Algorithm Analysis and Design

Advanced Data Structure

(B Tree)

(Deletion)

Lecture -33-35

Overview

- B-trees are balanced search trees designed to work well on magnetic disks or other direct-access secondary storage devices.
- Time complexity of B Tree in big O notation

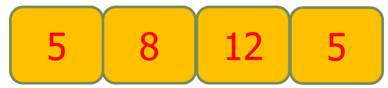
Algorithm	Average	Worst case
Space	O(n)	O(n)
Search	O(log n)	O(log n)
Insert	O(log n)	O(log n)
Delete	O(log n)	O(log n)

B Tree

A B-tree T is a rooted tree (with root root[T]) having the following properties.

1. Every node x has the following fields:

   a. $n[x]$, the number of keys currently stored in node x, For example: $n[x] = 4$



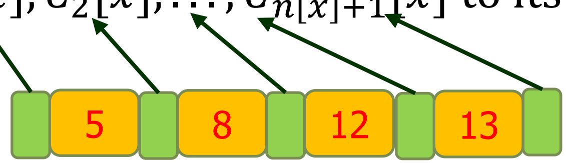
   b. the $n[x]$ keys themselves, stored in nondecreasing order:

   $key_1\ [x] \leq key_2[x] \leq \cdots \leq key_{n[x]}[x]$, and

   c. $leaf\ [x]$, a boolean value that is TRUE if x is a leaf and FALSE if x is an internal node.

2. If x is an internal node, it also contains $n[x]\ +\ 1$ pointers.

   $C_1[x], C_2[x], \ldots, C_{n[x]+1}[x]$ to its children.

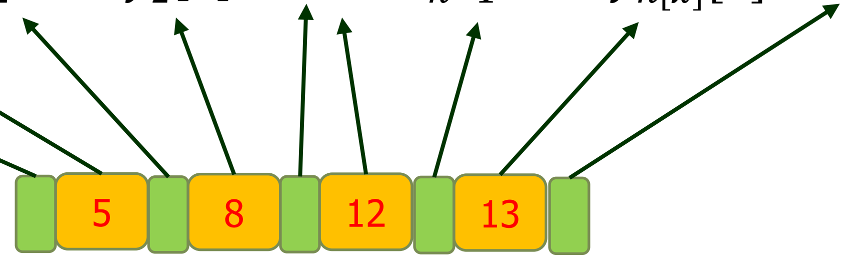


Leaf nodes have no children, so their C_i fields are undefined.

B Tree

3. The keys $key_i[x]$ separate the ranges of keys stored in each subtree: if k_i is any key stored in the subtree with root$C_i[x]$, then

$$k_1 \leq key_1[x] \leq k_2 \leq key_2[x] \leq \cdots \leq k_{n-1} \leq key_{n[x]}[x] \leq k_n$$



4. All leaf nodes has the same depth, which is the tree's height h.

B Tree

5. There are lower and upper bounds on the number of keys a node can contain. These bounds can be expressed in terms of a fixed integer $t \geq 2$ called the minimum degree of the B-tree:

   a. Every node other than the root must have at least t - 1 keys. Every internal node other than the root thus has at least t children.

      If the tree is nonempty, the root must have at least one key.

   b. Every node can contain at most 2t - 1 keys. Therefore, an internal node can have at most 2t children.

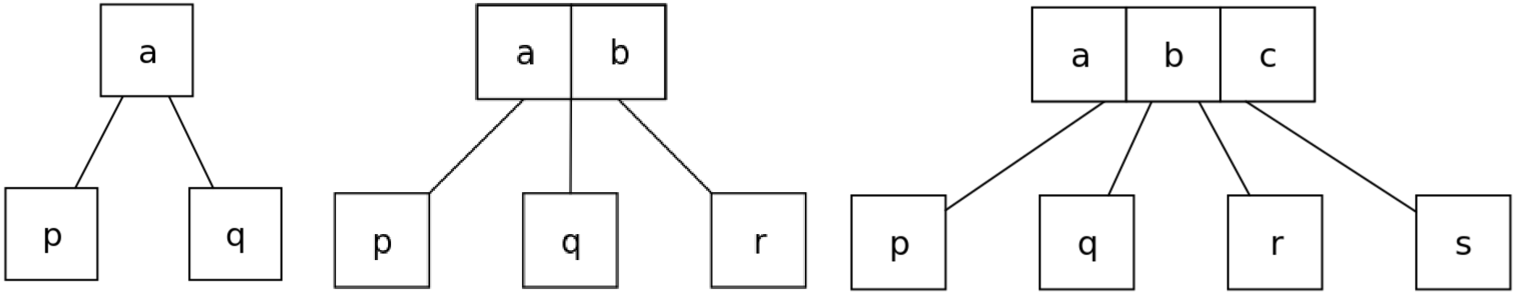
      (We say that a node is full if it contains exactly 2t - 1 keys.)

B Tree

The simplest B-tree occurs when t = 2. Every internal node then has either 2, 3, or 4 children, and we have a 2-3-4 tree.

- a 2-node has one data element, and if internal has two child nodes;
- a 3-node has two data elements, and if internal has three child nodes;
- a 4-node has three data elements, and if internal has four child nodes;

For Example



In practice, however, much larger values of t are typically used.

B Tree (Insertion)

Example 1:

Draw a B-Tree of minimum degree t=3 of the given sequence and assume that B-Tree is initially empty.

<78, 56, 52, 95, 88, 105, 15, 35, 22, 47, 43, 50, 19, 31, 40, 41, 59>

So, The required minimum key =t-1=3-1=2

The required maximum key = 2t-1=6-1=5

B Tree (Insertion)

Example 1:

Draw a B-Tree of minimum degree t=3 of the given sequence and assume that B-Tree is initially empty.

<78, 56, 52, 95, 88, 105, 15, 35, 22, 47, 43, 50, 19, 31, 40, 41, 59>

So, The required minimum key =t-1=3-1=2

The required maximum key = 2t-1=6-1=5

Insert : 78

78

B Tree (Insertion)

Example 1:

Draw a B-Tree of minimum degree t=3 of the given sequence and assume that B-Tree is initially empty.

<78, 56, 52, 95, 88, 105, 15, 35, 22, 47, 43, 50, 19, 31, 40, 41, 59>

So, The required minimum key =t-1=3-1=2

The required maximum key = 2t-1=6-1=5

Insert : 78

78

Insert : 56

56	78

B Tree (Insertion)

Example 1:

Draw a B-Tree of minimum degree t=3 of the given sequence and assume that B-Tree is initially empty.

<78, 56, 52, 95, 88, 105, 15, 35, 22, 47, 43, 50, 19, 31, 40, 41, 59>

Insert : 52

52	56	78

B Tree (Insertion)

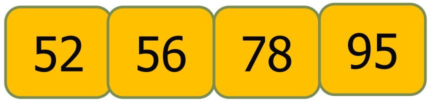
Example 1:

Draw a B-Tree of minimum degree t=3 of the given sequence and assume that B-Tree is initially empty.

<78, 56, 52, 95, 88, 105, 15, 35, 22, 47, 43, 50, 19, 31, 40, 41, 59>

Insert : 52

Insert : 95



B Tree (Insertion)

Example 1:

Draw a B-Tree of minimum degree t=3 of the given sequence and assume that B-Tree is initially empty.

<78, 56, 52, 95, 88, 105, 15, 35, 22, 47, 43, 50, 19, 31, 40, 41, 59>

Insert : 52

52 | 56 | 78

Insert : 95

52 | 56 | 78 | 95

Insert : 88

52 | 56 | 78 | 88 | 95

B Tree (Insertion)

Example 1:

Draw a B-Tree of minimum degree t=3 of the given sequence and assume that B-Tree is initially empty.

<78, 56, 52, 95, 88, 105, 15, 35, 22, 47, 43, 50, 19, 31, 40, 41, 59>

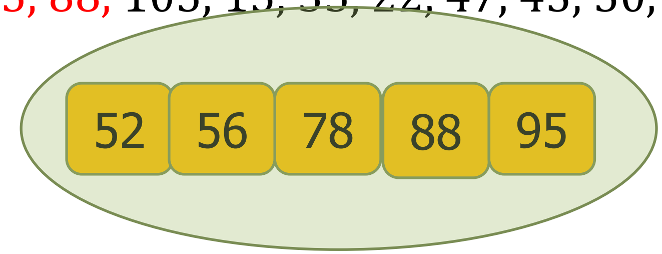
52	56	78	88	95

B Tree (Insertion)

Example 1:

Draw a B-Tree of minimum degree t=3 of the given sequence and assume that B-Tree is initially empty.

<78, 56, 52, 95, 88, 105, 15, 35, 22, 47, 43, 50, 19, 31, 40, 41, 59>



Now we can't insert a new key into an existing leaf node as the maximum key size limit is achieved.

Hence we introduced a split function , which split the tree by it's median key y.

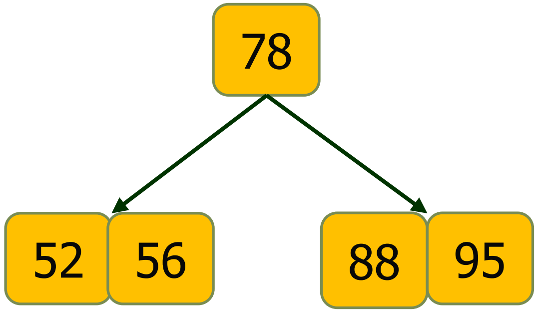
B Tree (Insertion)

Example 1:

Draw a B-Tree of minimum degree t=3 of the given sequence and assume that B-Tree is initially empty.

<78, 56, 52, 95, 88, 105, 15, 35, 22, 47, 43, 50, 19, 31, 40, 41, 59>

First split and then Insert : 105



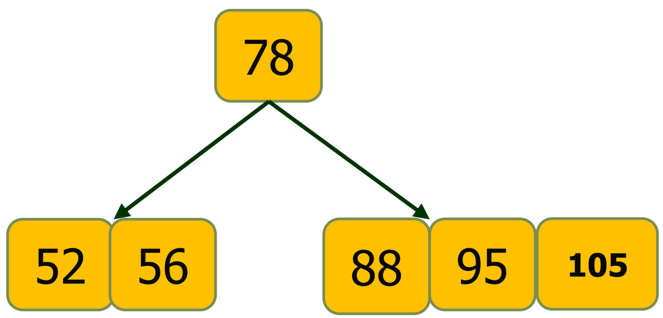
B Tree (Insertion)

Example 1:

Draw a B-Tree of minimum degree t=3 of the given sequence and assume that B-Tree is initially empty.

<78, 56, 52, 95, 88, 105, 15, 35, 22, 47, 43, 50, 19, 31, 40, 41, 59>

Insert : 105



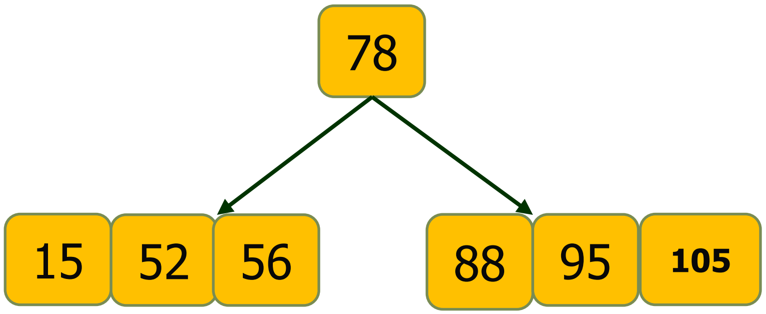
B Tree (Insertion)

Example 1:

Draw a B-Tree of minimum degree t=3 of the given sequence and assume that B-Tree is initially empty.

<78, 56, 52, 95, 88, 105, 15, 35, 22, 47, 43, 50, 19, 31, 40, 41, 59>

Insert : 15



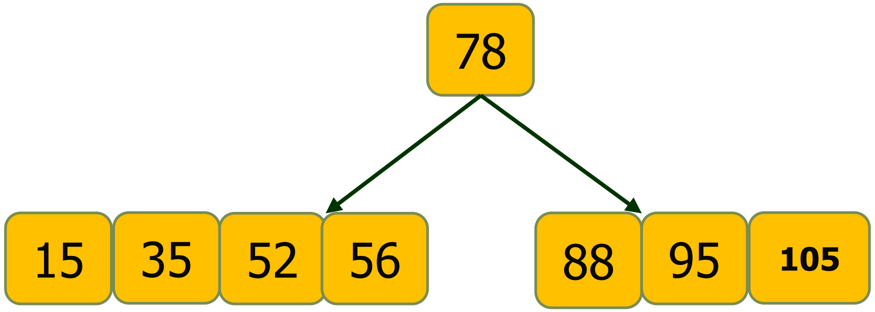
B Tree (Insertion)

Example 1:

Draw a B-Tree of minimum degree t=3 of the given sequence and assume that B-Tree is initially empty.

<78, 56, 52, 95, 88, 105, 15, 35, 22, 47, 43, 50, 19, 31, 40, 41, 59>

Insert : 35



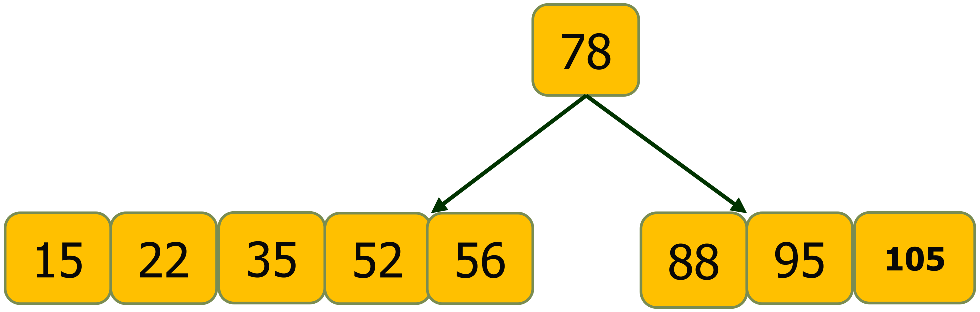
B Tree (Insertion)

Example 1:

Draw a B-Tree of minimum degree t=3 of the given sequence and assume that B-Tree is initially empty.

<78, 56, 52, 95, 88, 105, 15, 35, 22, 47, 43, 50, 19, 31, 40, 41, 59>

Insert : 22



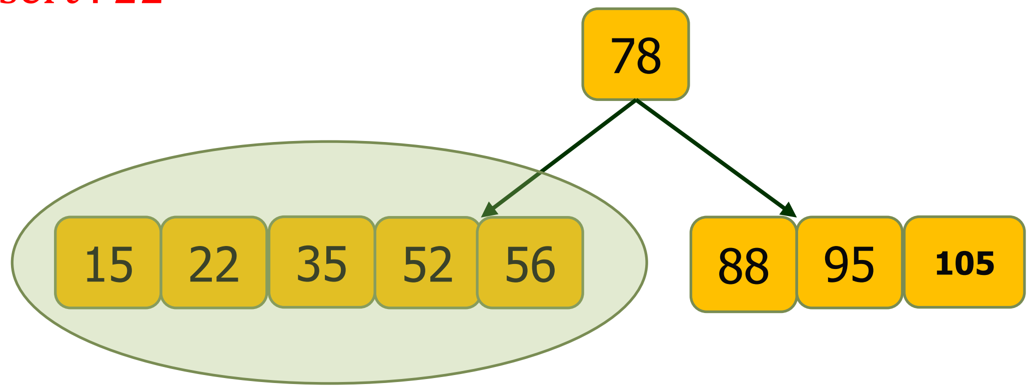
B Tree (Insertion)

Example 1:

Draw a B-Tree of minimum degree t=3 of the given sequence and assume that B-Tree is initially empty.

<78, 56, 52, 95, 88, 105, 15, 35, 22, 47, 43, 50, 19, 31, 40, 41, 59>

Insert : 22



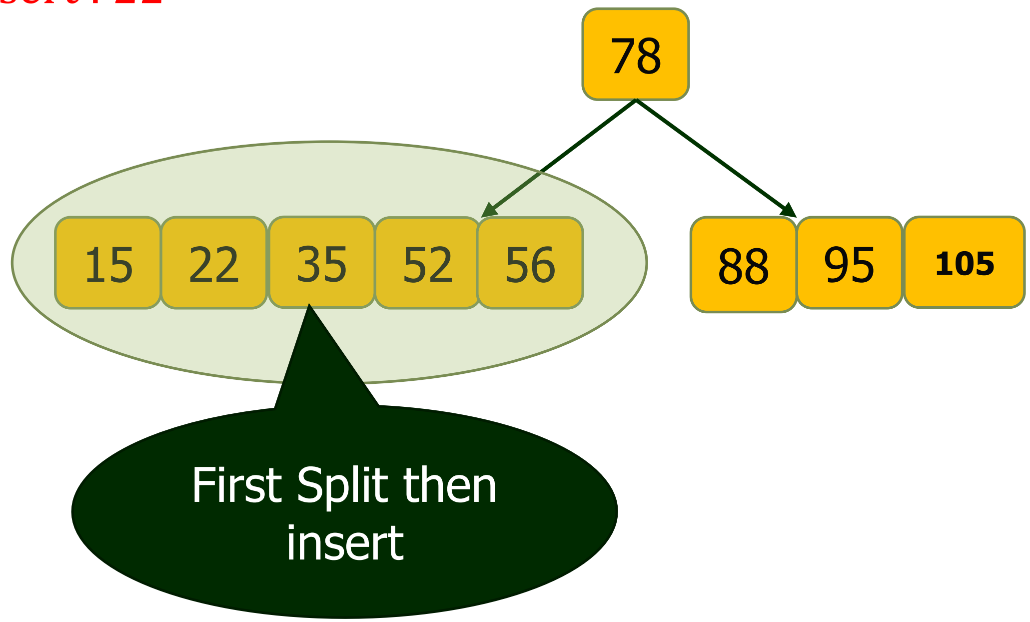
B Tree (Insertion)

Example 1:

Draw a B-Tree of minimum degree t=3 of the given sequence and assume that B-Tree is initially empty.

<78, 56, 52, 95, 88, 105, 15, 35, 22, 47, 43, 50, 19, 31, 40, 41, 59>

Insert : 22



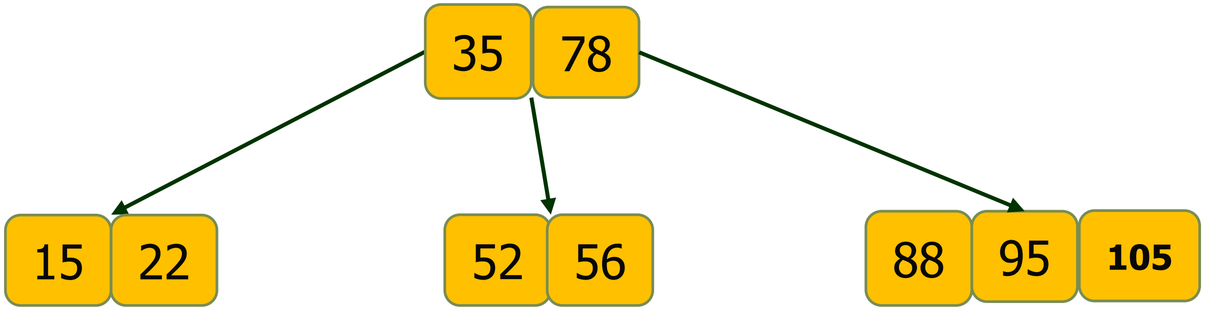
B Tree (Insertion)

Example 1:

Draw a B-Tree of minimum degree t=3 of the given sequence and assume that B-Tree is initially empty.

