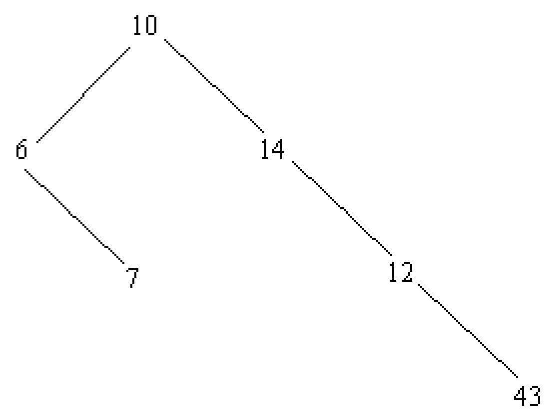
Chapter 6

6.1 Construct a BST for the following set of numbers.

10, 6, 7, 12, 14, 43

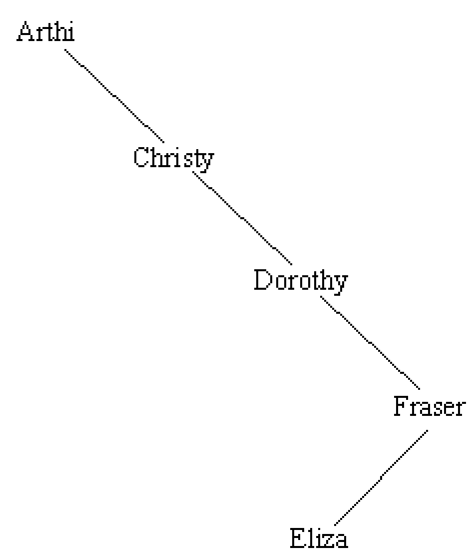
***Solution*:**



6.2 Construct a BST for the following set of names

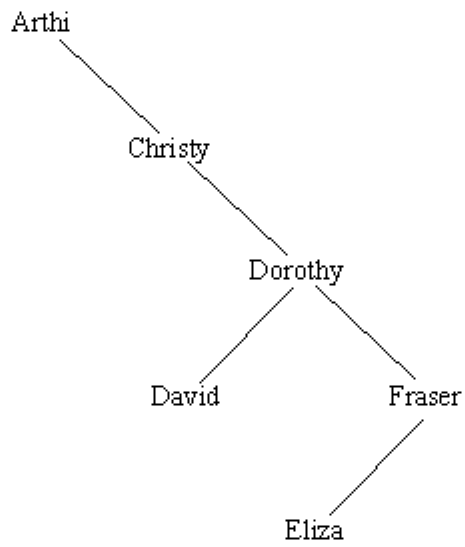
Arthi Christy Dorothy Fraser Eliza

***Solution*:**



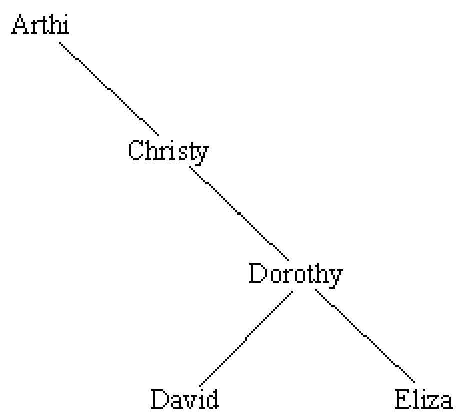
a) Insert a name 'David' into BST

***Solution*:**



b) Delete the name Fraser from the BST

***Solution*:**



6.3 Write a procedure to find maximum element in a BST

Solution:

Algorithm BST-minimum(x)

%% Input : x tree

%% Output : minimum element

Begin

while left[x] $\neq$ Null do

$x \leftarrow$ left[x]

