.

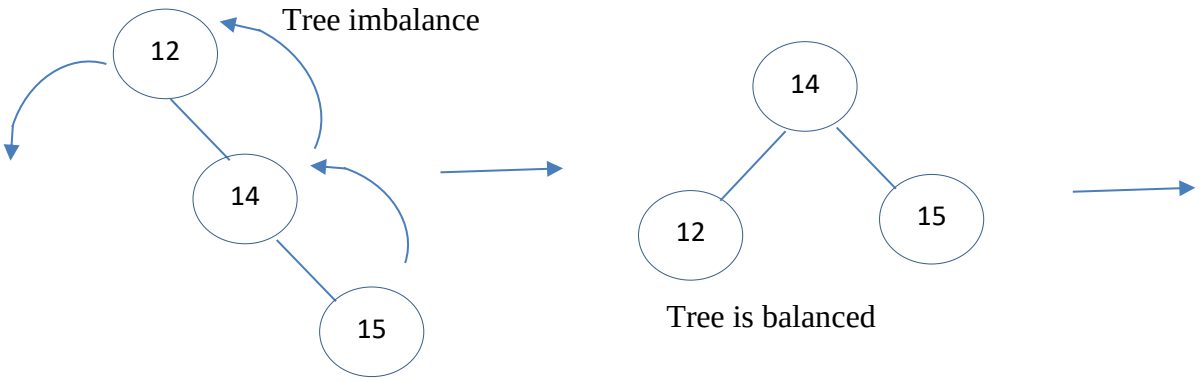
i. Write the structural properties of an AVL tree.
Answer:

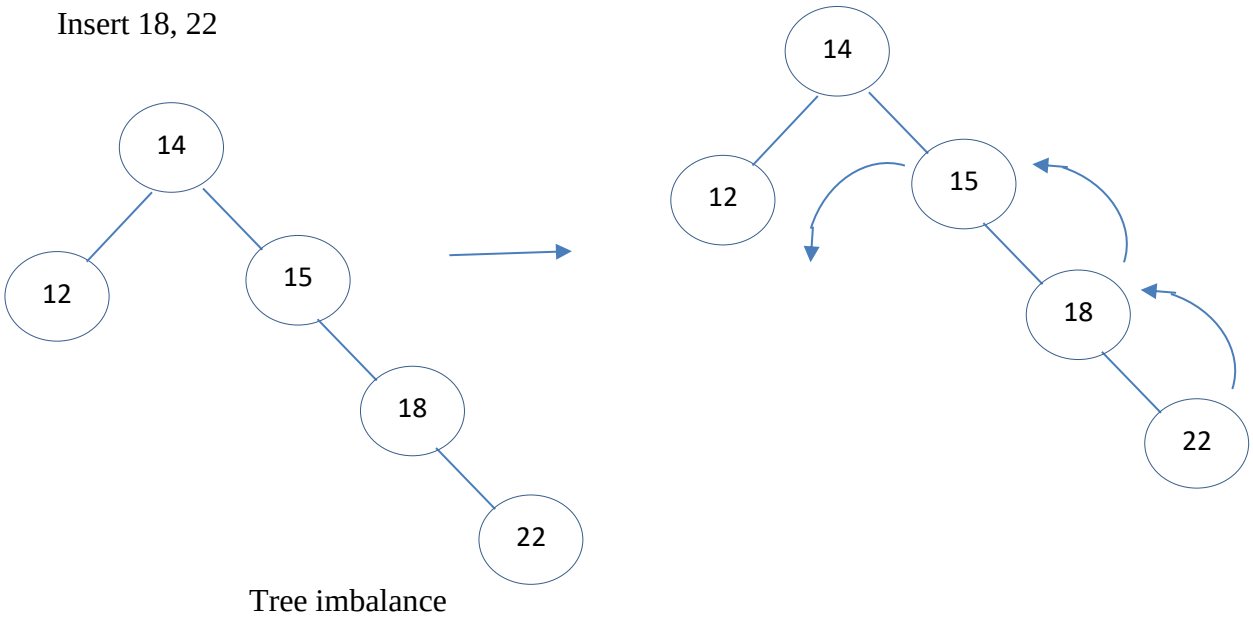
- ***Binary tree property***
- ***Balance property:***
  - ***balance of every node is between -1 and 1***
- ***Ordering property***
  - ***Same as for BST***