<78, 56, 52, 95, 88, 105, 15, 35, 22, 47, 43, 50, 19, 31, 40, 41, 59>

First split and then Insert :47



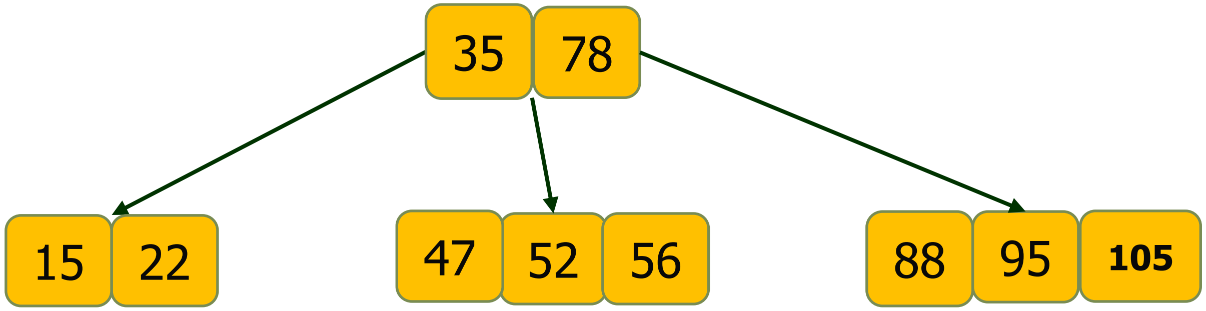
B Tree (Insertion)

Example 1:

Draw a B-Tree of minimum degree t=3 of the given sequence and assume that B-Tree is initially empty.

<78, 56, 52, 95, 88, 105, 15, 35, 22, 47, 43, 50, 19, 31, 40, 41, 59>

First split and then Insert : 47



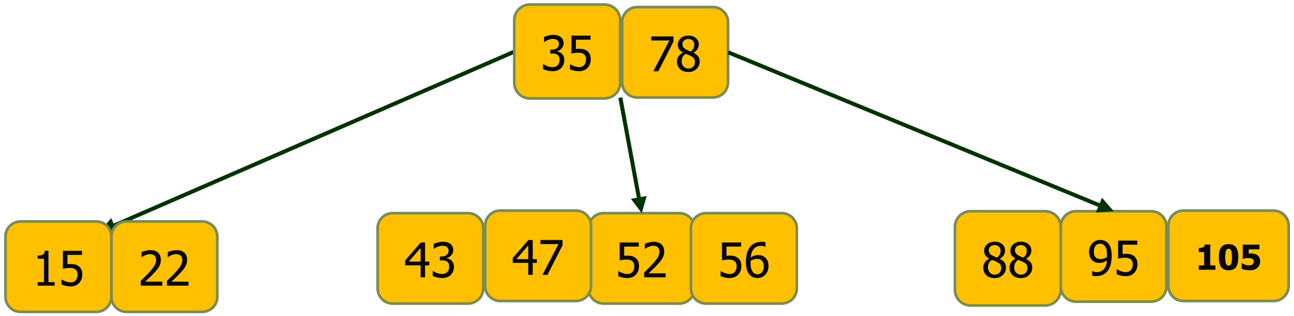
B Tree (Insertion)

Example 1:

Draw a B-Tree of minimum degree t=3 of the given sequence and assume that B-Tree is initially empty.

<78, 56, 52, 95, 88, 105, 15, 35, 22, 47, 43, 50, 19, 31, 40, 41, 59>

Insert : 43



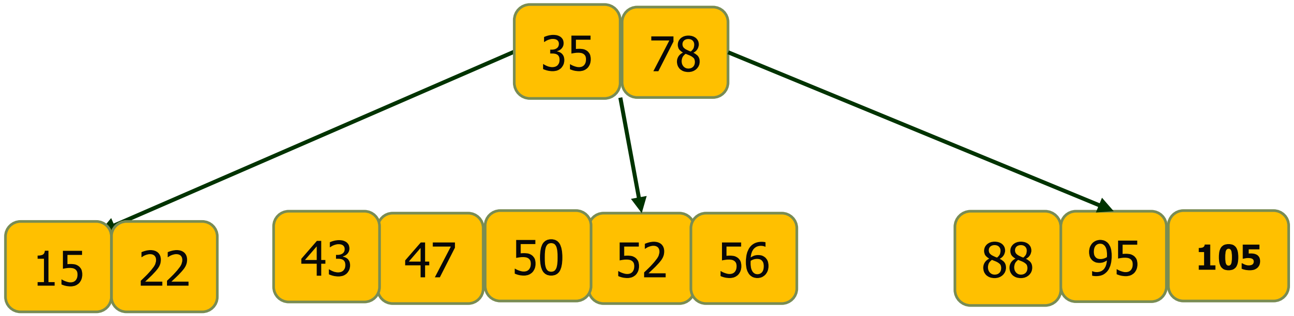
B Tree (Insertion)

Example 1:

Draw a B-Tree of minimum degree t=3 of the given sequence and assume that B-Tree is initially empty.

<78, 56, 52, 95, 88, 105, 15, 35, 22, 47, 43, 50, 19, 31, 40, 41, 59>

Insert : 50



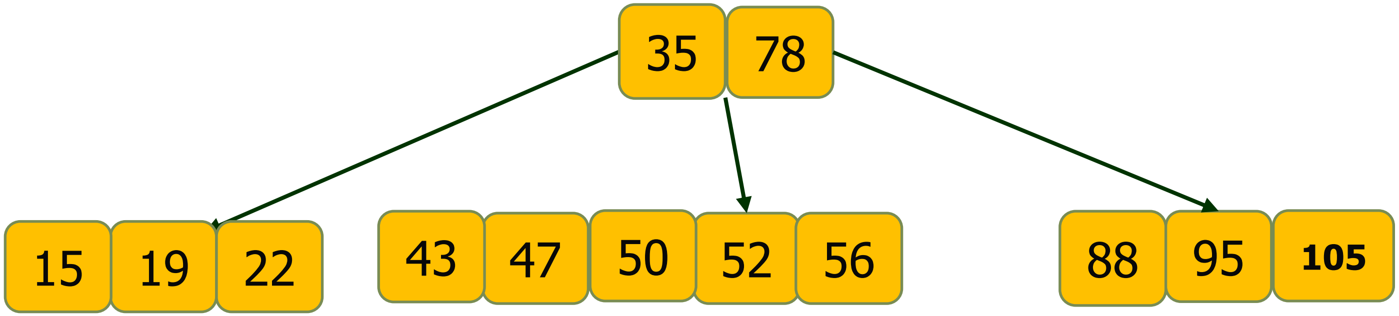
B Tree (Insertion)

Example 1:

Draw a B-Tree of minimum degree t=3 of the given sequence and assume that B-Tree is initially empty.

<78, 56, 52, 95, 88, 105, 15, 35, 22, 47, 43, 50, 19, 31, 40, 41, 59>

Insert : 19



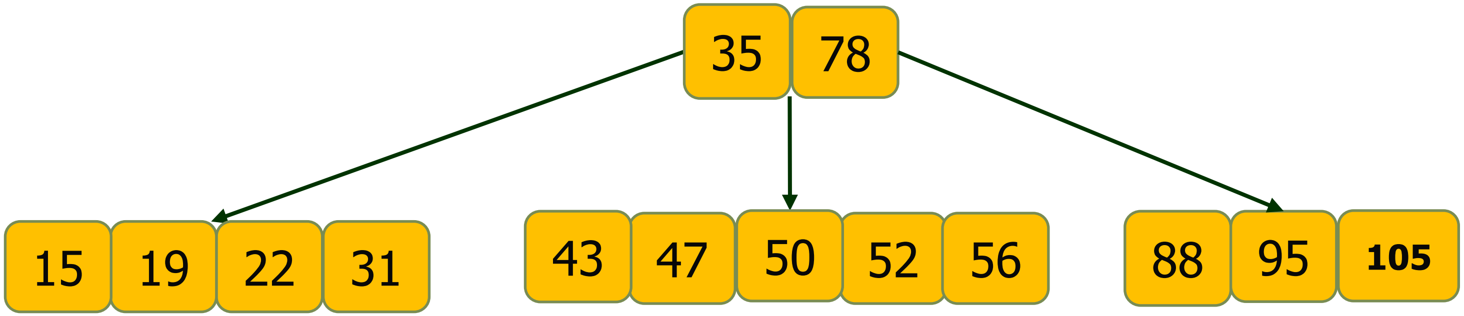
B Tree (Insertion)

Example 1:

Draw a B-Tree of minimum degree t=3 of the given sequence and assume that B-Tree is initially empty.

<78, 56, 52, 95, 88, 105, 15, 35, 22, 47, 43, 50, 19, 31, 40, 41, 59>

Insert : 31



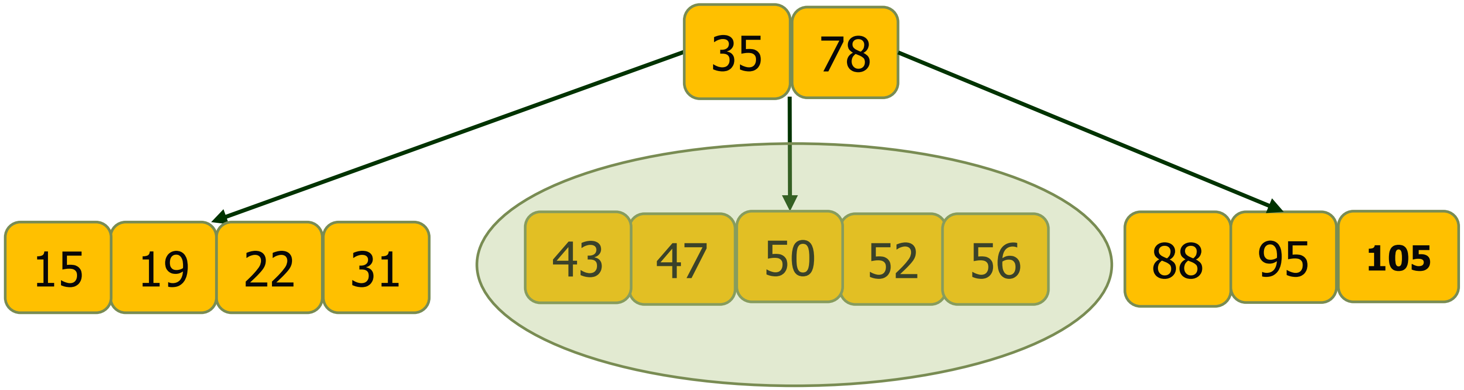
B Tree (Insertion)

Example 1:

Draw a B-Tree of minimum degree t=3 of the given sequence and assume that B-Tree is initially empty.

<78, 56, 52, 95, 88, 105, 15, 35, 22, 47, 43, 50, 19, 31, 40, 41, 59>

Insert : 40



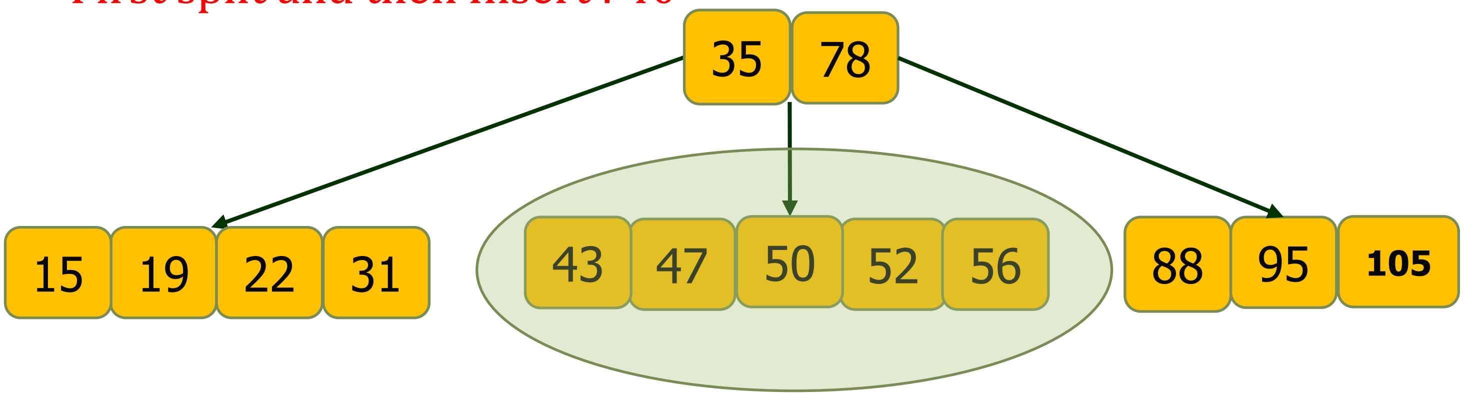
B Tree (Insertion)

Example 1:

Draw a B-Tree of minimum degree t=3 of the given sequence and assume that B-Tree is initially empty.

<78, 56, 52, 95, 88, 105, 15, 35, 22, 47, 43, 50, 19, 31, 40, 41, 59>

First split and then Insert : 40



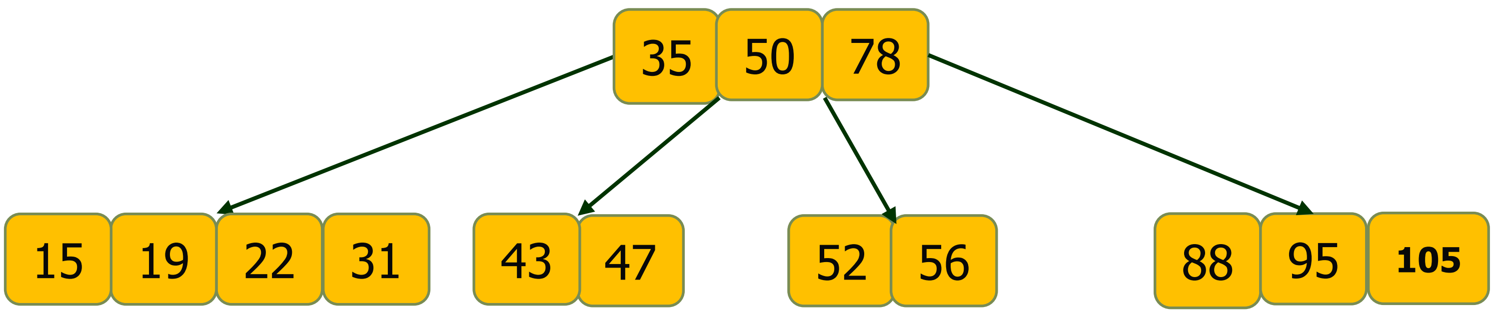
B Tree (Insertion)

Example 1:

Draw a B-Tree of minimum degree t=3 of the given sequence and assume that B-Tree is initially empty.

<78, 56, 52, 95, 88, 105, 15, 35, 22, 47, 43, 50, 19, 31, 40, 41, 59>

First split and then Insert : 40



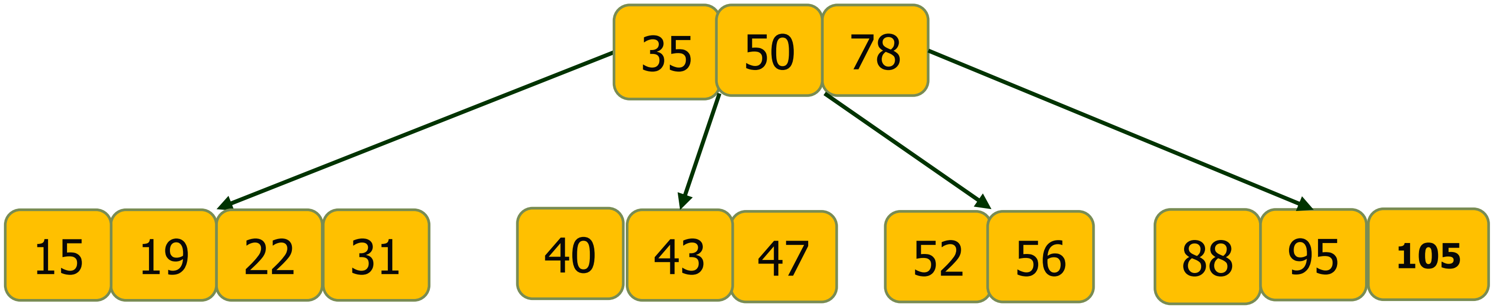
B Tree (Insertion)

Example 1:

Draw a B-Tree of minimum degree t=3 of the given sequence and assume that B-Tree is initially empty.

<78, 56, 52, 95, 88, 105, 15, 35, 22, 47, 43, 50, 19, 31, 40, 41, 59>

First split and then Insert : 40



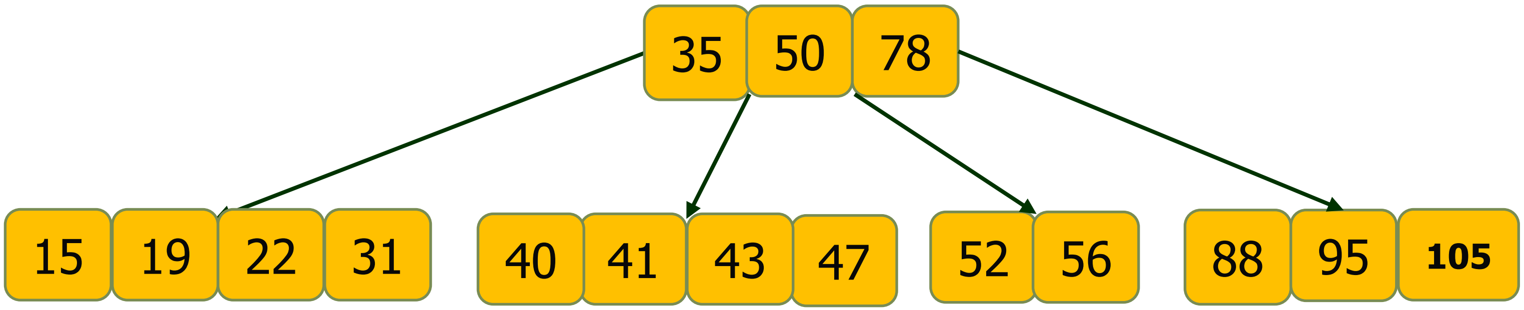
B Tree (Insertion)

Example 1:

Draw a B-Tree of minimum degree t=3 of the given sequence and assume that B-Tree is initially empty.

<78, 56, 52, 95, 88, 105, 15, 35, 22, 47, 43, 50, 19, 31, 40, 41, 59>

Insert : 41



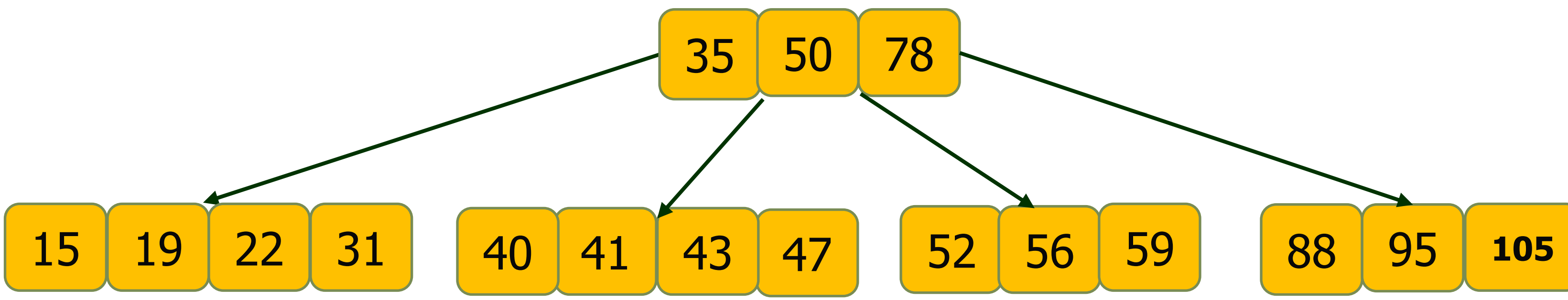
B Tree (Insertion)

Example 1:

Draw a B-Tree of minimum degree t=3 of the given sequence and assume that B-Tree is initially empty.

<78, 56, 52, 95, 88, 105, 15, 35, 22, 47, 43, 50, 19, 31, 40, 41, 59>

Insert : 59



B Tree (Insertion)

Example 2:

Construct a B-Tree of degree t=3 on following data set and assume that B-Tree is initially empty.

<E, A, S, Y, Q, U, E, S, T, I, O, N, I, N, C>

B Tree (Insertion)

Example 2:

Construct a B-Tree of degree t=3 on following data set and assume that B-Tree is initially empty.

