TUGAS

Latihan BST

Tugas mata kuliah Struktur Data dan Pemrograman Berorientasi Objek (B)

Dosen Pengampu : Dr.techn. Ir. Raden Venantius Hari Ginardi, M.Sc

Di Susun Oleh :

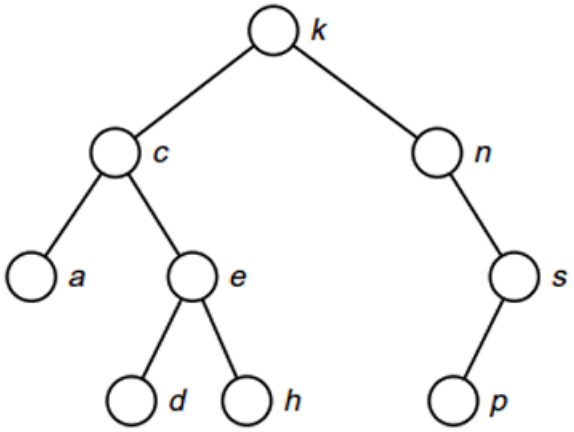
Nabiel Nizar Anwari – 5027231087

INSTITUT TEKNOLOGI SEPULUH NOPEMBER

S1 – Teknologi Informasi 2023

Data Structures and Program Design in C++ (Kruse)

Pages 458-460 (Exercise 10.2)



E1. Show the keys with which each of the following targets will be compared in a search of the preceding binary search tree.

a) C = root - c
b) S = root - n - s
c) K = root
d) A = root - c - a
e) D = root - c - e - d
f) M = root - n - (m not found)
g) F = root - c - e - h - (f not found)
h) B = root - c - a - (b not found)
i) T = root - n - s - (t not found)

E2. Insert each of the following keys into the preceding binary search tree. Show the comparisons of keys that will be made in each case. Do each part independently, inserting the key into the original tree.

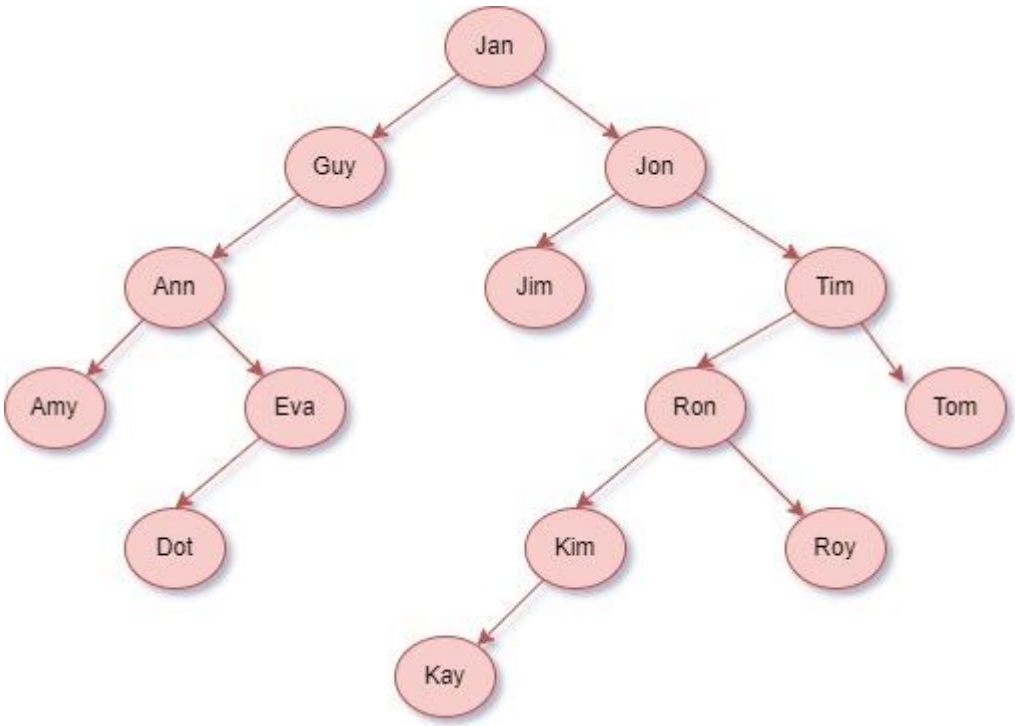
a) M = k (go right) -> n (go left, as left child)
b) F = k (go left) -> c (go right) -> e (go right, as right child)
c) B = k (go left) -> c (go left) -> a (go right, as right child)
d) T = k (go left) -> n (go right) -> s (go right, as right child)
e) C = k (go left, replace existing c node)
f) S = k (go right) -> n (go left, replace existing c node)

E3. Delete each of the following keys from the preceding binary search tree, using the algorithm developed in this section. Do each part independently, deleting the key from the original tree.