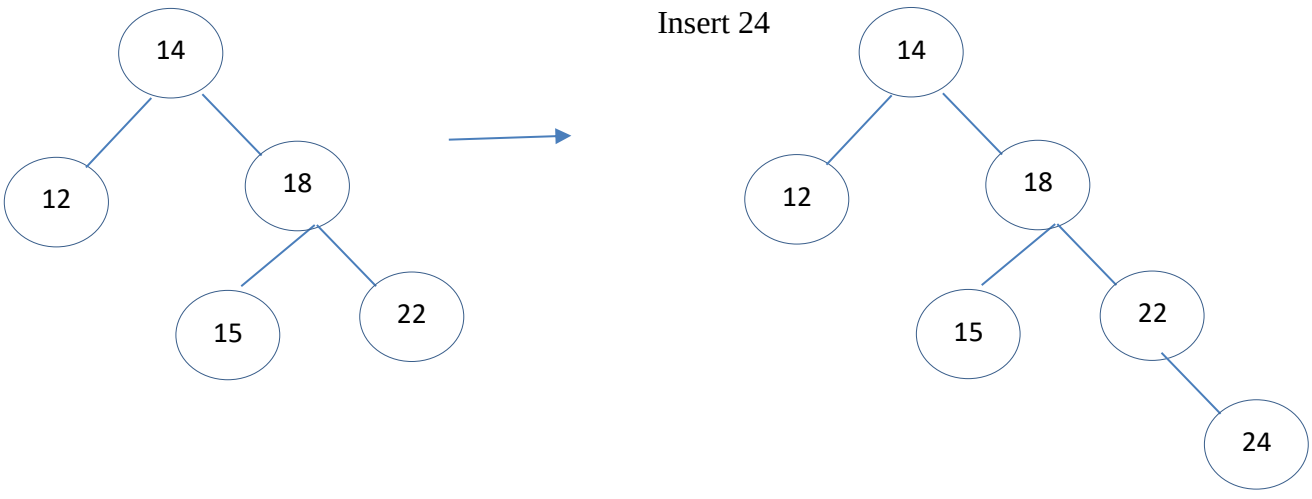
ii. Construct an AVL tree for the following elements showing intermediate steps clearly.