<E, A, S, Y, Q, U, E, S, T, I, O, N, I, N, C>

Insert : E

B Tree (Insertion)

Example 2:

Construct a B-Tree of degree t=3 on following data set and assume that B-Tree is initially empty.

<E, A, S, Y, Q, U, E, S, T, I, O, N, I, N, C>

Insert : E

Insert : A

B Tree (Insertion)

Example 2:

Construct a B-Tree of degree t=3 on following data set and assume that B-Tree is initially empty.

<E, A, S, Y, Q, U, E, S, T, I, O, N, I, N, C>

Insert : E

Insert : A

Insert : S

B Tree (Insertion)

Example 2:

Construct a B-Tree of degree t=3 on following data set and assume that B-Tree is initially empty.

<E, A, S, Y, Q, U, E, S, T, I, O, N, I, N, C>

Insert : E

Insert : A

Insert : S

Insert : Y

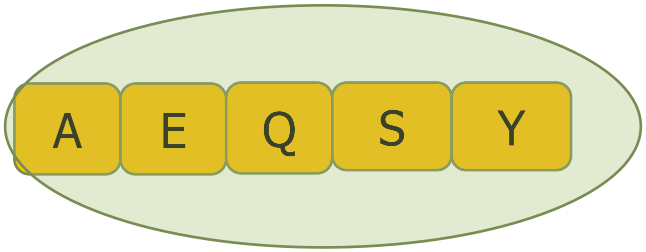
B Tree (Insertion)

Example 2:

Construct a B-Tree of degree t=3 on following data set and assume that B-Tree is initially empty.

<E, A, S, Y, Q, U, E, S, T, I, O, N, I, N, C>

Insert : Q



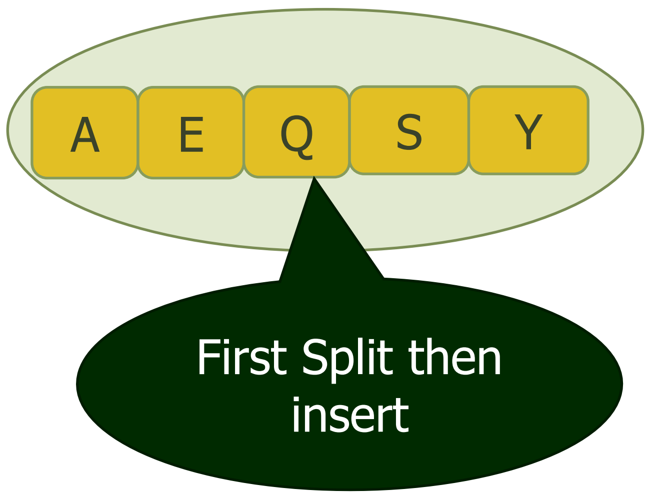
B Tree (Insertion)

Example 2:

Construct a B-Tree of degree t=3 on following data set and assume that B-Tree is initially empty.

<E, A, S, Y, Q, U, E, S, T, I, O, N, I, N, C>

Insert : Q



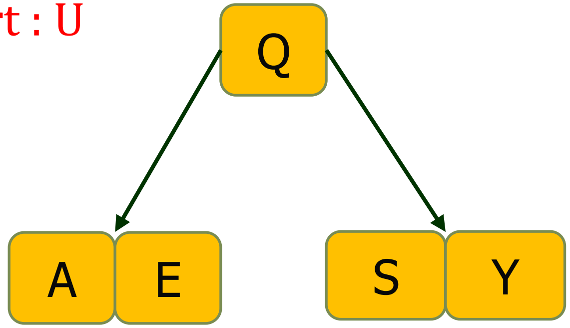
B Tree (Insertion)

Example 2:

Construct a B-Tree of degree t=3 on following data set and assume that B-Tree is initially empty.

<E, A, S, Y, Q, U, E, S, T, I, O, N, I, N, C>

First split and then Insert : U



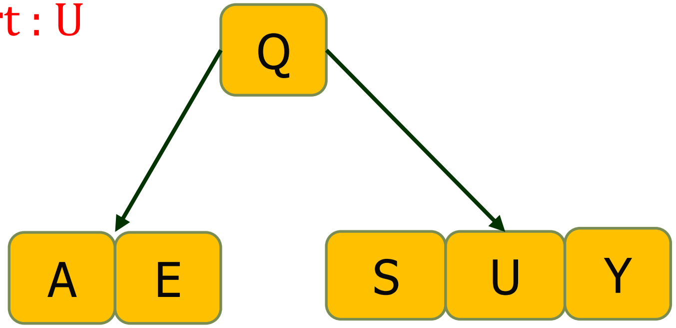
B Tree (Insertion)

Example 2:

Construct a B-Tree of degree t=3 on following data set and assume that B-Tree is initially empty.

<E, A, S, Y, Q, U, E, S, T, I, O, N, I, N, C>

First split and then Insert : U



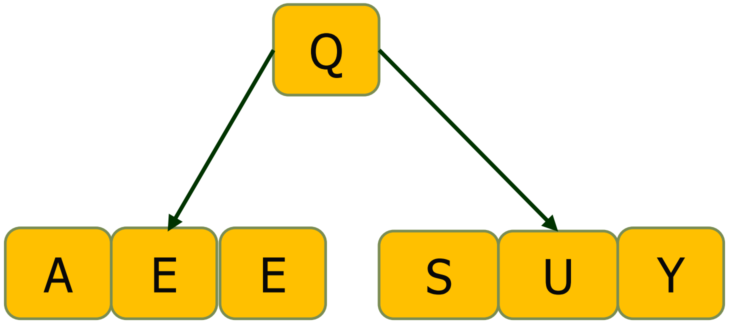
B Tree (Insertion)

Example 2:

Construct a B-Tree of degree t=3 on following data set and assume that B-Tree is initially empty.

<E, A, S, Y, Q, U, E, S, T, I, O, N, I, N, C>

Insert : E



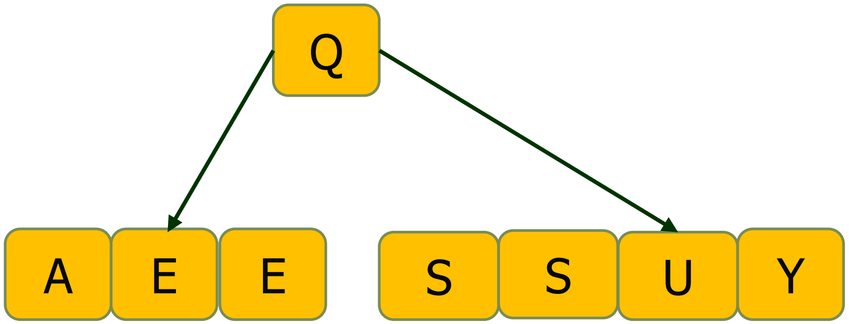
B Tree (Insertion)

Example 2:

Construct a B-Tree of degree t=3 on following data set and assume that B-Tree is initially empty.

<E, A, S, Y, Q, U, E, S, T, I, O, N, I, N, C>

Insert : S



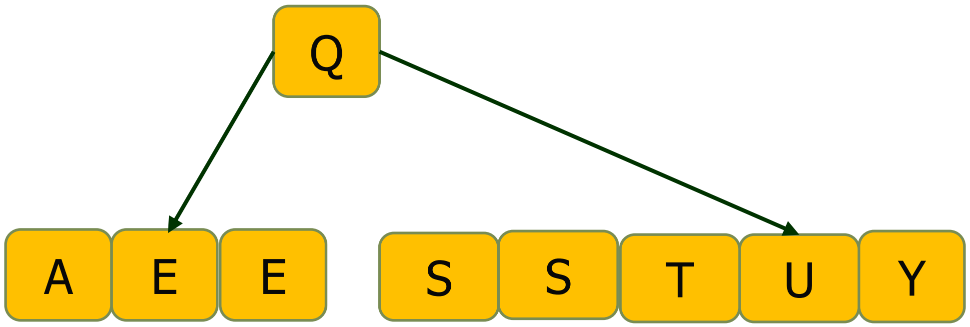
B Tree (Insertion)

Example 2:

Construct a B-Tree of degree t=3 on following data set and assume that B-Tree is initially empty.

<E, A, S, Y, Q, U, E, S, T, I, O, N, I, N, C>

Insert : T



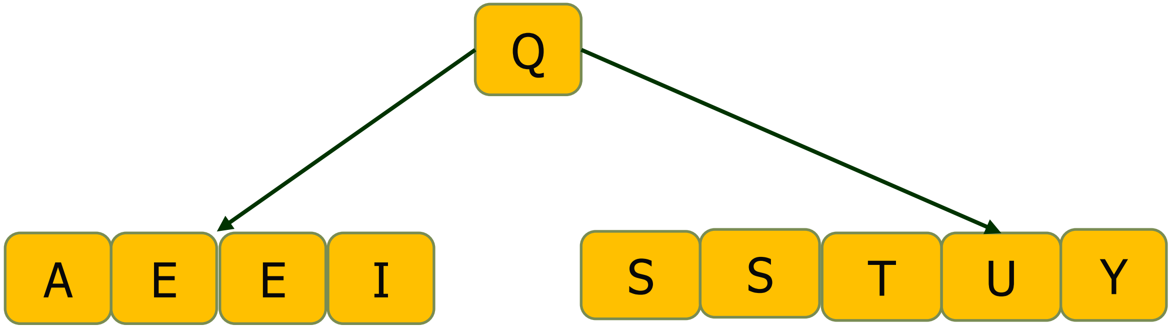
B Tree (Insertion)

Example 2:

Construct a B-Tree of degree t=3 on following data set and assume that B-Tree is initially empty.

<E, A, S, Y, Q, U, E, S, T, I, O, N, I, N, C>

Insert : I



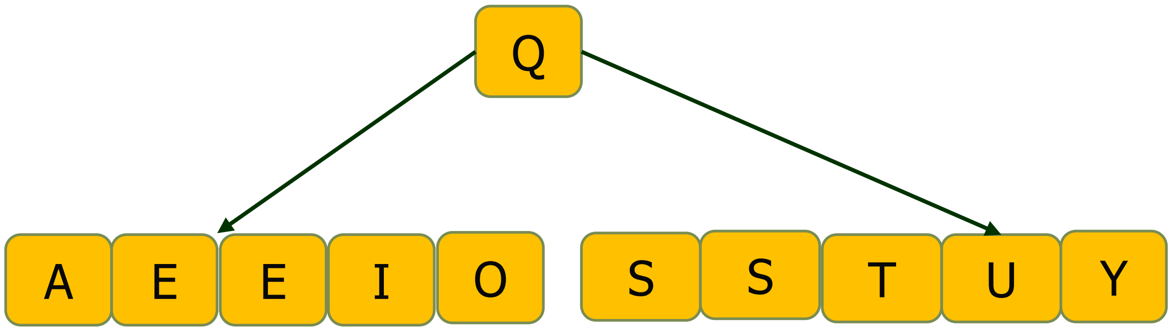
B Tree (Insertion)

Example 2:

Construct a B-Tree of degree t=3 on following data set and assume that B-Tree is initially empty.

<E, A, S, Y, Q, U, E, S, T, I, O, N, I, N, C>

Insert : O



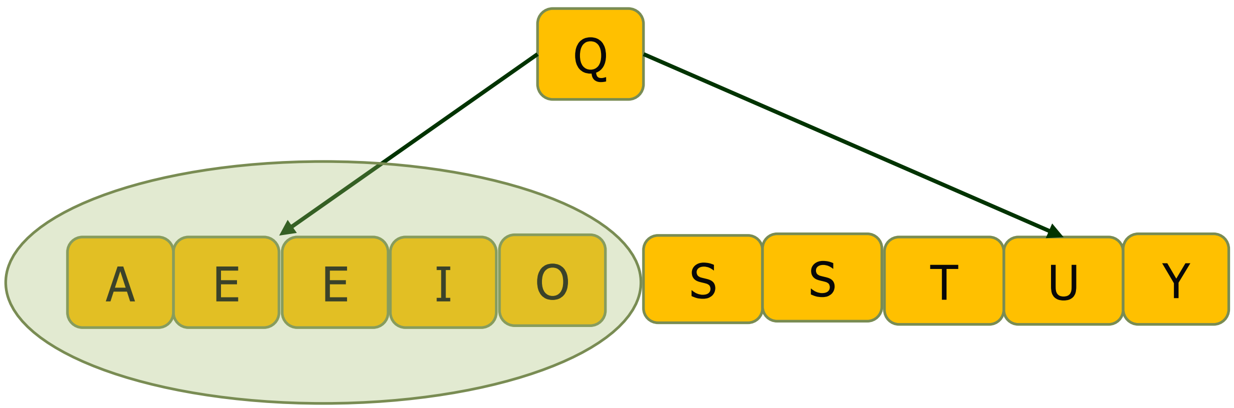
B Tree (Insertion)

Example 2:

Construct a B-Tree of degree t=3 on following data set and assume that B-Tree is initially empty.

<E, A, S, Y, Q, U, E, S, T, I, O, N, I, N, C>

Insert : N



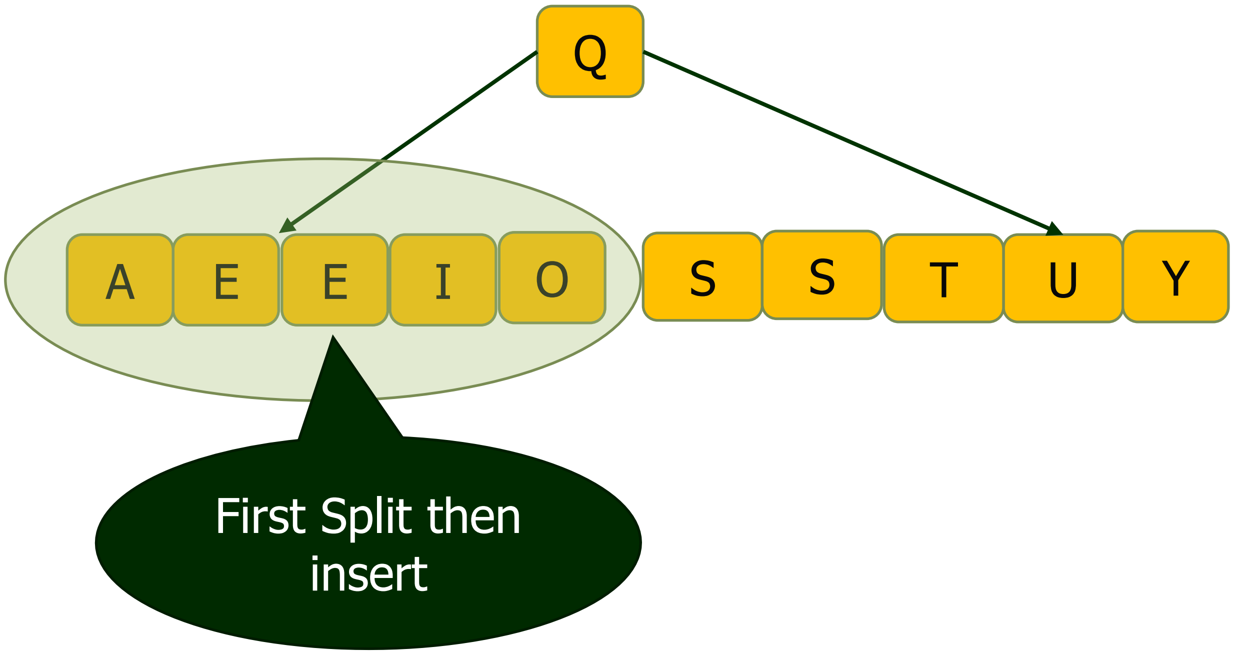
B Tree (Insertion)

Example 2:

Construct a B-Tree of degree t=3 on following data set and assume that B-Tree is initially empty.

<E, A, S, Y, Q, U, E, S, T, I, O, N, I, N, C>

Insert : N



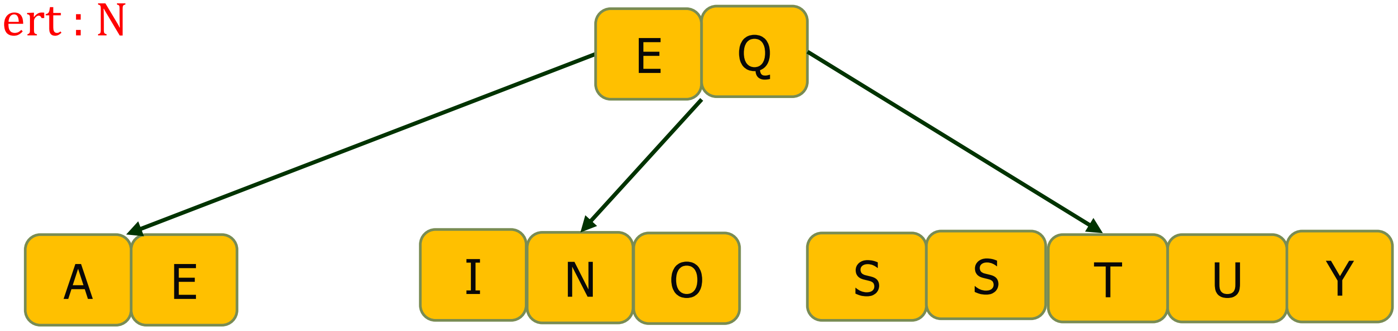
B Tree (Insertion)

Example 2:

Construct a B-Tree of degree t=3 on following data set and assume that B-Tree is initially empty.

<E, A, S, Y, Q, U, E, S, T, I, O, N, I, N, C>

Insert : N



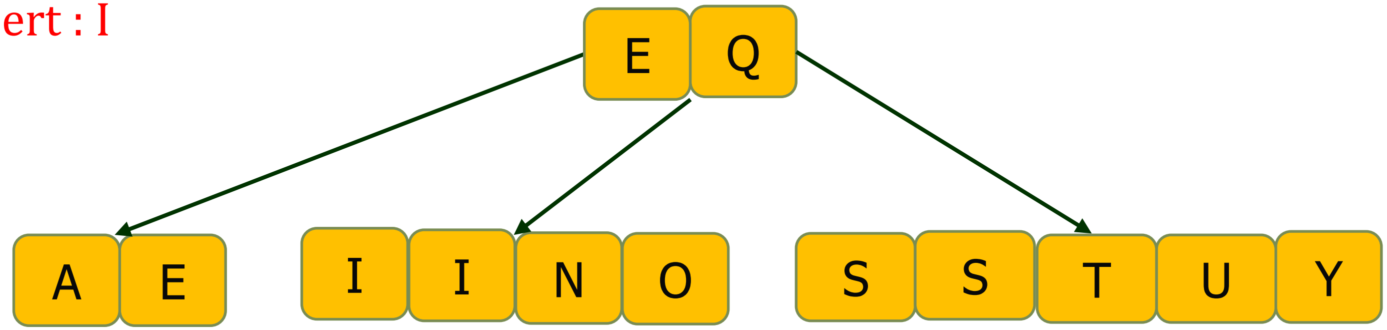
B Tree (Insertion)

Example 2:

Construct a B-Tree of degree t=3 on following data set and assume that B-Tree is initially empty.

<E, A, S, Y, Q, U, E, S, T, I, O, N, I, N, C>

Insert : I



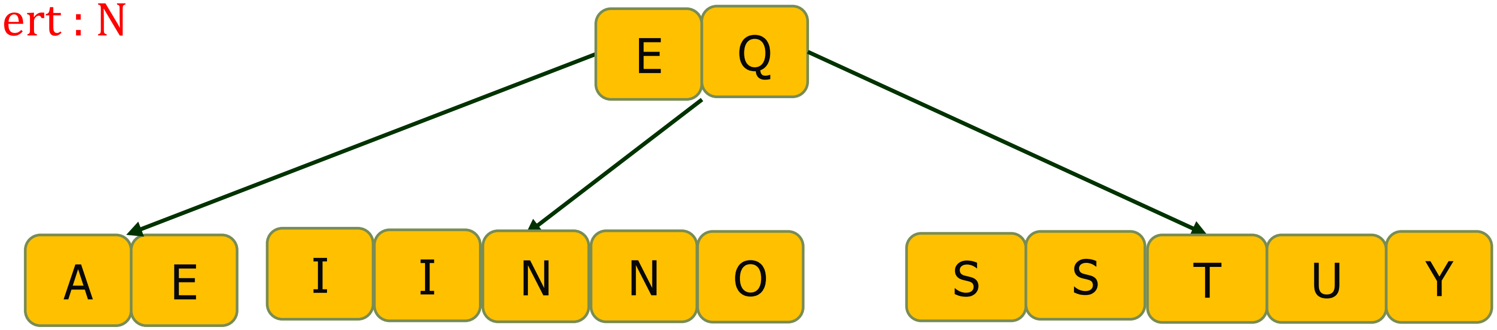
B Tree (Insertion)

Example 2:

Construct a B-Tree of degree t=3 on following data set and assume that B-Tree is initially empty.