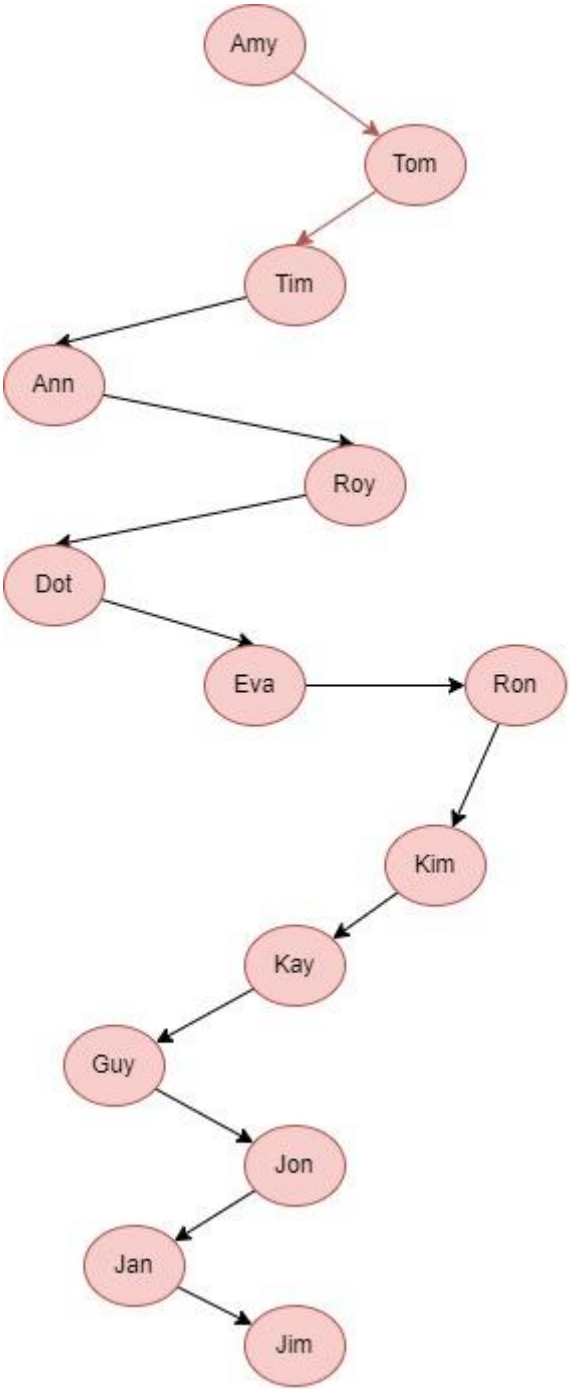
a) A = root -> c -> a -> delete (no node movements)
b) P = root -> n -> s -> p -> delete (no node movements)
c) N = root -> n -> delete (move root - right to s)
d) S = root -> n -> s -> delete (move n - right to p)
e) E = root -> c -> e -> delete (move c - right to d; d - right to h)
f) K = root -> delete -> find successor (root = h)

E4. Drawthebinarysearchtreesthat function insert will construct for the list of 14 namespresentedineachofthefollowingordersandinsertedintoapreviously empty binary search tree.

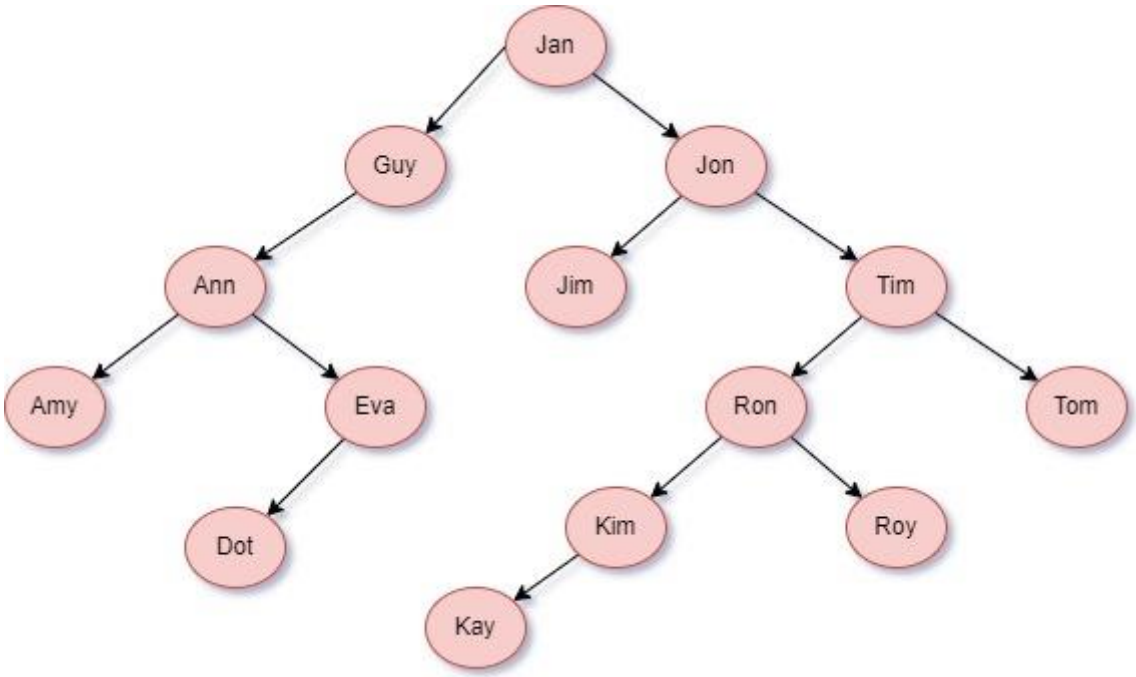
a) Jan Guy Jon Ann Jim Eva Amy Tim Ron Kim Tom Roy Kay Dot