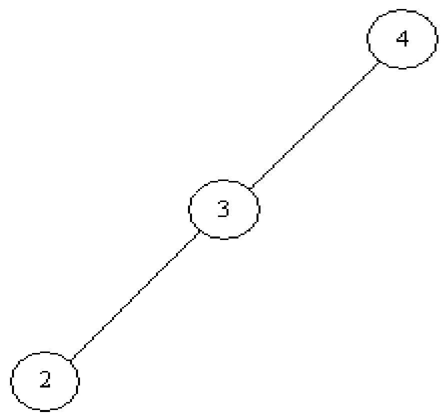
end while

return *x*

End

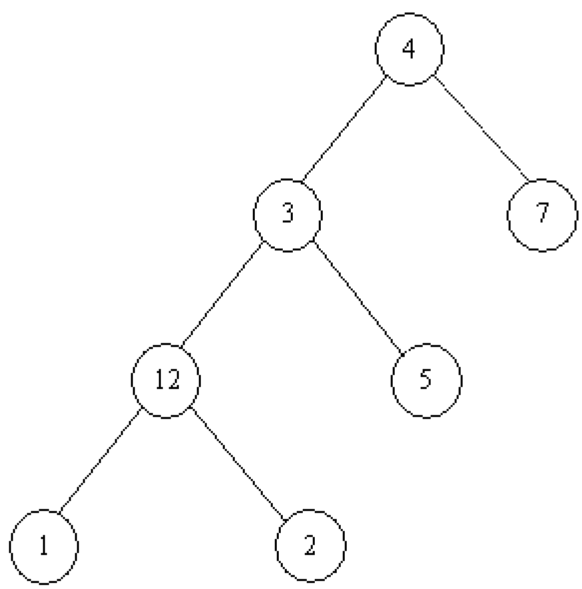
6.4

a) Identify the following trees one BST's or not



***Solution*:** This is a BST

b) Check whether these trees are AVL trees or not. Compute the balance factor for these trees.