<E, A, S, Y, Q, U, E, S, T, I, O, N, I, N, C>

Insert : N



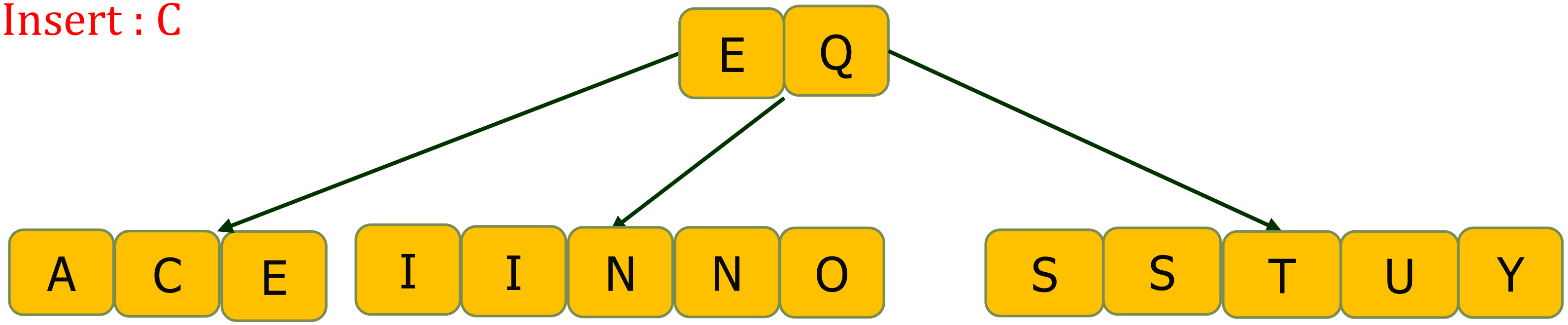
B Tree (Insertion)

Example 2:

Construct a B-Tree of degree t=3 on following data set and assume that B-Tree is initially empty.

<E, A, S, Y, Q, U, E, S, T, I, O, N, I, N, C>

Insert : C



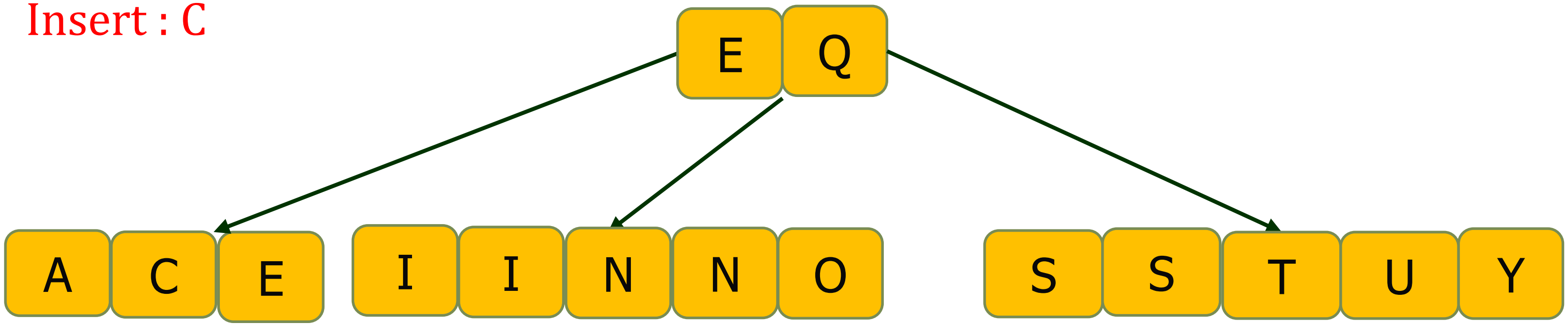
B Tree (Insertion)

Example 2:

Construct a B-Tree of degree t=3 on following data set and assume that B-Tree is initially empty.

<E, A, S, Y, Q, U, E, S, T, I, O, N, I, N, C>

Insert : C



B Tree (Insertion)

A B-Tree can be constructed by order as well as degree.

The question is , how to find Maximum and Minimum key in both the case

<u>Order(m)</u>

Maximum Key=m-1

Minimum Key= $\left\lceil \frac{m}{2} \right\rceil - 1$

<u>Degree(t)</u>

Maximum Key=2t-1

Minimum Key= t-1

B Tree (Insertion)

Example 3:

Construct a B-Tree of order 5 on following data set and assume that B-Tree is initially empty.

<F, S, Q, K, C, L, H, T, V, W, M, R, N, P, A, B, X, Y, Z, D, E>

B Tree (Insertion)

Example 3:

Construct a B-Tree of order 5 on following data set and assume that B-Tree is initially empty.

<F, S, Q, K, C, L, H, T, V, W, M, R, N, P, A, B, X, Y, Z, D, E>

Soln.

Order(m)=5

Maximum Key=m-1 = 5-1=4

Minimum Key=$\left\lceil \frac{m}{2} \right\rceil - 1$= 3-1=2

B Tree (Insertion)

Example 3:

Construct a B-Tree of order 5 on following data set and assume that B-Tree is initially empty.

<F, S, Q, K, C, L, H, T, V, W, M, R, N, P, A, B, X, Y, Z, D, E>

Insert: F

B Tree (Insertion)

Example 3:

Construct a B-Tree of order 5 on following data set and assume that B-Tree is initially empty.

<F, S, Q, K, C, L, H, T, V, W, M, R, N, P, A, B, X, Y, Z, D, E>

Insert: F

Insert: S

B Tree (Insertion)

Example 3:

Construct a B-Tree of order 5 on following data set and assume that B-Tree is initially empty.

<F, S, Q, K, C, L, H, T, V, W, M, R, N, P, A, B, X, Y, Z, D, E>

Insert: F

Insert: S

Insert: Q

B Tree (Insertion)

Example 3:

Construct a B-Tree of order 5 on following data set and assume that B-Tree is initially empty.

<F, S, Q, K, C, L, H, T, V, W, M, R, N, P, A, B, X, Y, Z, D, E>

Insert: F

F

Insert: S

F	S

Insert: Q

F	Q	S

Insert: K

F	K	Q	S

B Tree (Insertion)

Example 3:

Construct a B-Tree of order 5 on following data set and assume that B-Tree is initially empty.

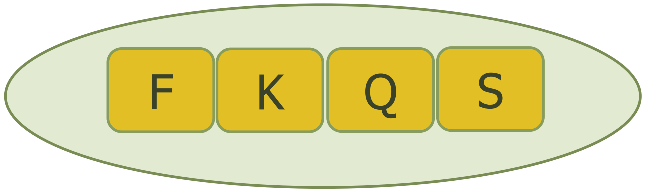
<F, S, Q, K, C, L, H, T, V, W, M, R, N, P, A, B, X, Y, Z, D, E>

Insert: F

Insert: S

Insert: Q

Insert: K



B Tree (Insertion)

Example 3:

Construct a B-Tree of order 5 on following data set and assume that B-Tree is initially empty.

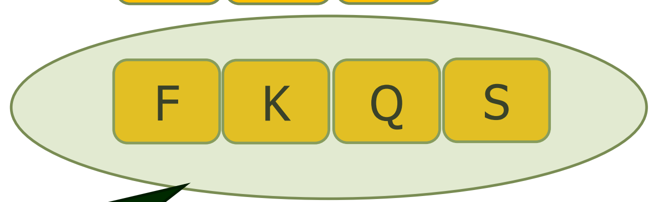
<F, S, Q, K, C, L, H, T, V, W, M, R, N, P, A, B, X, Y, Z, D, E>

Insert: F

Insert: S

Insert: Q

Insert: K



First Split then insert

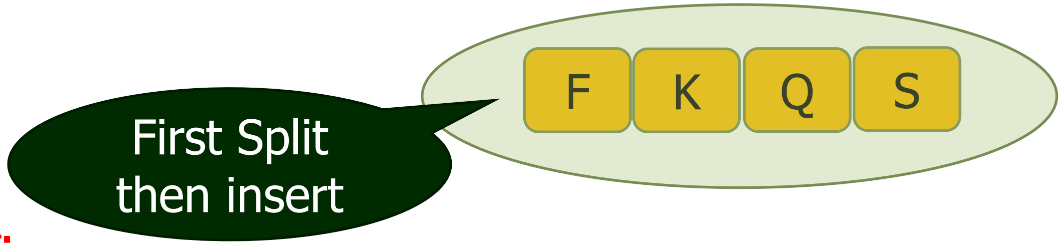
B Tree (Insertion)

Example 3:

Construct a B-Tree of order 5 on following data set and assume that B-Tree is initially empty.

<F, S, Q, K, C, L, H, T, V, W, M, R, N, P, A, B, X, Y, Z, D, E>

Insert: K



Rules for Splitting:

When the key elements are even and there is a need of split, at that time we get two median(i.e. m and m+1), So on these cases following rules are helping for splitting.

Rule 1: If the inserted item is < m then split from ‘m’.

Rule 2: If the inserted item is >m+1 then split from ‘m+1’.

Rule 3: If the inserted item is > m and <m+1 then split on inserted item.

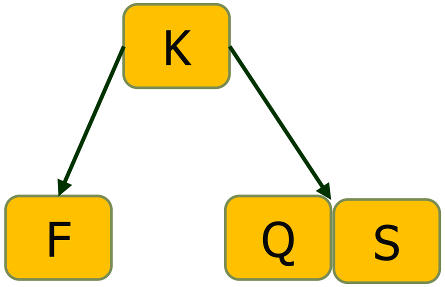
B Tree (Insertion)

Example 3:

Construct a B-Tree of order 5 on following data set and assume that B-Tree is initially empty.

<F, S, Q, K, C, L, H, T, V, W, M, R, N, P, A, B, X, Y, Z, D, E>

Insert: K



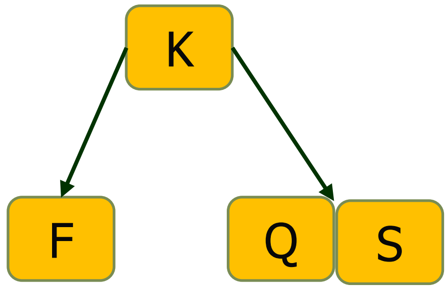
B Tree (Insertion)

Example 3:

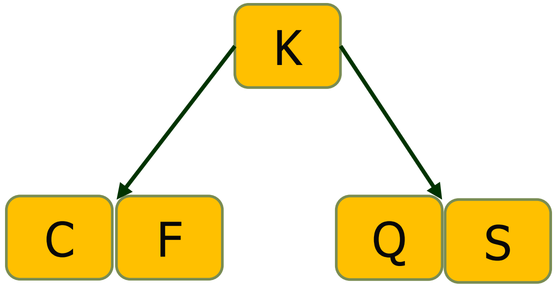
Construct a B-Tree of order 5 on following data set and assume that B-Tree is initially empty.

<F, S, Q, K, C, L, H, T, V, W, M, R, N, P, A, B, X, Y, Z, D, E>

Insert: K



Insert: C



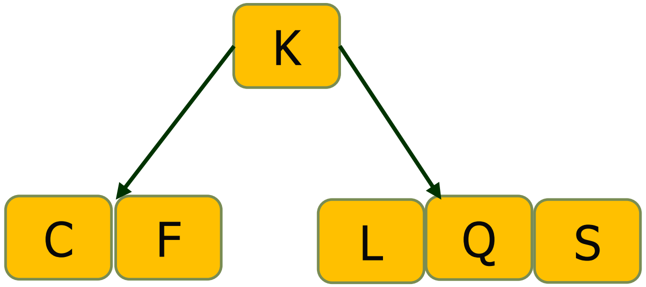
B Tree (Insertion)

Example 3:

Construct a B-Tree of order 5 on following data set and assume that B-Tree is initially empty.

<F, S, Q, K, C, L, H, T, V, W, M, R, N, P, A, B, X, Y, Z, D, E>

Insert: L



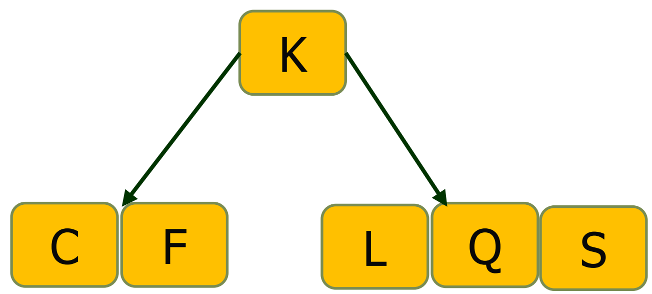
B Tree (Insertion)

Example 3:

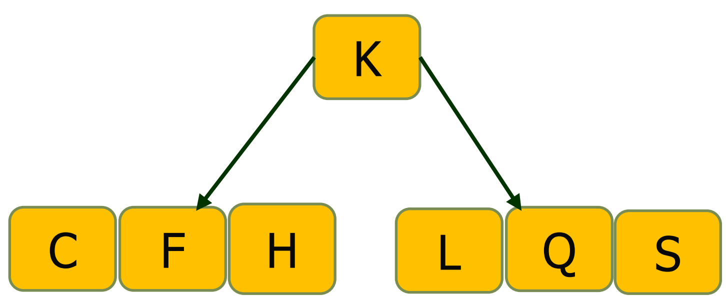
Construct a B-Tree of order 5 on following data set and assume that B-Tree is initially empty.

<F, S, Q, K, C, L, H, T, V, W, M, R, N, P, A, B, X, Y, Z, D, E>

Insert: L



Insert: H



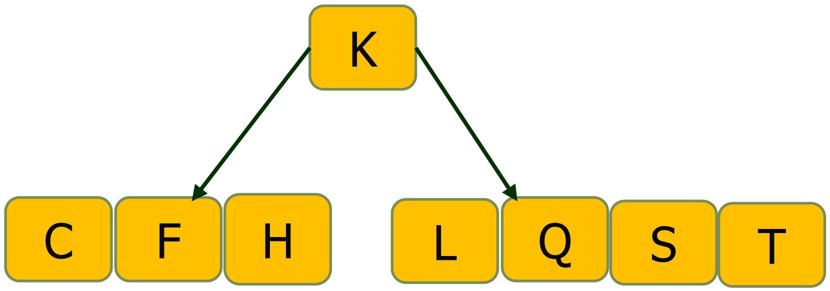
B Tree (Insertion)

Example 3:

Construct a B-Tree of order 5 on following data set and assume that B-Tree is initially empty.

<F, S, Q, K, C, L, H, T, V, W, M, R, N, P, A, B, X, Y, Z, D, E>

Insert: T



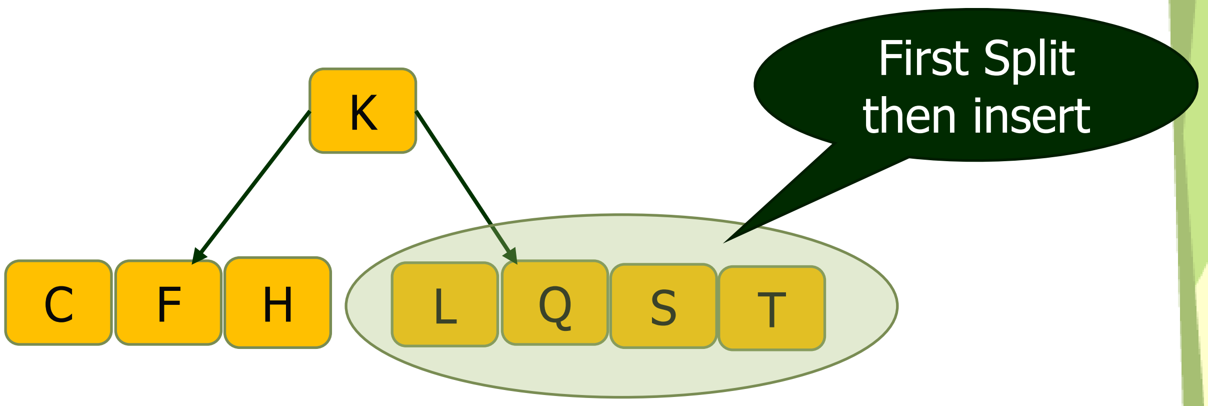
B Tree (Insertion)

Example 3:

Construct a B-Tree of order 5 on following data set and assume that B-Tree is initially empty.

<F, S, Q, K, C, L, H, T, V, W, M, R, N, P, A, B, X, Y, Z, D, E>

Insert: T



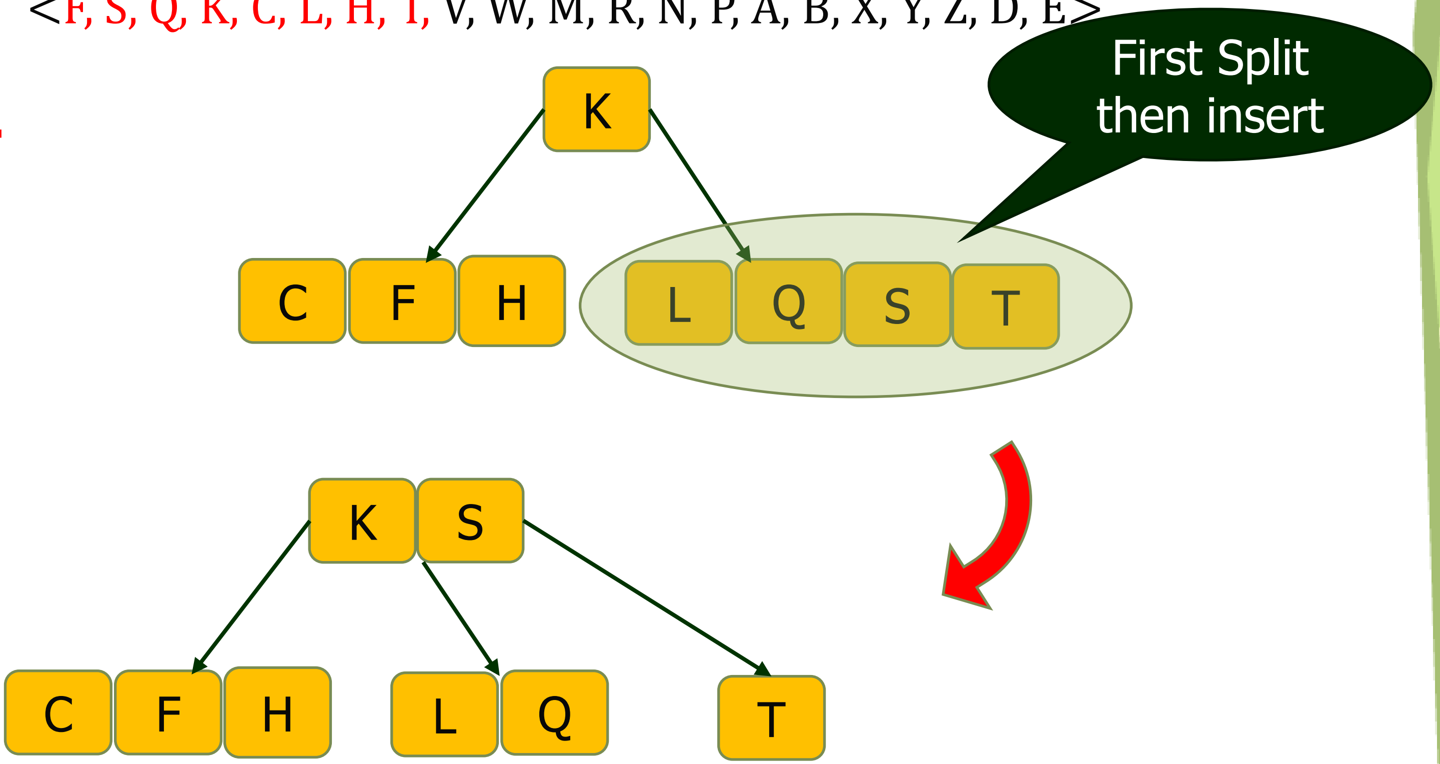
B Tree (Insertion)

Example 3:

Construct a B-Tree of order 5 on following data set and assume that B-Tree is initially empty.