12, 14, 15, 18, 22, 24, 28
Answer:
Insert 12,14,15

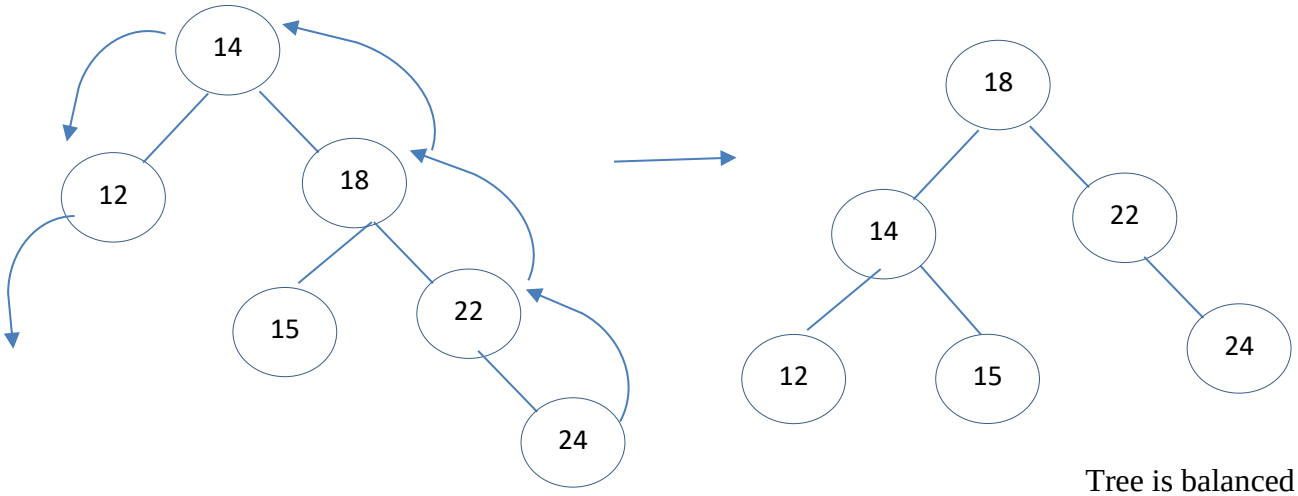



Tree imbalance
12
14
15
14
12
15
Tree is balanced


Insert 18, 22
14
12
15
18
22
Tree imbalance
14
12
15
18
22

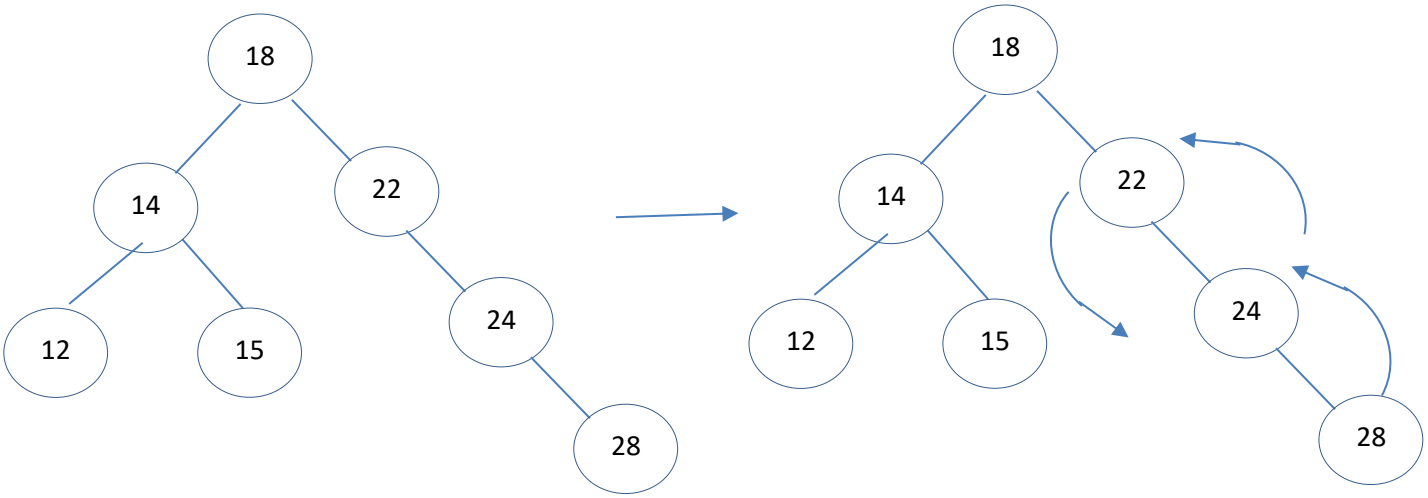

14
12
18
15
22
Insert 24
14
12
18
15
22
24

Tree imbalanced

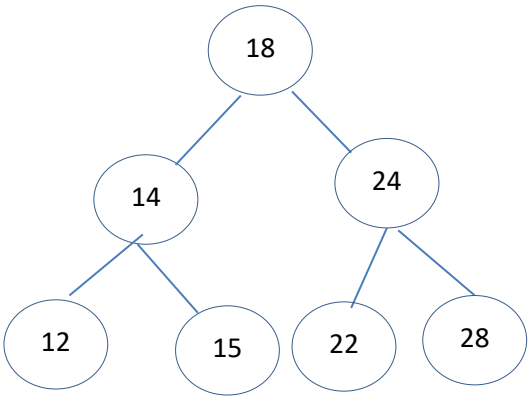


Tree is balanced

Insert 28



Tree imbalanced



Balanced final tree

5.

a.