***Solution*:**

This is not AVL

Balance factor for 12 = 0

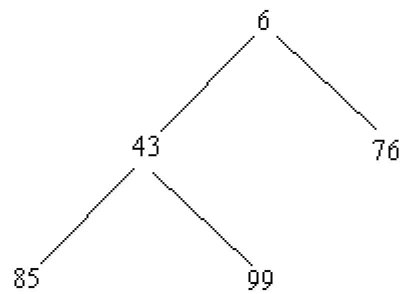
for 3 = 1

for 4 = 2

6.5 Construct a minimum-heap for the following set of numbers.

6, 43, 76, 85, 100

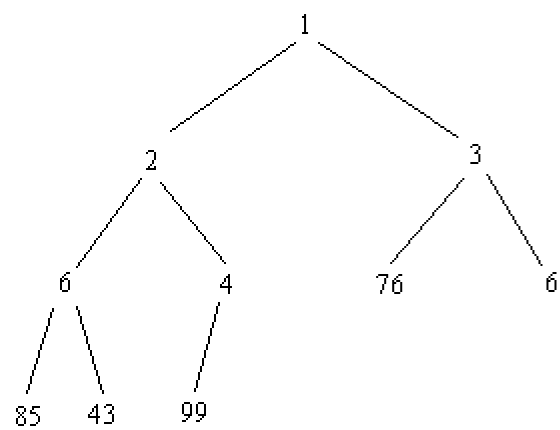
***Solution*:**



Step 1 : I

nsert the following numbers into minimum-heap tree

3, 2, 1, 6, 4

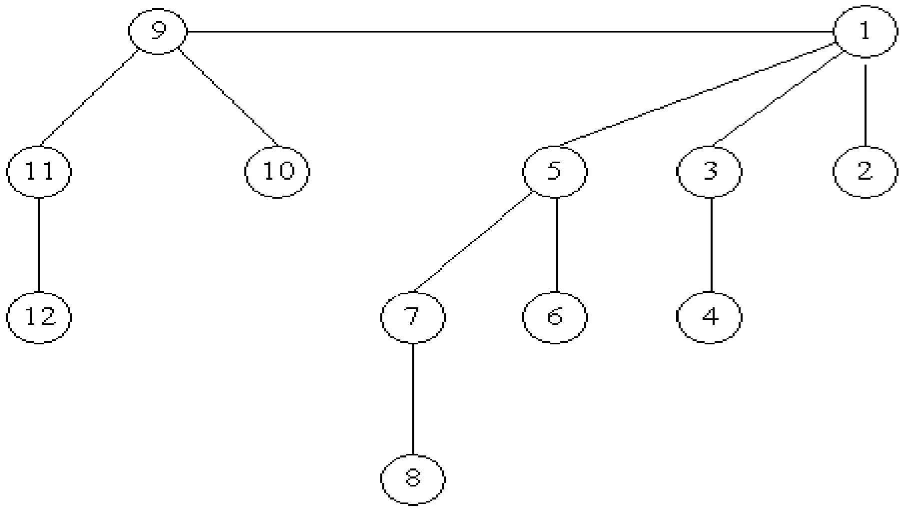


6.6 Construct a binomial tree for 12 and 14 elements.

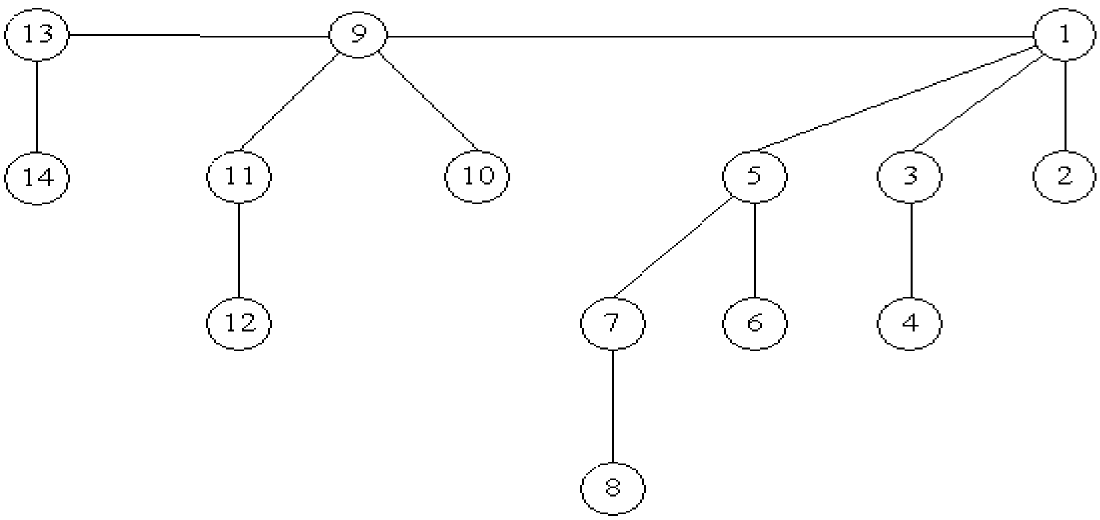
Solution :

Binomial tree for 12 elements would appear like this assuming the elements are

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12.



After inserting the element 13 and 14, the binomial tree would appear like



6.7 Write a procedure for melding binomial trees.

The procedure is given informally as follows for melding H_1 and H_2 like binary addition. The number of B_j's at location J $[0,k]$ is

Step 0 : Location j of H is set to null.

Step 1 : Location of j of H points to B_j.

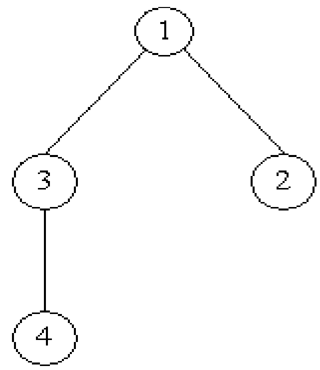
Step 2 : Two B_j's are linked together to form B_{j+1} which is stored as carry at location j+1 of H, and location j is set to Nil.

Step 3 : The carry is stored at j+1 of H and 3rd B_j is stored at location j.