<F, S, Q, K, C, L, H, T, V, W, M, R, N, P, A, B, X, Y, Z, D, E>

Insert: T



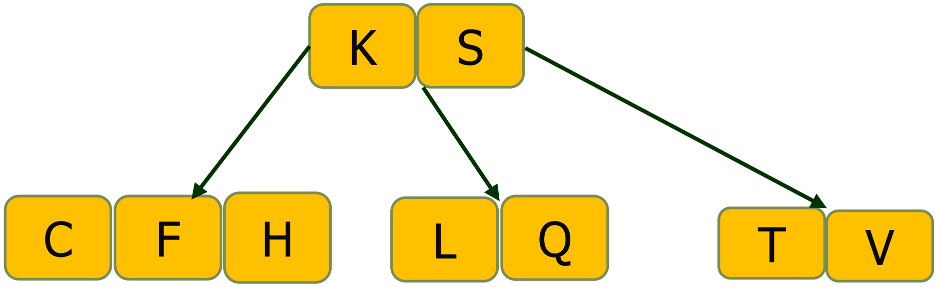
B Tree (Insertion)

Example 3:

Construct a B-Tree of order 5 on following data set and assume that B-Tree is initially empty.

<F, S, Q, K, C, L, H, T, V, W, M, R, N, P, A, B, X, Y, Z, D, E>

Insert: V



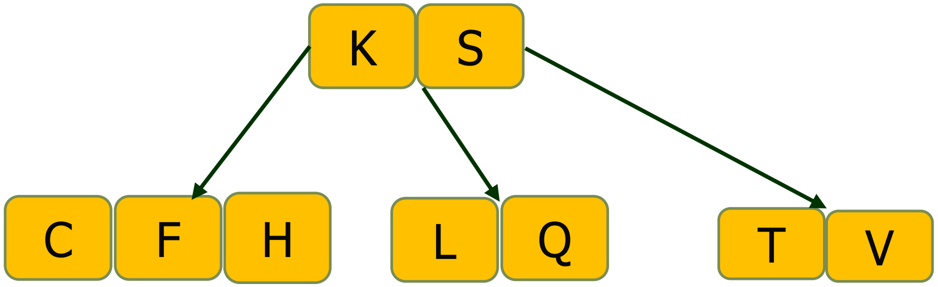
B Tree (Insertion)

Example 3:

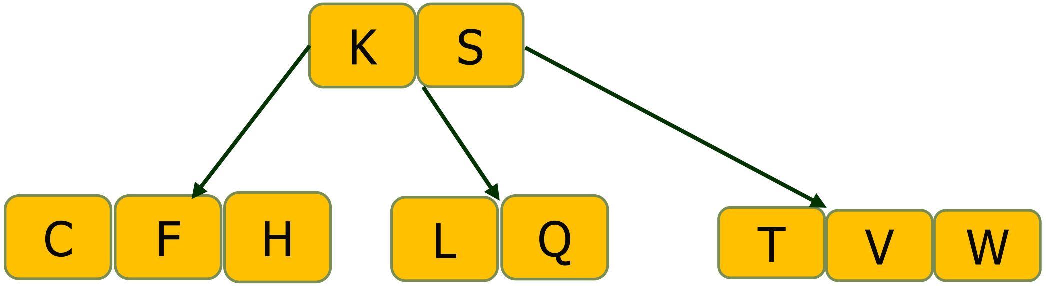
Construct a B-Tree of order 5 on following data set and assume that B-Tree is initially empty.

<F, S, Q, K, C, L, H, T, V, W, M, R, N, P, A, B, X, Y, Z, D, E>

Insert: V



Insert: W



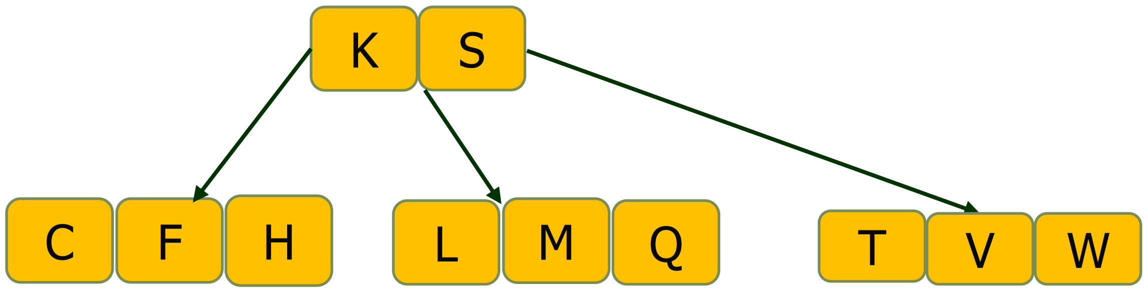
B Tree (Insertion)

Example 3:

Construct a B-Tree of order 5 on following data set and assume that B-Tree is initially empty.

<F, S, Q, K, C, L, H, T, V, W, M, R, N, P, A, B, X, Y, Z, D, E>

Insert: M



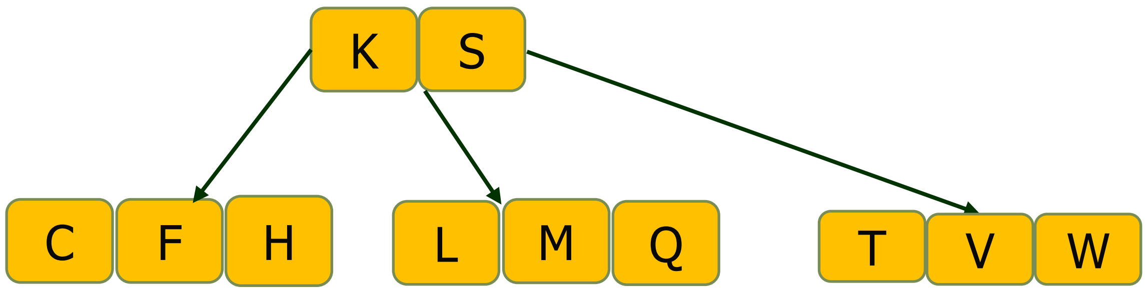
B Tree (Insertion)

Example 3:

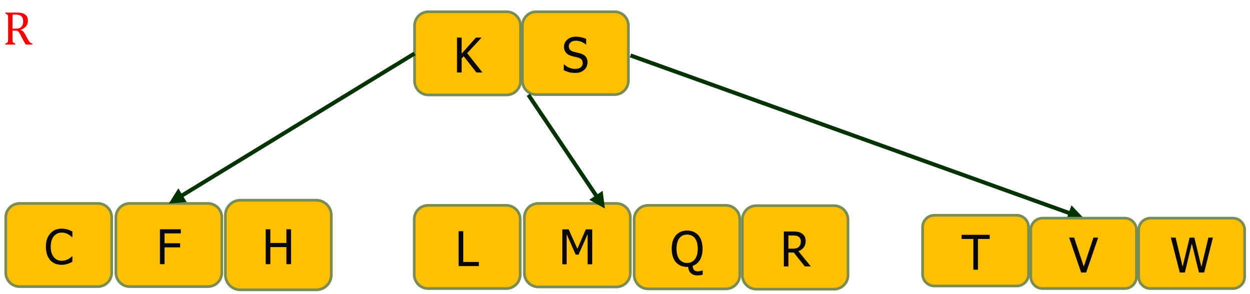
Construct a B-Tree of order 5 on following data set and assume that B-Tree is initially empty.

<F, S, Q, K, C, L, H, T, V, W, M, R, N, P, A, B, X, Y, Z, D, E>

Insert: M



Insert: R



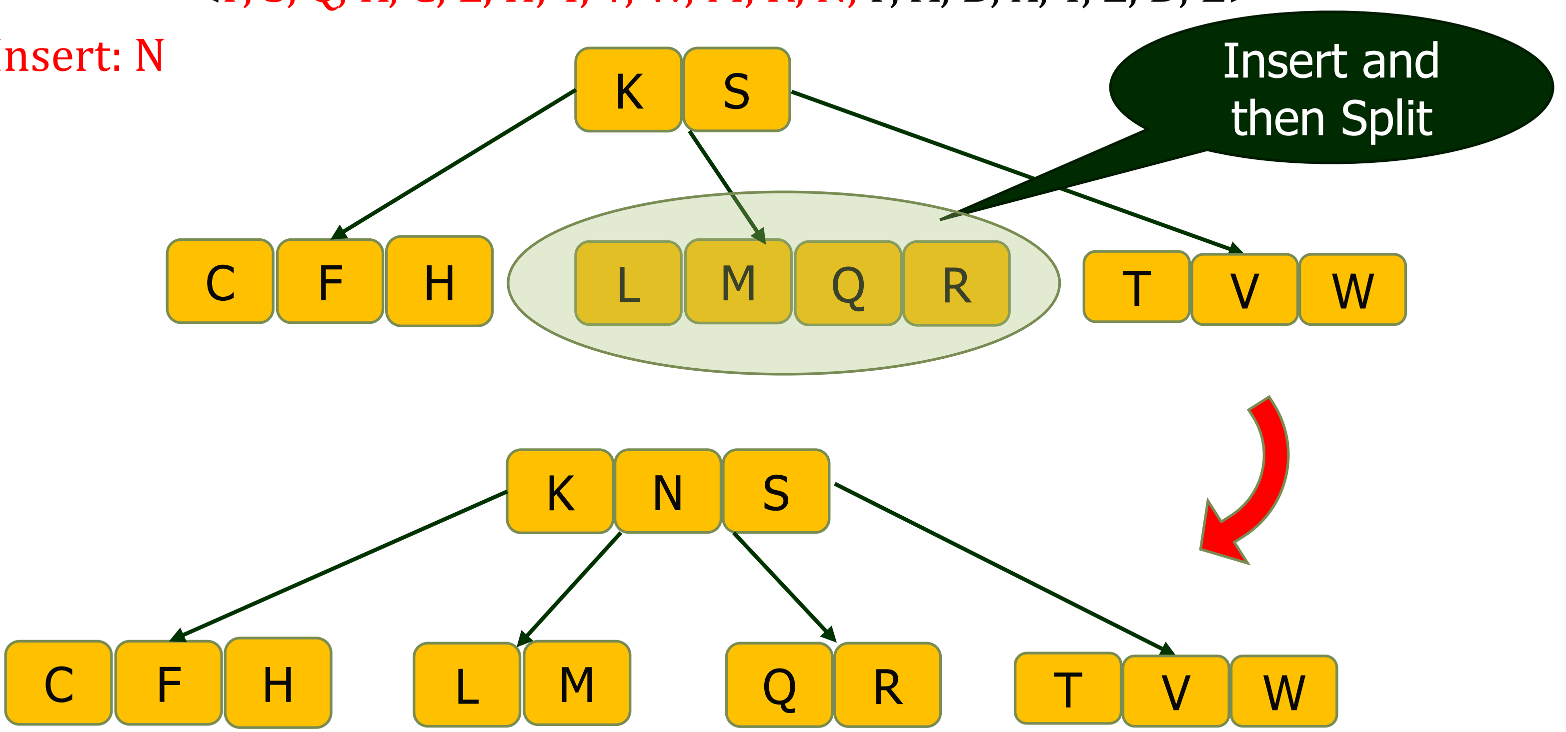
B Tree (Insertion)

Example 3:

Construct a B-Tree of order 5 on following data set and assume that B-Tree is initially empty.

<F, S, Q, K, C, L, H, T, V, W, M, R, N, P, A, B, X, Y, Z, D, E>

Insert: N



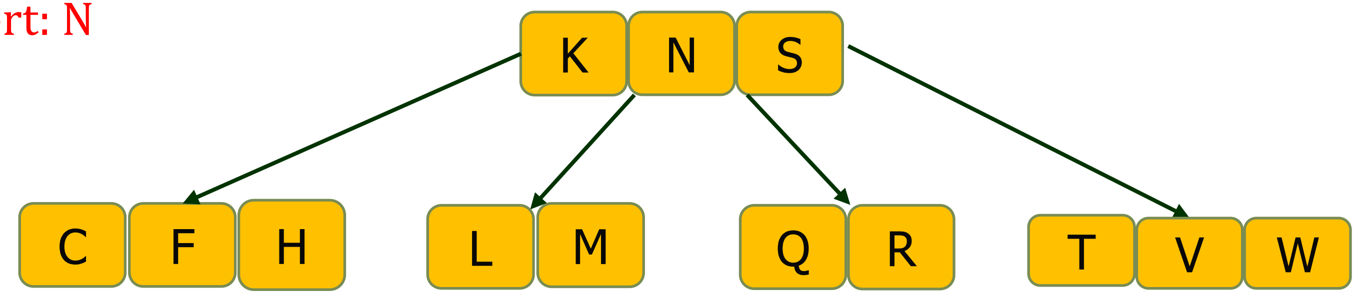
B Tree (Insertion)

Example 3:

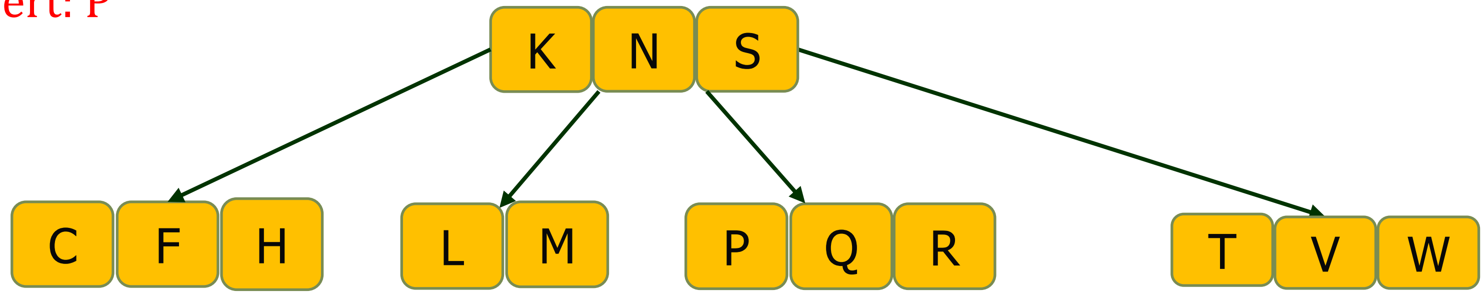
Construct a B-Tree of order 5 on following data set and assume that B-Tree is initially empty.

<F, S, Q, K, C, L, H, T, V, W, M, R, N, P, A, B, X, Y, Z, D, E>

Insert: N



Insert: P



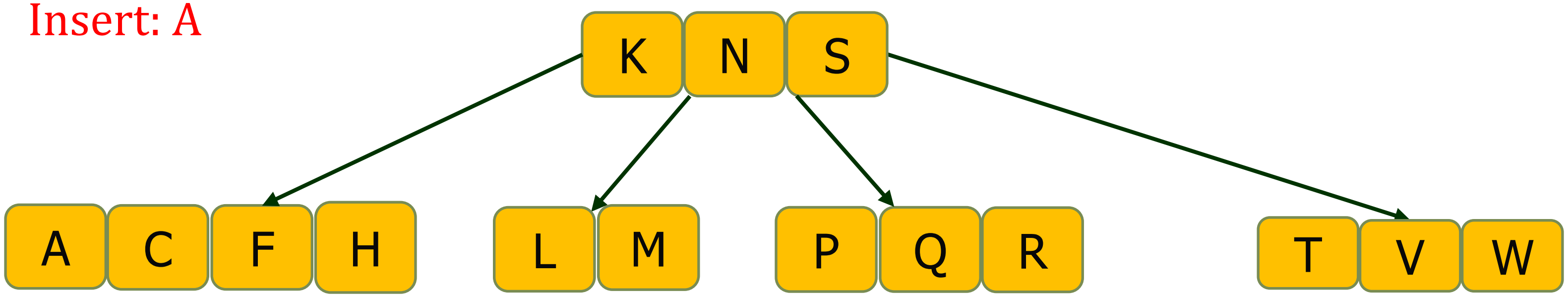
B Tree (Insertion)

Example 3:

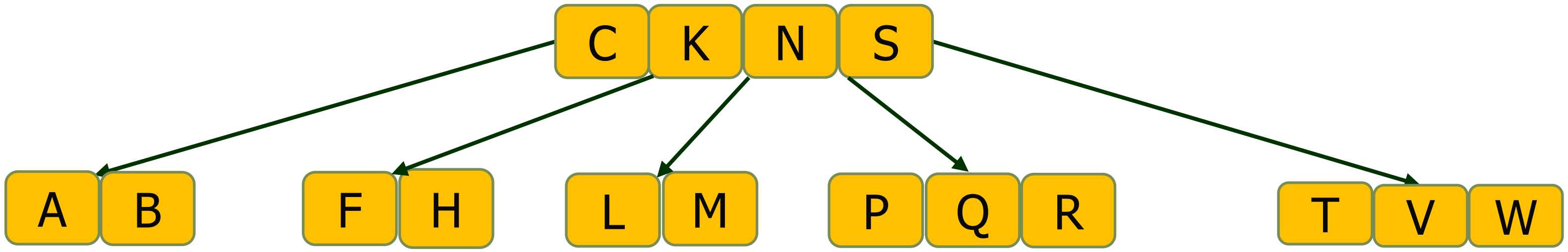
Construct a B-Tree of order 5 on following data set and assume that B-Tree is initially empty.

<F, S, Q, K, C, L, H, T, V, W, M, R, N, P, A, B, X, Y, Z, D, E>

Insert: A



Insert: B (First Insert B then Split)



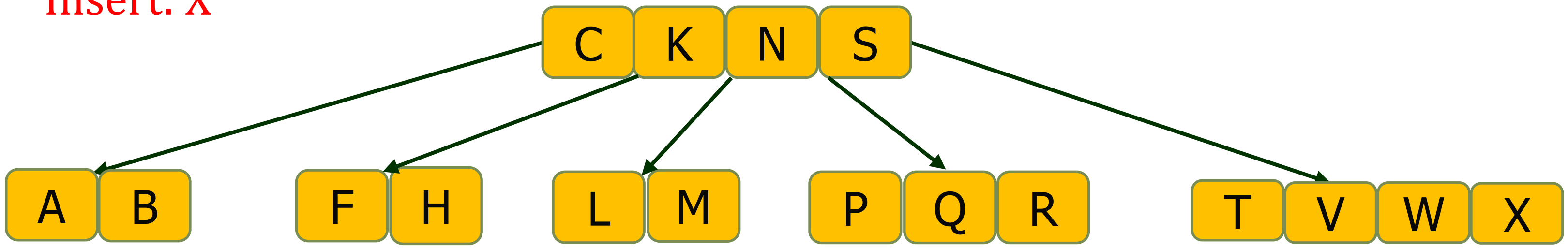
B Tree (Insertion)

Example 3:

Construct a B-Tree of order 5 on following data set and assume that B-Tree is initially empty.