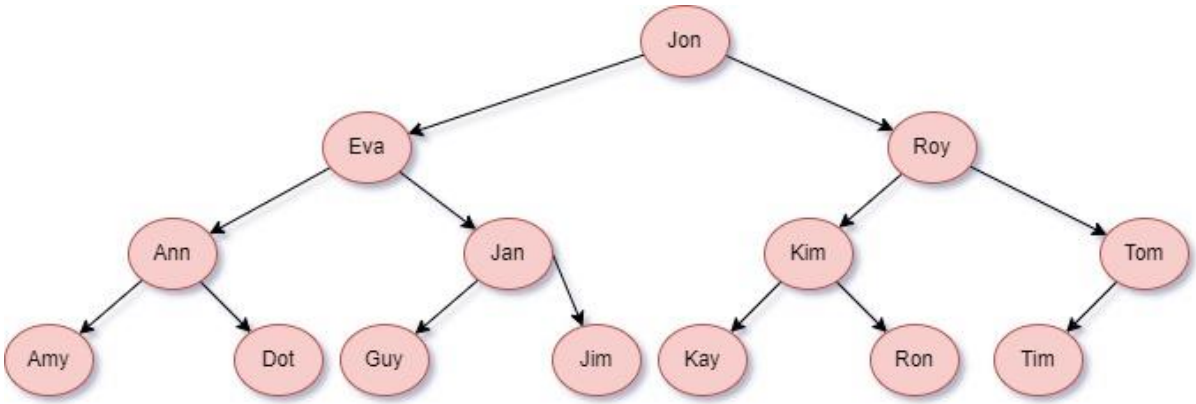
b) Amy Tom Tim Ann Roy Dot Eva Ron Kim Kay Guy Jon Jan Jim



c) Jan Jon Tim Ron Guy Ann Jim Tom Amy Eva Roy Kim Dot Kay



d) Jon Roy Tom Eva Tim Kim Ann Ron Jan Amy Dot Guy Jim Kay



E5. Consider building two binary search trees containing the integer keys 1 to 63,

inclusive, received in the orders.

a). Odd integers first (1, 3, 5, ..., 63), then even integers (2, 4, 6, ...):

The tree will be more balanced initially due to the even distribution of odd numbers.

b). Even integers first (32, 16, 48), then odd integers (1, 3, 5, ..., 63):

This will be quicker to build because the tree is being built in a more balanced manner from the start. The tree built using the second method (b) will likely be quicker to build and more balanced initially because inserting median values first helps maintain balance.

E6. All parts of this exercise refer to the binary search trees shown in Figure 10.8 and concern the different orders in which the keys a, b, . . . , g can be inserted into an initially empty binary search tree.

a) Give four different orders for inserting the keys, each of which will yield the binary search tree shown in part (a).
   d – b – a – c – f – e – g
   d – f – e – g – b – a – c
   d – b – f – e – g – a – c
   d – f – b – a – c – e – g

b) Give four different orders for inserting the keys, each of which will yield the binary search tree shown in part (b).
   e – f – g – b – a – d – c
   e – b – a – d – c – f – g
   e – b – f – a – d – c – g
   e – b – f – g – a – d – g

c) Give four different orders for inserting the keys, each of which will yield the binary search tree shown in part (c).
   a – g – e – b – f – d – c
   a – g – e – f – b – d – c
   a – g – e – b – d – f – c
   a – g – e – b – d – c – f

d) Explain why there is only one order for inserting the keys that will produce a binary search tree that reduces to a given chain, such as the one shown in part (d) or in part (e).

This is because both of these trees have no branches, which would mean the insert order has to be one specific way, or else a branch will occur on the tree which both of these examples wouldn't want.