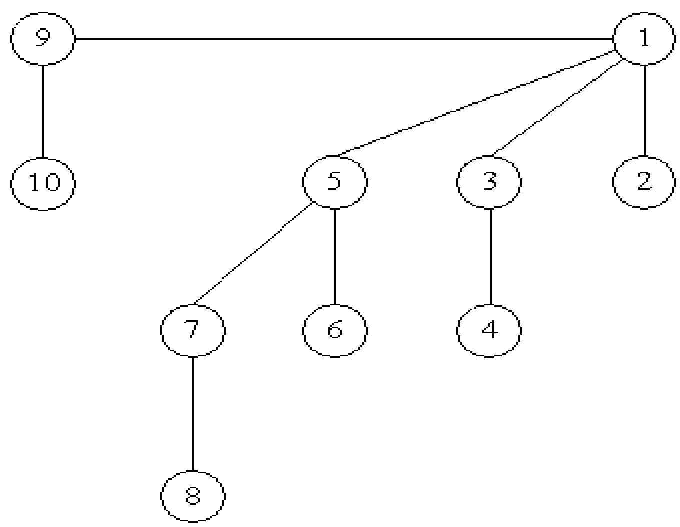
6.8 Construct a binomial tree for 10 and 4 elements. Show that they can be melded.

***Solution*:**

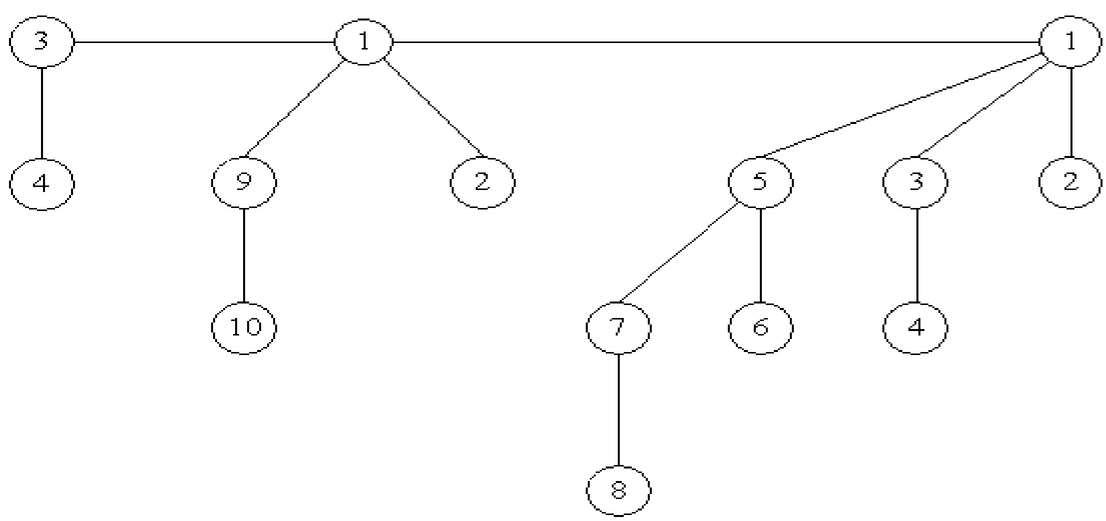
For 4 elements



For 10 elements



After merge



6.9 Write a procedure for decreasing key in F-heap.

***Solution*:**

Let Q be the heap

key be key

Algorithm decrease-key(u,key)

Begin

if key > $V.key$ then

return (0)

else

$v.key = key$

update minimum element 0

end if

if $v \in rootlist$ of heap or $key \geq v.parent.key$ then return(0)

do

$parent = v.parent$

$Q.cut(v)$

$v = parent$

while $v.mark$ and $v \notin Q.rootlist$

if $v \notin Q.rootlist$ then

$v.mark$ = true

End

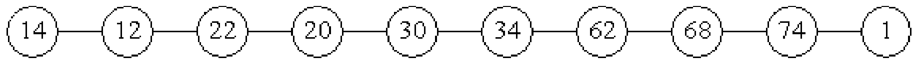
6.10 Construct a F-tree for 10 and 14 elements.

***Solution*:**

Assuming 10 elements

1, 14, 12, 22, 20, 30, 34, 62, 74

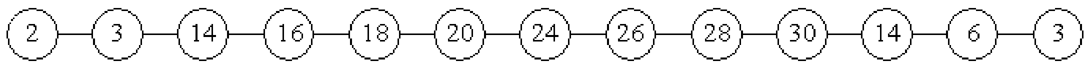
The F-tree would appear like this.