<F, S, Q, K, C, L, H, T, V, W, M, R, N, P, A, B, X, Y, Z, D, E>

Insert: X



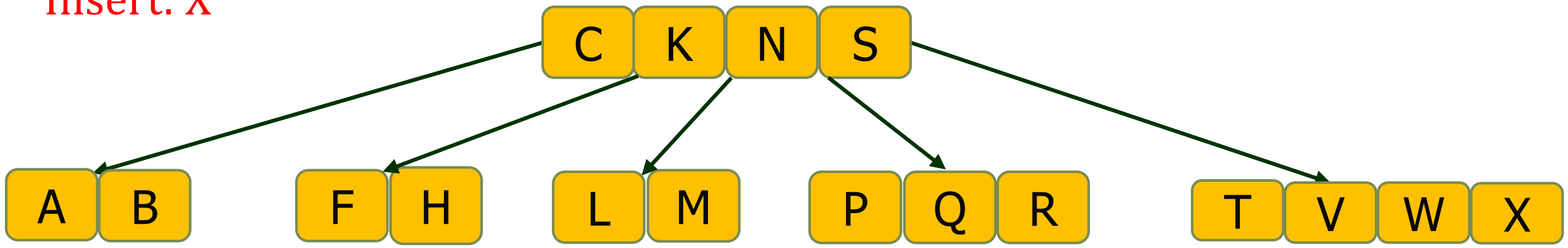
B Tree (Insertion)

Example 3:

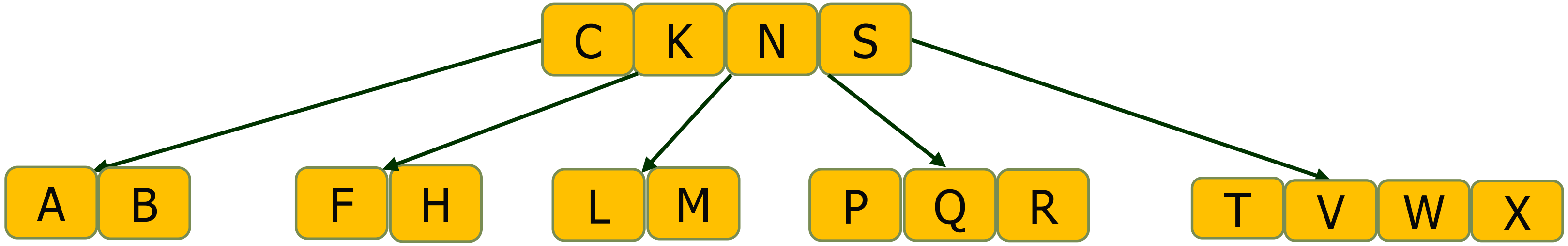
Construct a B-Tree of order 5 on following data set and assume that B-Tree is initially empty.

<F, S, Q, K, C, L, H, T, V, W, M, R, N, P, A, B, X, Y, Z, D, E>

Insert: X



Insert: Y (Insert Y and then split)



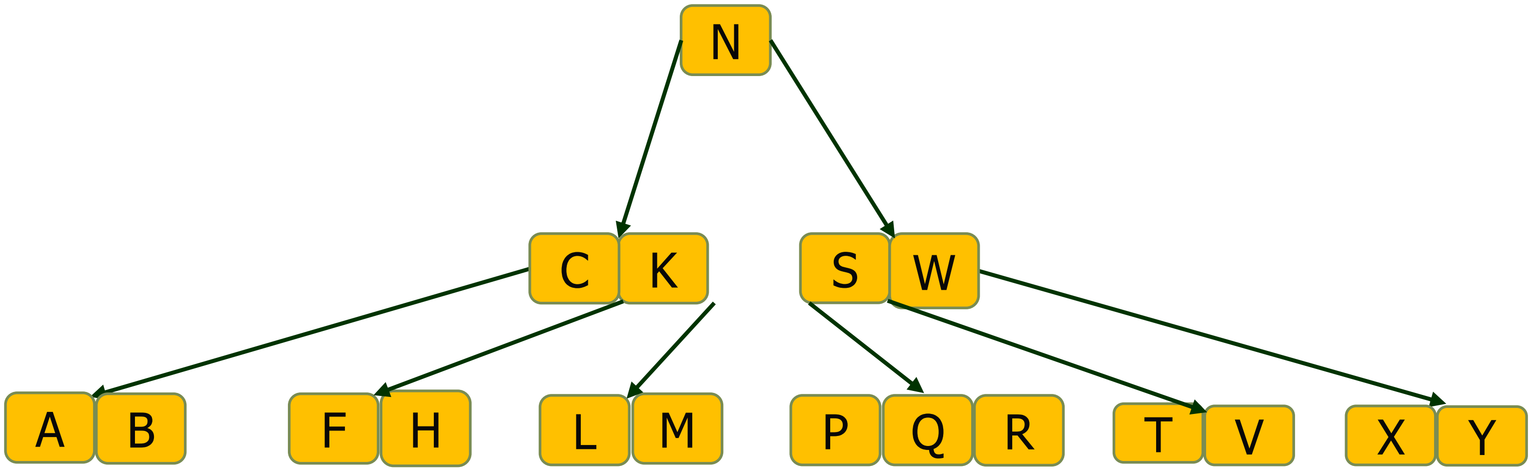
B Tree (Insertion)

Example 3:

Construct a B-Tree of order 5 on following data set and assume that B-Tree is initially empty.

<F, S, Q, K, C, L, H, T, V, W, M, R, N, P, A, B, X, Y, Z, D, E>

Insert: Y (Insert Y and then split and again insert W on Root and then again split)



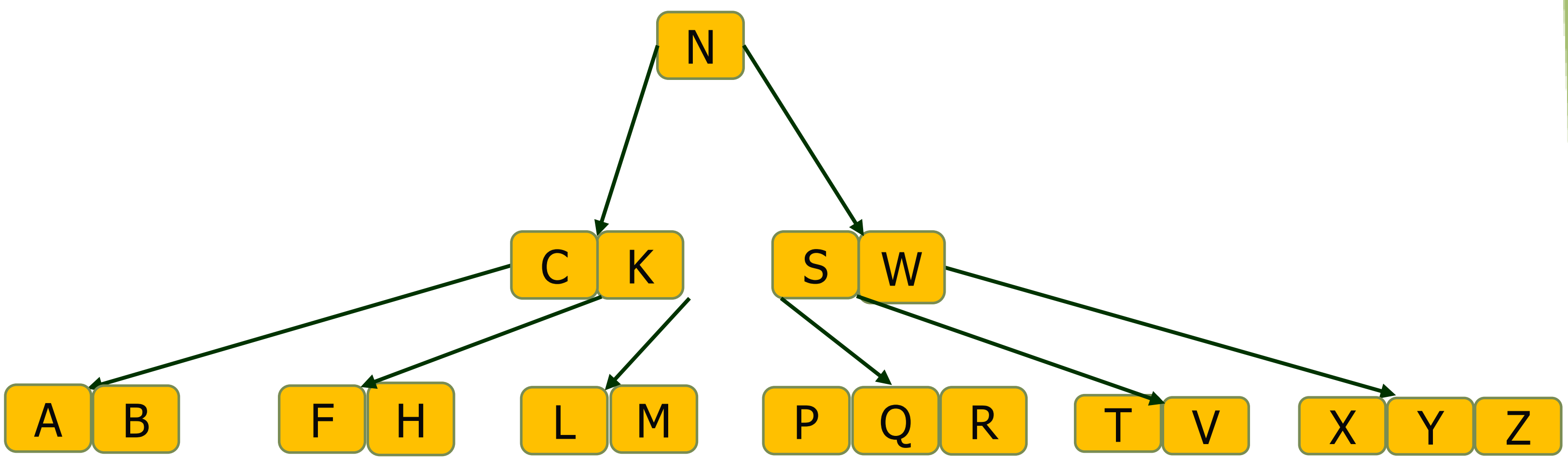
B Tree (Insertion)

Example 3:

Construct a B-Tree of order 5 on following data set and assume that B-Tree is initially empty.

<F, S, Q, K, C, L, H, T, V, W, M, R, N, P, A, B, X, Y, Z, D, E>

Insert: Z



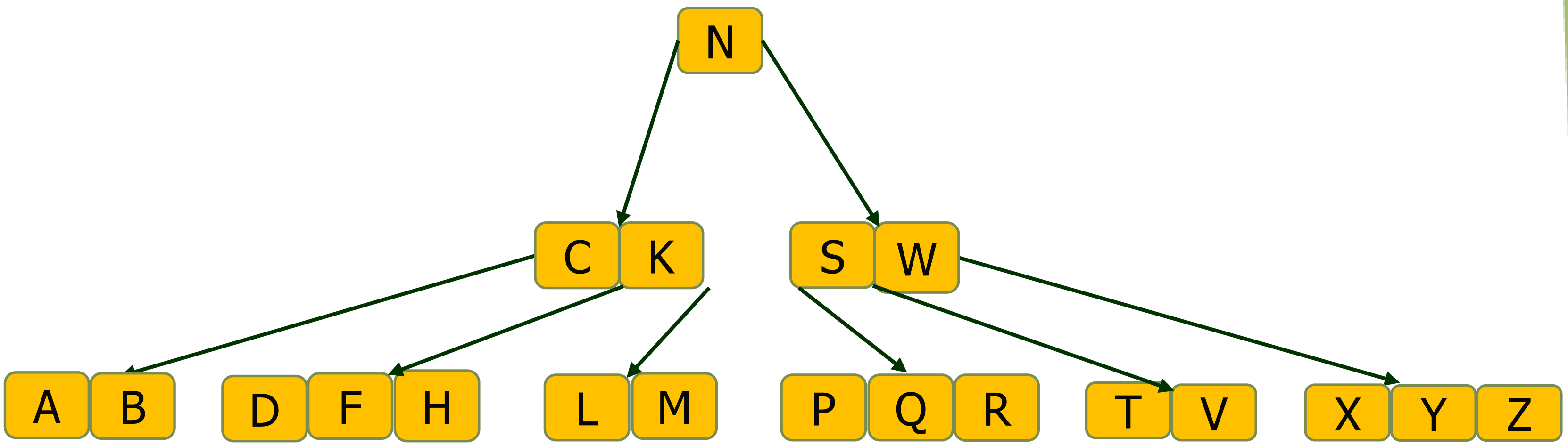
B Tree (Insertion)

Example 3:

Construct a B-Tree of order 5 on following data set and assume that B-Tree is initially empty.

<F, S, Q, K, C, L, H, T, V, W, M, R, N, P, A, B, X, Y, Z, D, E>

Insert: D



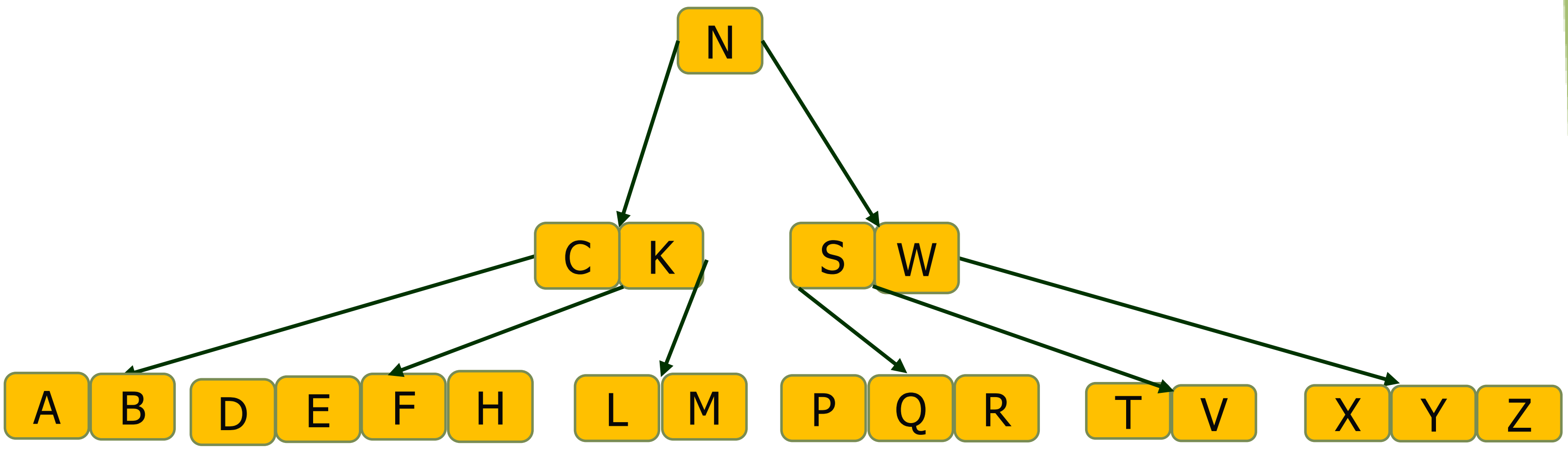
B Tree (Insertion)

Example 3:

Construct a B-Tree of order 5 on following data set and assume that B-Tree is initially empty.

<F, S, Q, K, C, L, H, T, V, W, M, R, N, P, A, B, X, Y, Z, D, E>

Insert: E



B Tree (Deletion)

B Tree (Deletion)

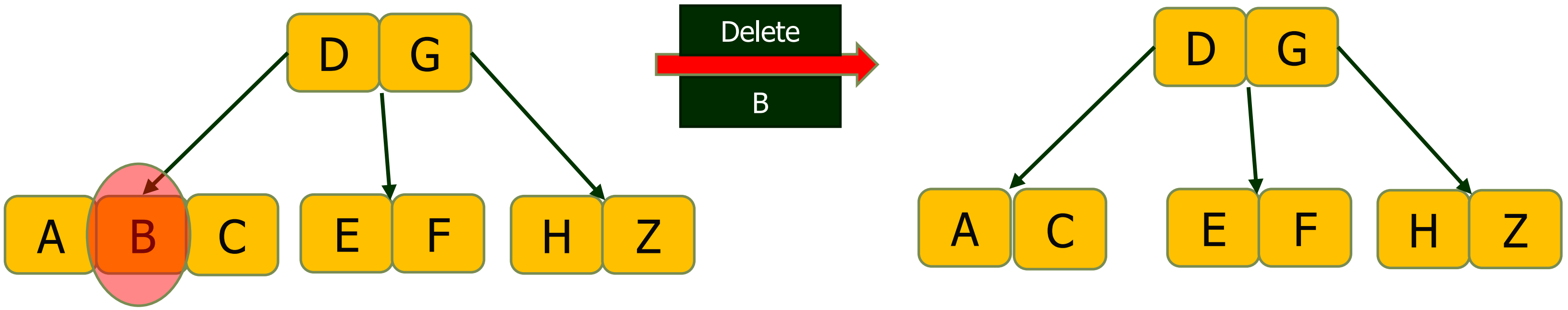
Deletion from a B-tree is analogous to insertion but a little more complicated.

Let us sketch illustrates the various cases of deleting keys from a B-tree.

Case 1: If x (one of the key to be deleted from) is a leaf node and the leaf node have more than (t-1) keys then the key can just be removed without disturbing the tree.

Let the degree (t)=3

For Example:

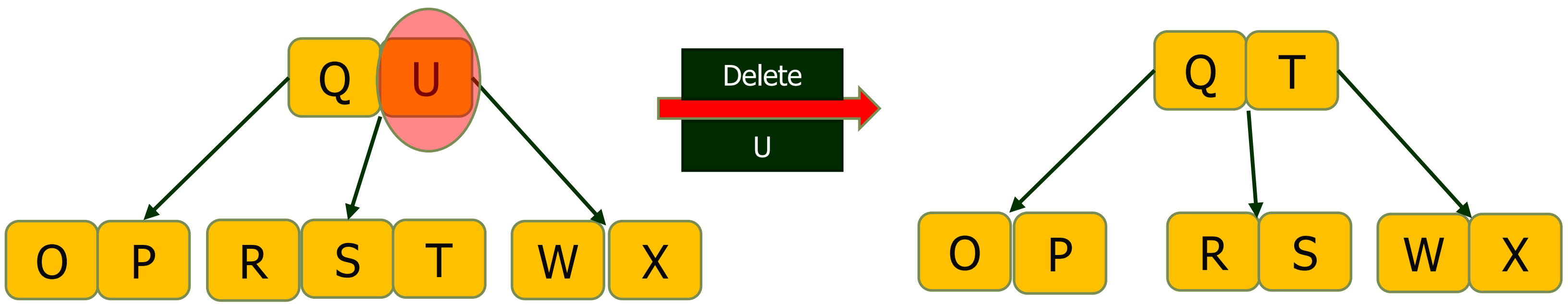


B Tree (Deletion)

Case 2(a): if x (one of the key to be deleted from) is an internal node and the key left children have at least t key, then the largest value can be moved up to replace the k.

Let the degree (t)=3

For Example:

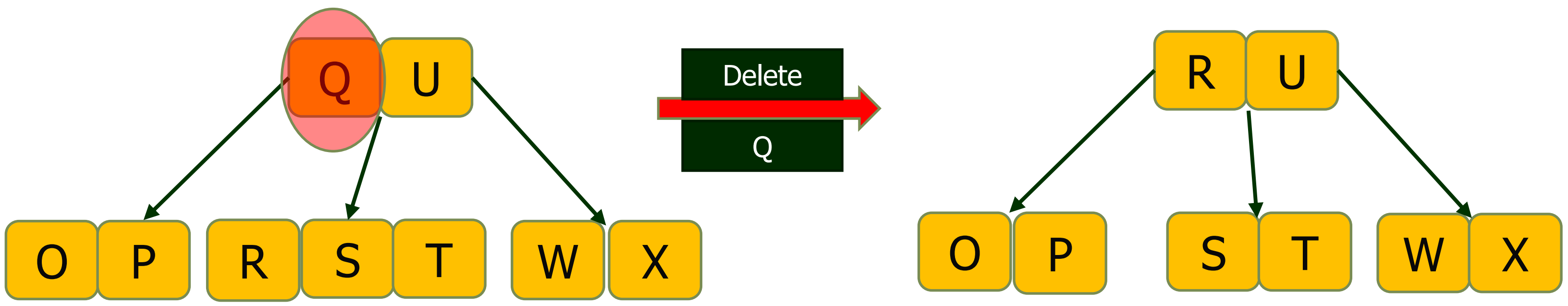


B Tree (Deletion)

Case 2(b): if x (one of the key to be deleted from) is an internal node and the key right children have at least t key, then the smallest value can be moved up to replace the k.

Let the degree (t)=3

For Example:

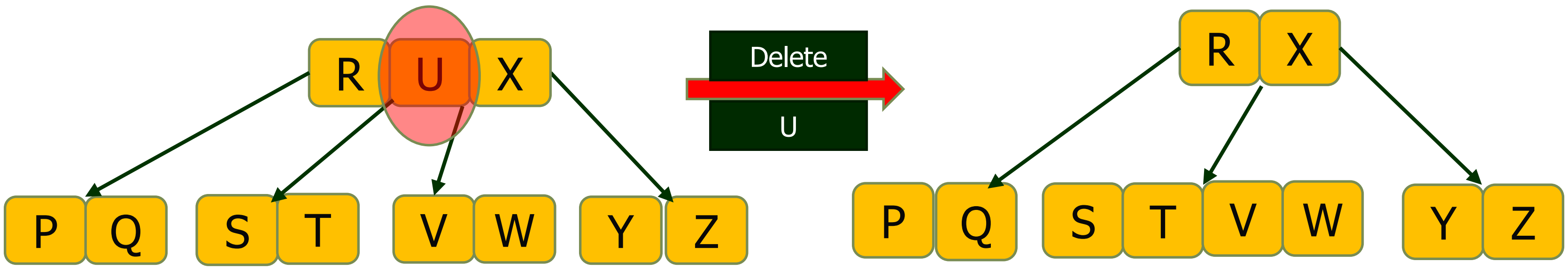


B Tree (Deletion)

Case 2(c): if x (one of the key to be deleted from) is an internal node neither its child has at least t keys then the two childs of keys must be merge into one and key must be removed.

Let the degree (t)=3

For Example:



B Tree (Deletion)

Case 3:

If the key k is not present in internal node x, determine the root of the appropriate subtree that must contain k, if k is in the tree at all.