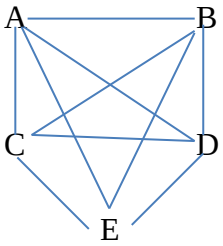
i. Suppose G1 is a complete graph with 5 vertices. Write the formal definition of the graph G1.

Answer:

G(5,10)

ii. Draw the graph corresponding to the above specification.

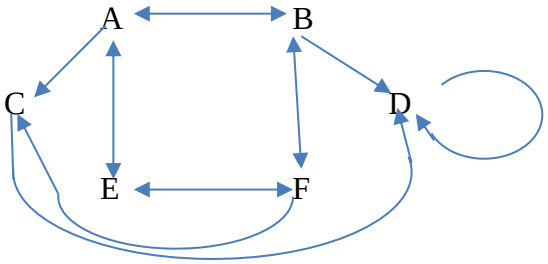
Answer:



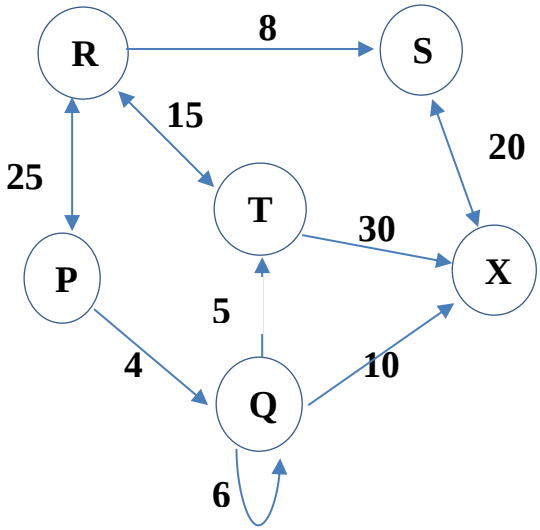
b. Draw the corresponding directed graph to the following adjacency matrix.

	A	B	C	D	E	F
A	0	1	1	0	1	0
B	1	0	0	1	0	1
C	0	0	0	1	0	0
D	0	0	0	1	0	0
E	1	0	0	0	0	1
F	0	1	1	0	1	0

Answer:



c. Consider the following weighted graph.



i. Write the adjacency matrix for the above given graph.
Answer:

	P	Q	R	S	T	X
P	∞	4	25	∞	∞	∞
Q	∞	6	∞	∞	5	10
R	25	∞	∞	8	15	∞
S	∞	∞	∞	∞	∞	20
T	∞	∞	15	∞	∞	30
X	∞	∞	∞	20	∞	∞

ii. Write the adjacency list for the given graph.
Answer:

P	→	Q	4	→	R	25			
Q	→	Q	6	→	T	5	→	X	10
R	→	P	25	→	S	8	→	T	15
S	→	X	20						
T	→	R	15	→	X	30			
X	→	S	20						

d.

i. Explain a situation where a collision can occur in a hash table.
Answer:
When two elements are trying to insert into same location in the hash table a collision can occur.

ii. Write three methods of collision resolution.
Answer:
Open addressing
Chaining
Bucket addressing

iii. Insert the following elements into a hash table assuming that ***TableSize***= 7 and the hash function is ***h(k)= k%TableSize***.

18, 21, 75, 48, 52

Answer:

18% 7= 4
21%7=0
75%7= 5
48%7= 6
52% 7= 3

05marks

21	0
	1
	2
52	3
18	4
75	5
48	6

Hash Table

****