The 14 elements

{ 1, 2, 3, 14, 16, 18, 20, 24, 26, 28, 30, 14, 6, 3}

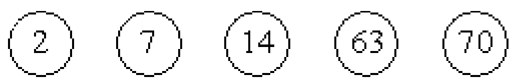
would appear like this



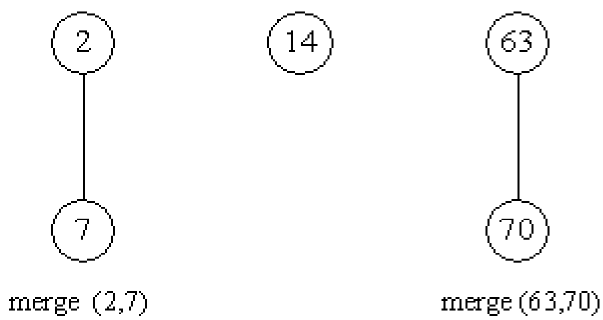
6.11 Construct a disjoint set for the following set of numbers.

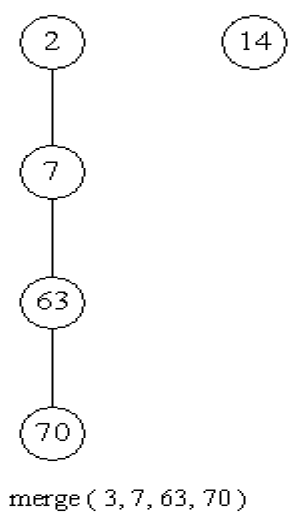
***Solution*:**

The disjoint set would be like this



Merge results in





6.12Write a procedure for performing merging by size and rank for disjoint set.

***Solution*:**

The procedure for merging using size is given below:

Step 0 : Assuming disjoint sets are *i, j*.

$root\ 1 = find(i)$

$root\ 2 = find(j)$

Step 1 : If roots are not equal and if first tree *I* has more nodes than *j* then

$parent[root\ 1] = parent[root\ 1] + parent[root\ 2]$

$parent[root\ 2] = root\ 1$

else

$parent[root\ 2] = parent[root\ 2] + parent[root\ 1]$

$parent[root\ 1] = root\ 2$

endif

endif

Step 2 : End

(***ps***: By rank is given in the book)

6.13 Design a binary counter and apply amortized analyzed to it.

***Solution*:**

The procedure for binary counter

A[0] = A[0] + 1

i = 0

while [A(*i*) = 2] do

A[*i* + 1] = A[*i*+1] + 1

A[*i*] = 0

end while

End

A[0] flips in each increment

A[1] flips in every second increment ($^n/_2$ times)

A[2] flips in every fourth increment ($^n/_4$ times)

$\vdots$

A[i] flips in every 2^i th increment $\left(\frac{n}{2^i}\right)$

$$T(n) = \sum_{i=0}^{\log n} \frac{n}{2^i}$$

$$\therefore \quad \le n \times \sum_{i=1}^{\log n} \left(\frac{1}{2}\right)^i$$

$$= O(n)$$

6.14 Design a table that gets enlarged automatically when the size is violated. Apply amortized analysis to it.

***Solution*:**

Initially the table is null. The table would be enlarged when table space is not enough. i^{th} operation causes in expansion only when i-1 is power of 2.

$$\therefore \quad C_i = \begin{cases} i & \textit{if } i \textit{ is exact power of } 2 \\ 1 & \textit{otherwise} \end{cases}$$

$$\sum C_i = n + \sum_{j=0}^{\lfloor \log n \rfloor} 2^j$$

$$\therefore \quad < n + \sum_{j=0}^{\log n} 2^j$$

$$< n + 2^n$$

$$= 3n$$

$$\therefore \text{ The amortized cost of } \frac{3n}{n} = 3 = O(1)$$

This is the analysis using aggregate method.