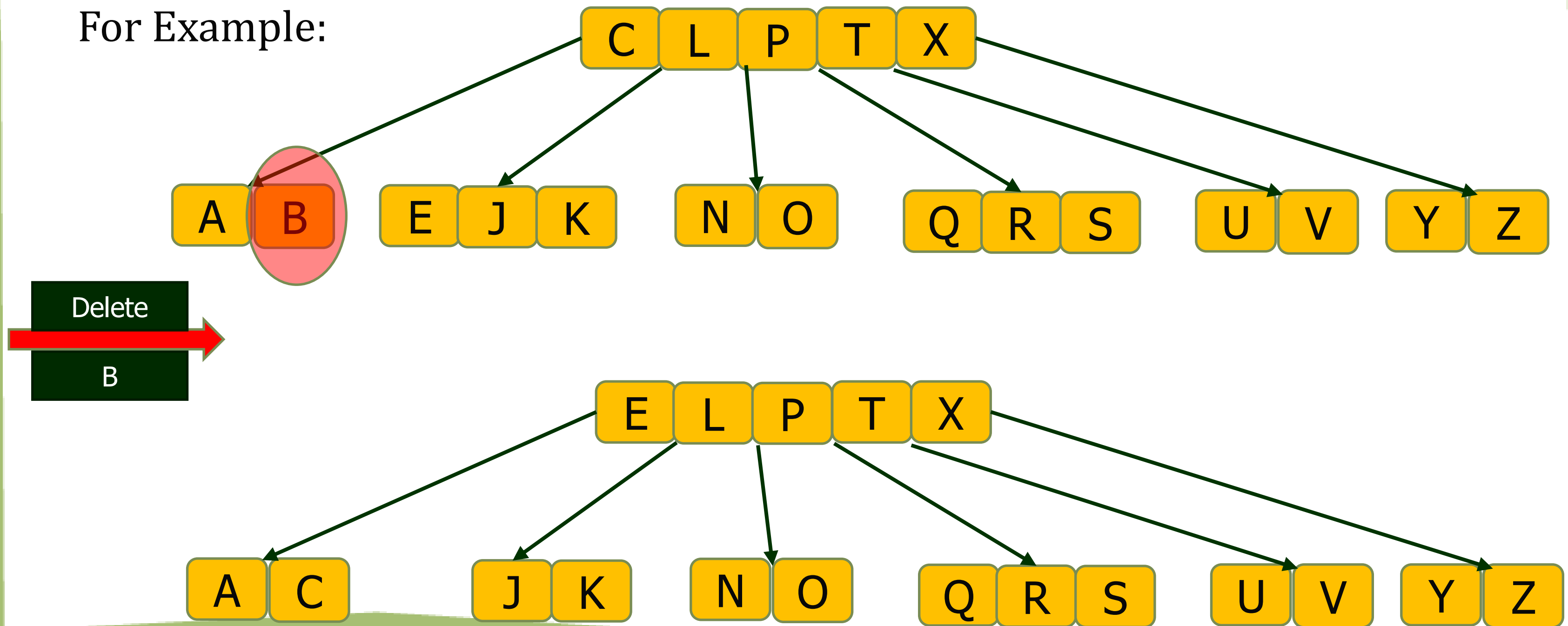
If x has only $t - 1$ keys, execute step 3a or 3b as necessary to guarantee that we descend to a node containing at least t keys. Then finish by recursing on the appropriate child of x.

B Tree (Deletion)

Case 3 (a): If x has (t-1) keys but has an immediate sibling with at least t keys, give x an extra key by moving a key from p[x] down into x, moving a key from x's immediate left or right sibling up into p[x], and moving the appropriate child pointer from the sibling into x.

Let the degree (t)=3

For Example:



B Tree (Deletion)

Case 3: If x has (t-1) keys

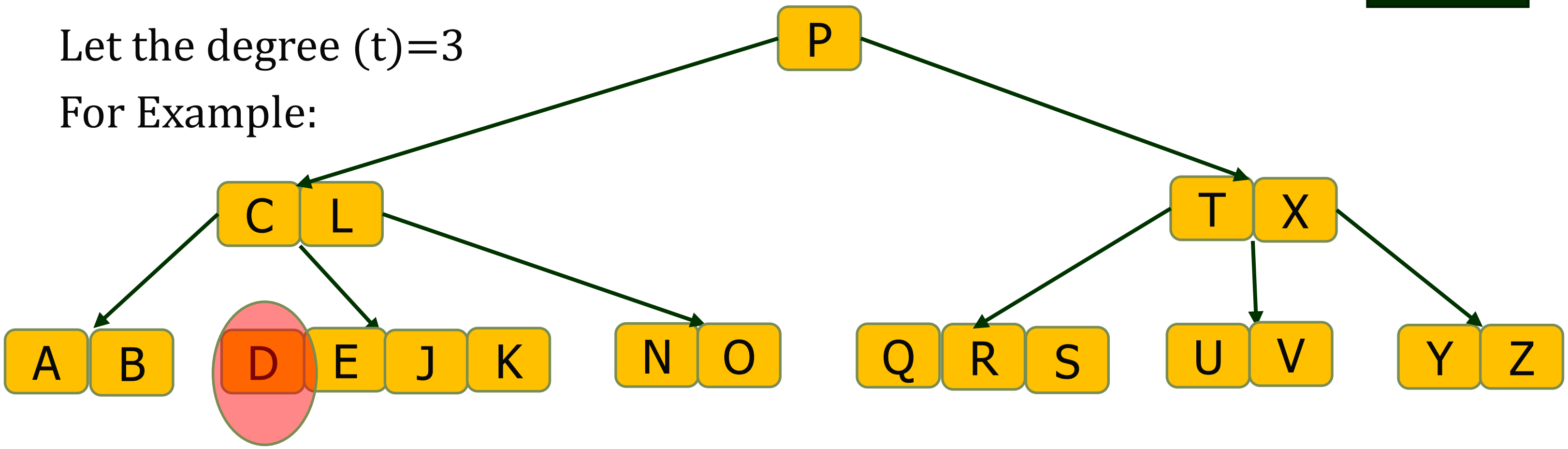
(b) if p[x] (one of the key to be deleted from) and its immediate sibling also have (t-1) keys, then merge p[x] with one of its sibling by bringing down the p[p[x]] to be the median value and then delete the desired key.

Delete

D

Let the degree (t)=3

For Example:



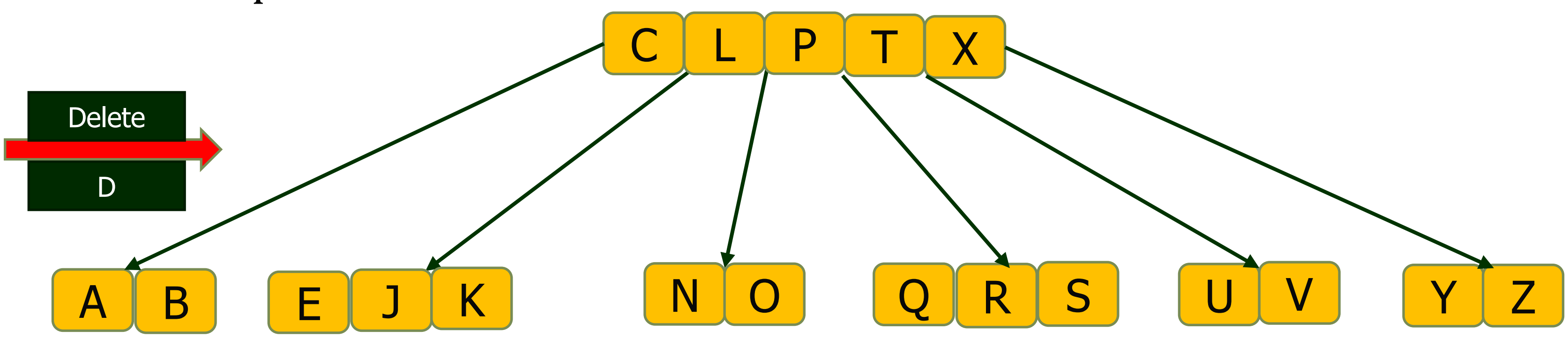
B Tree (Deletion)

Case 3: If x has (t-1) keys

(b) if p[x] (one of the key to be deleted from) and its immediate sibling also have (t-1) keys, then merge p[x] with one of its sibling by bringing down the p[p[x]] to be the median value and then delete the desired key.

Let the degree (t)=3

For Example:

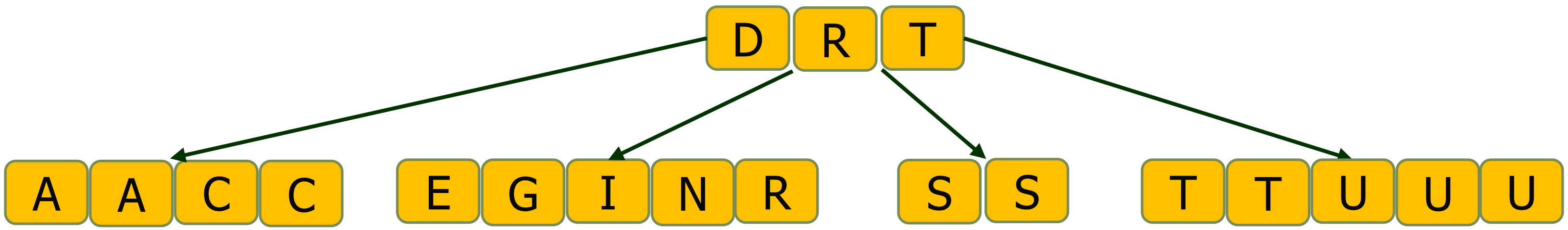


B Tree (Deletion)

Example 1:

Perform the deletion operation with the following data sequentially on the given B-Tree of degree 3.

<T S U T D C I N R C>



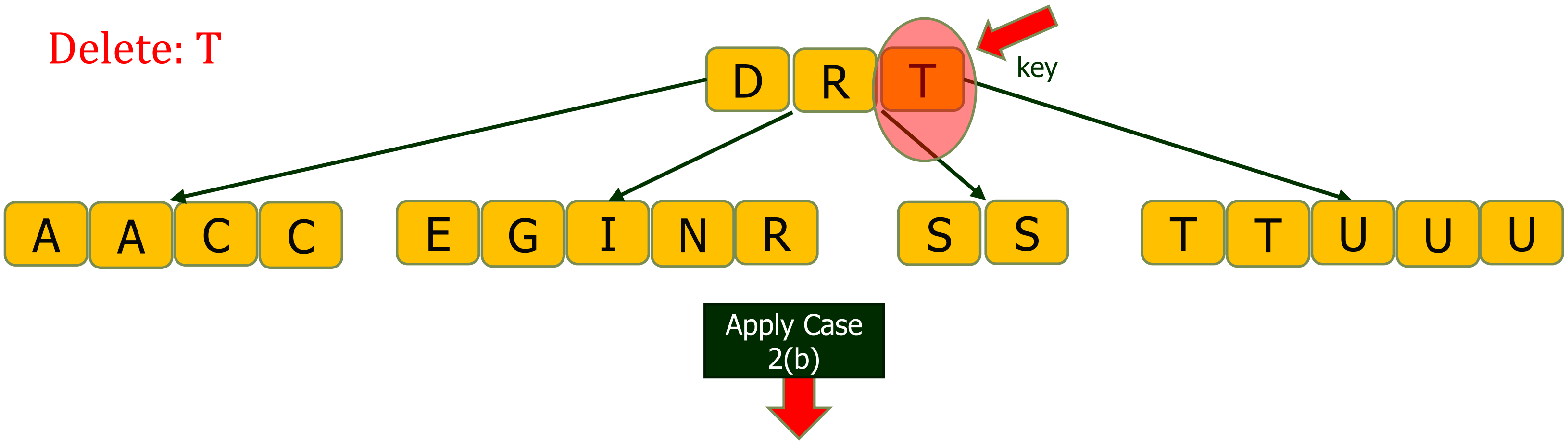
B Tree (Deletion)

Example 1:

Perform the deletion operation with the following data sequentially on the given B-Tree of degree 3.

<T S U T D C I N R C>

Delete: T



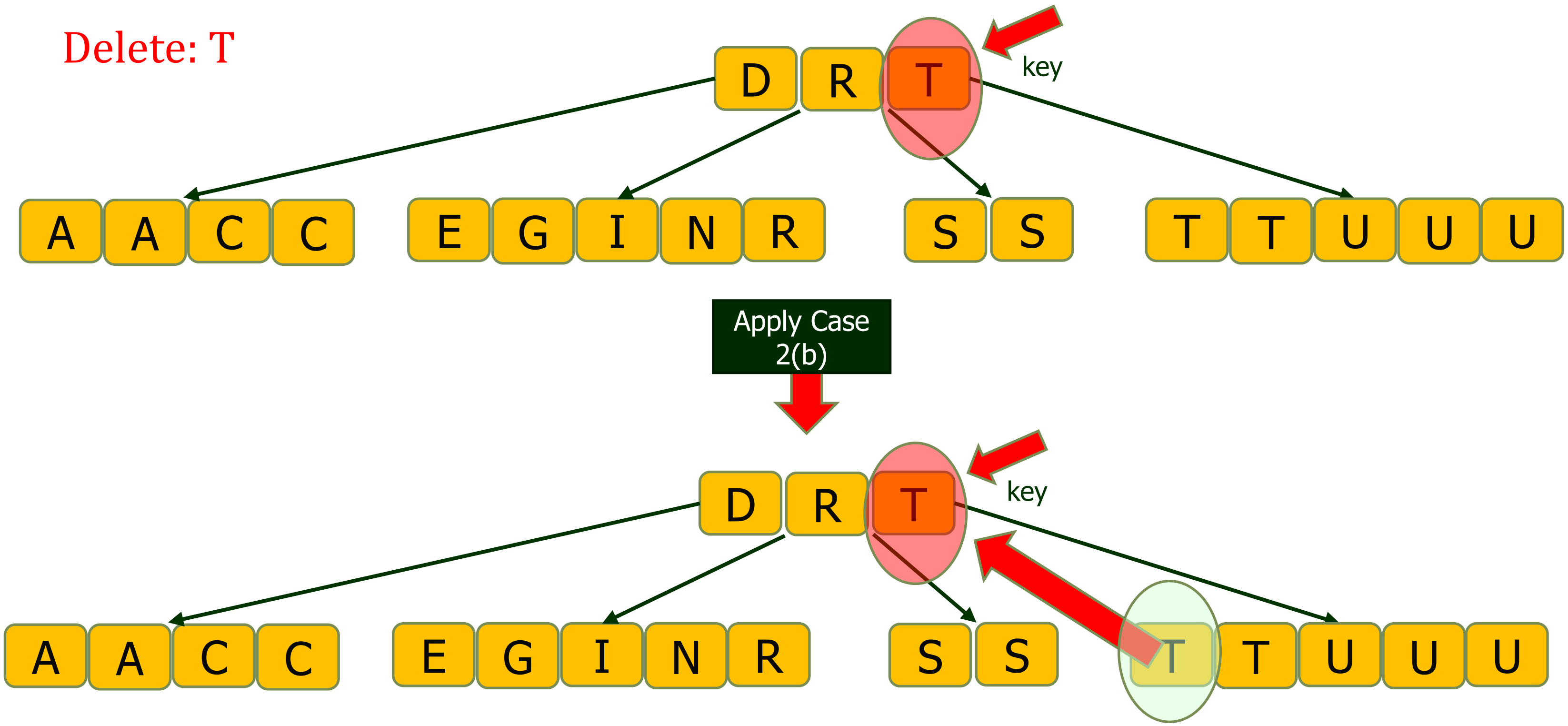
B Tree (Deletion)

Example 1:

Perform the deletion operation with the following data sequentially on the given B-Tree of degree 3.

<T S U T D C I N R C>

Delete: T



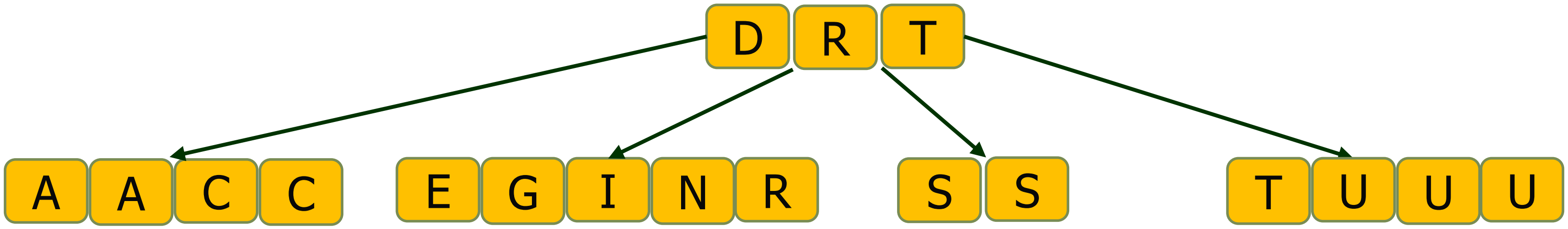
B Tree (Deletion)

Example 1:

Perform the deletion operation with the following data sequentially on the given B-Tree of degree 3.

<T S U T D C I N R C>

The updated Tree is



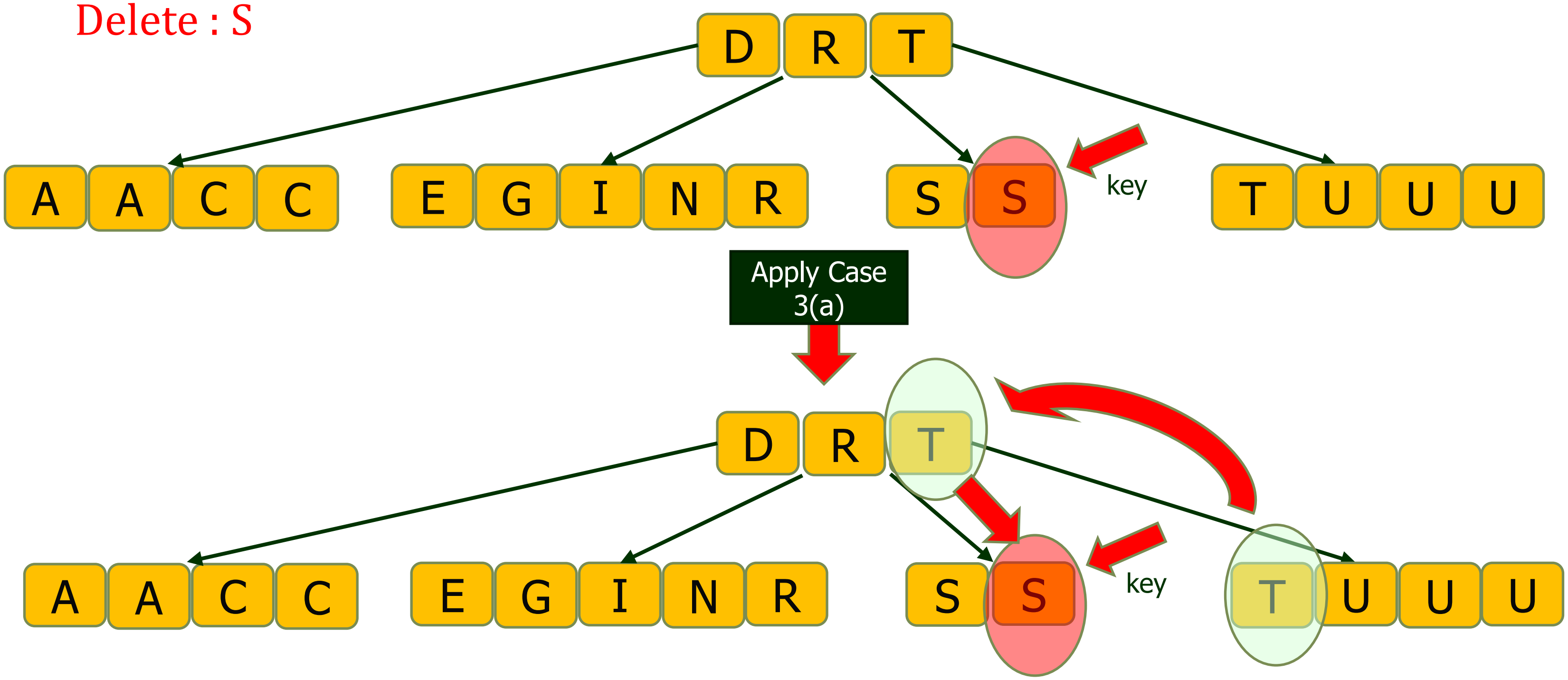
B Tree (Deletion)

Example 1:

Perform the deletion operation with the following data sequentially on the given B-Tree of degree 3.

<T S U T D C I N R C>

Delete : S



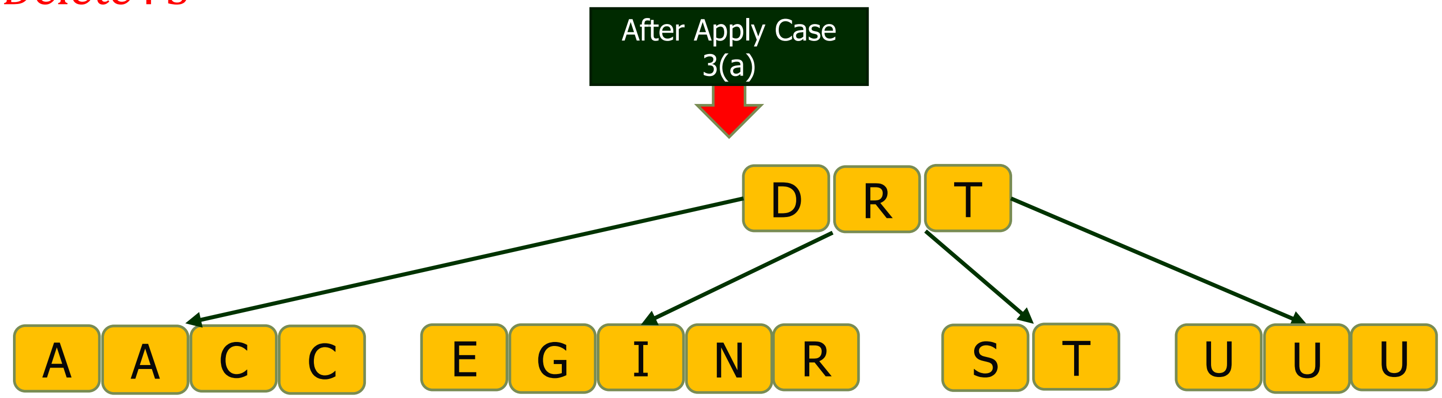
B Tree (Deletion)

Example 1:

Perform the deletion operation with the following data sequentially on the given B-Tree of degree 3.

<T S U T D C I N R C>

Delete : S



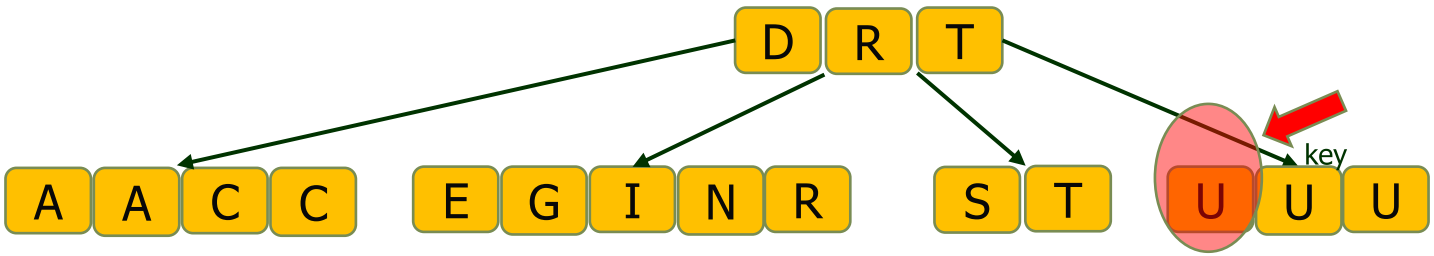
B Tree (Deletion)

Example 1:

Perform the deletion operation with the following data sequentially on the given B-Tree of degree 3.

<T S U T D C I N R C>

Delete : U



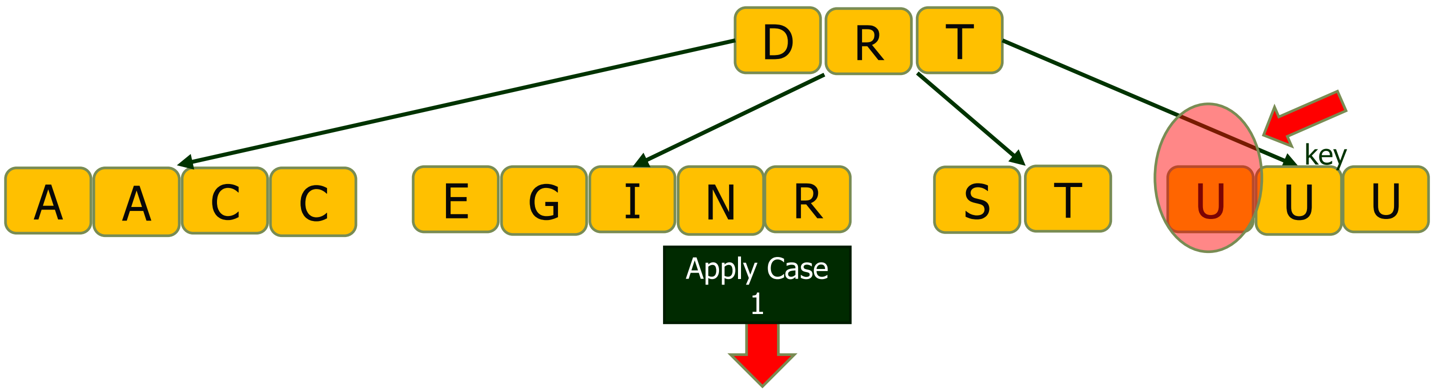
B Tree (Deletion)

Example 1:

Perform the deletion operation with the following data sequentially on the given B-Tree of degree 3.

<T S U T D C I N R C>

Delete : U



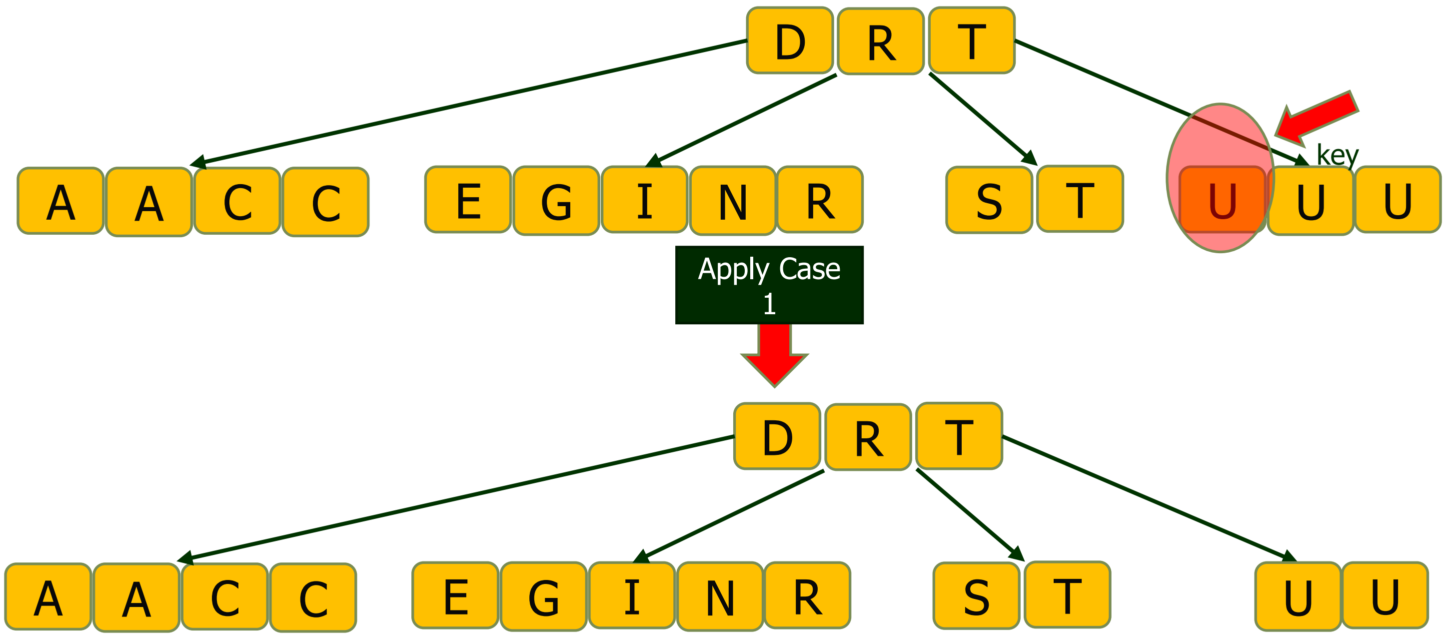
B Tree (Deletion)

Example 1:

Perform the deletion operation with the following data sequentially on the given B-Tree of degree 3.

<T S U T D C I N R C>

Delete : U



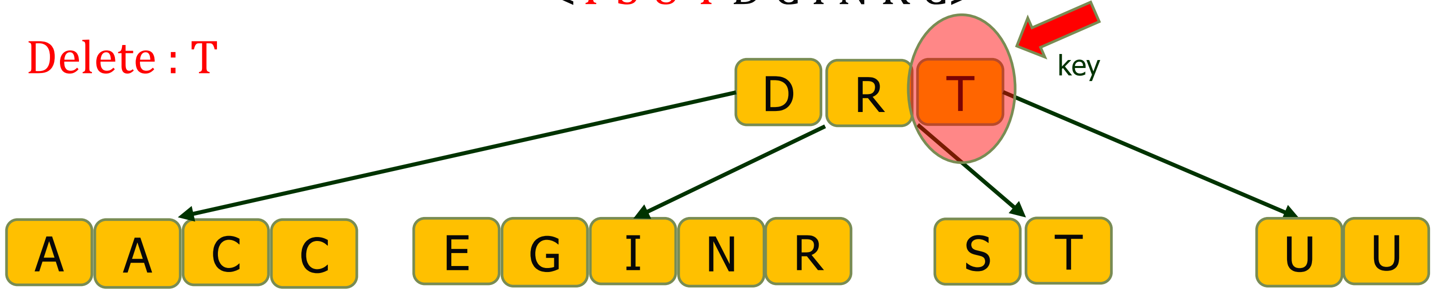
B Tree (Deletion)

Example 1:

Perform the deletion operation with the following data sequentially on the given B-Tree of degree 3.

<T S U T D C I N R C>

Delete : T



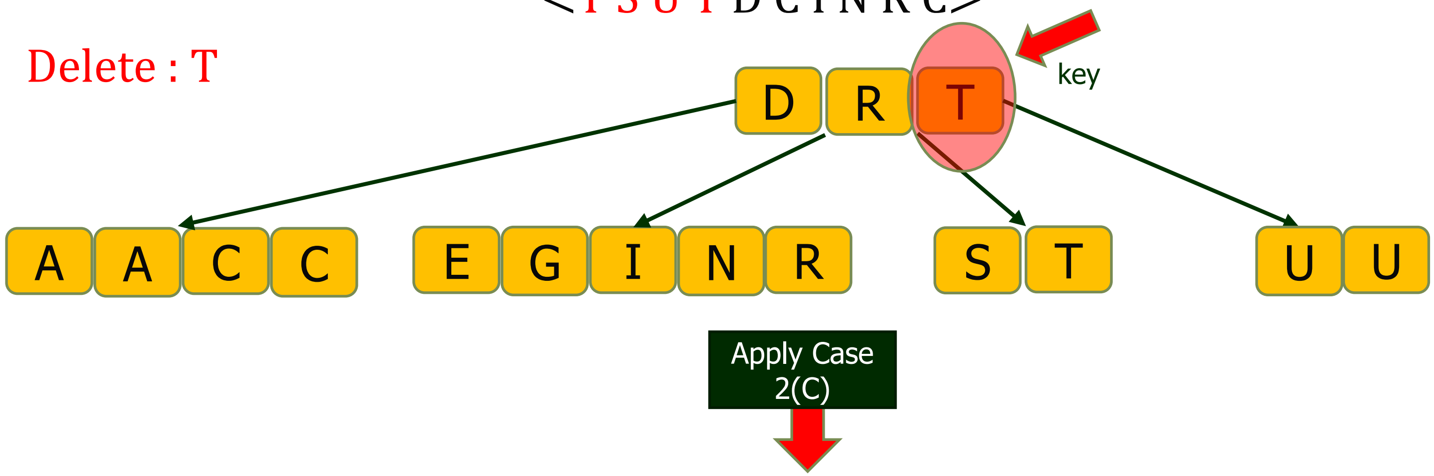
B Tree (Deletion)

Example 1:

Perform the deletion operation with the following data sequentially on the given B-Tree of degree 3.

<T S U T D C I N R C>

Delete : T



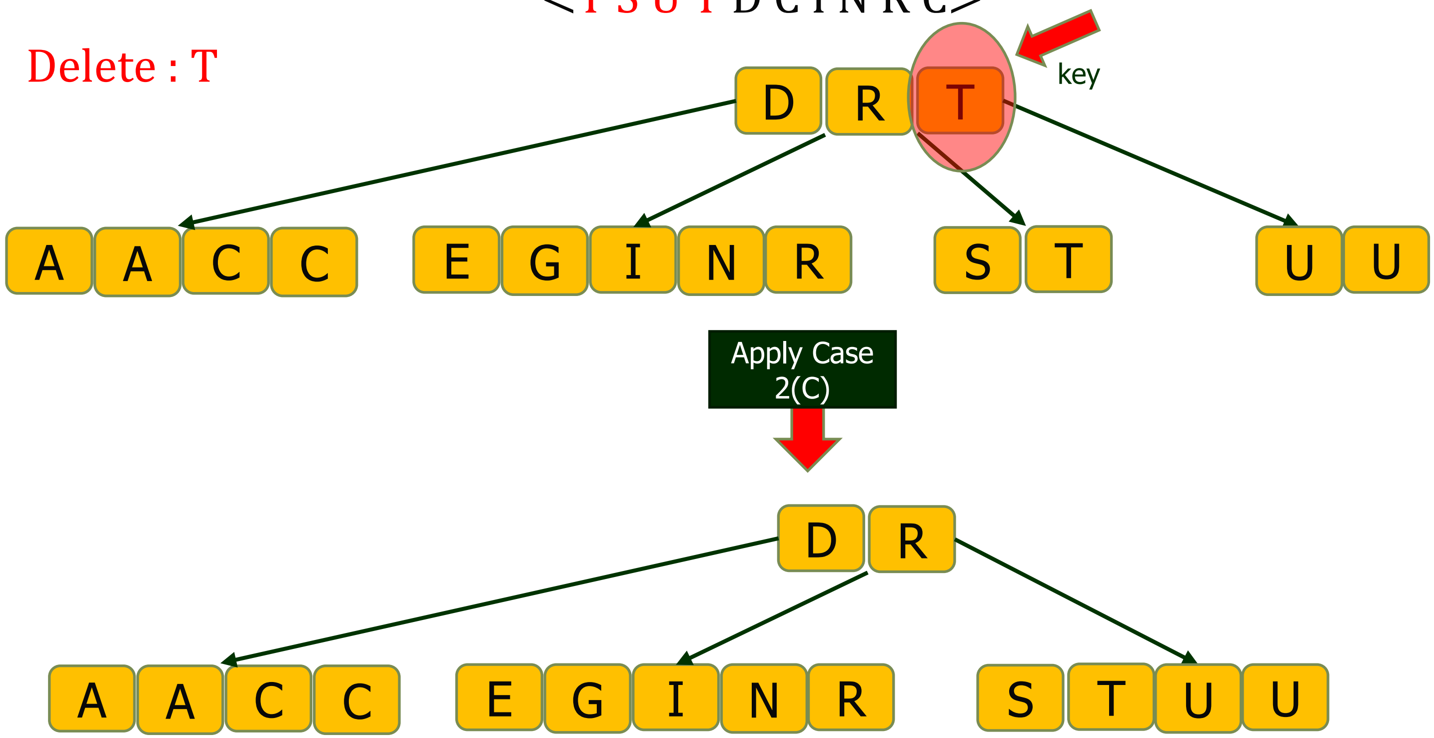
B Tree (Deletion)

Example 1:

Perform the deletion operation with the following data sequentially on the given B-Tree of degree 3.

<T S U T D C I N R C>

Delete : T



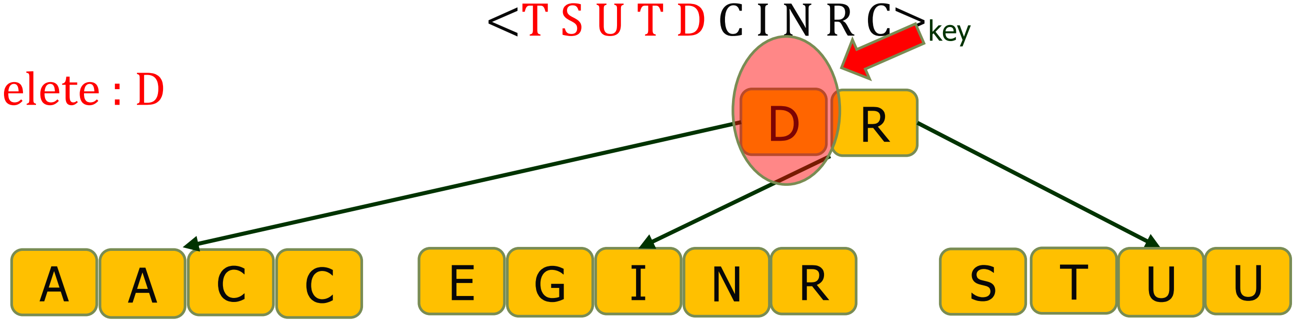
B Tree (Deletion)

Example 1:

Perform the deletion operation with the following data sequentially on the given B-Tree of degree 3.

<T S U T D C I N R C>

Delete : D



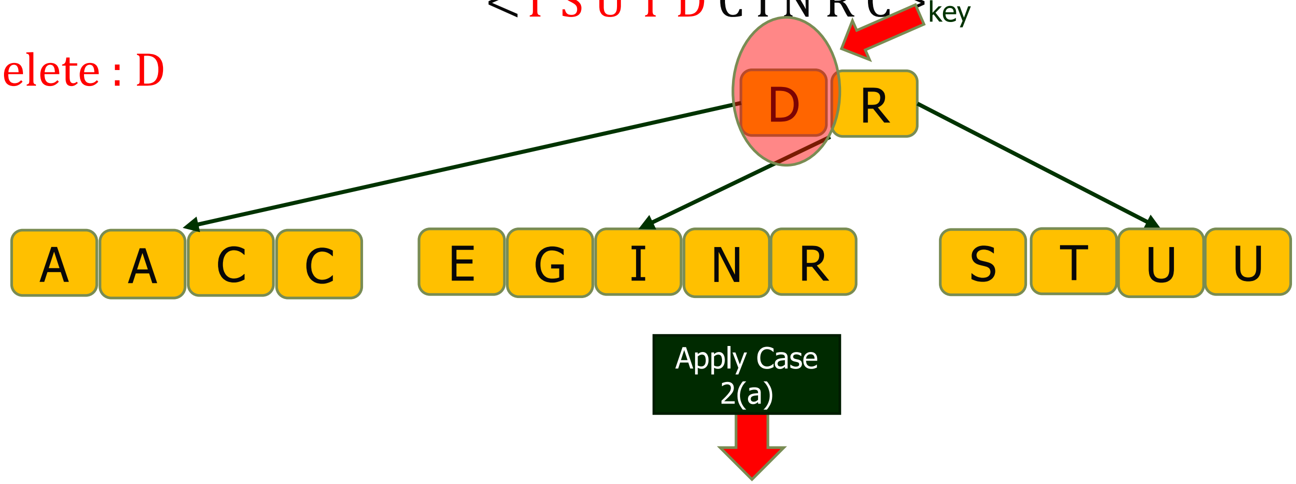
B Tree (Deletion)

Example 1:

Perform the deletion operation with the following data sequentially on the given B-Tree of degree 3.

<T S U T D C I N R C>

Delete : D



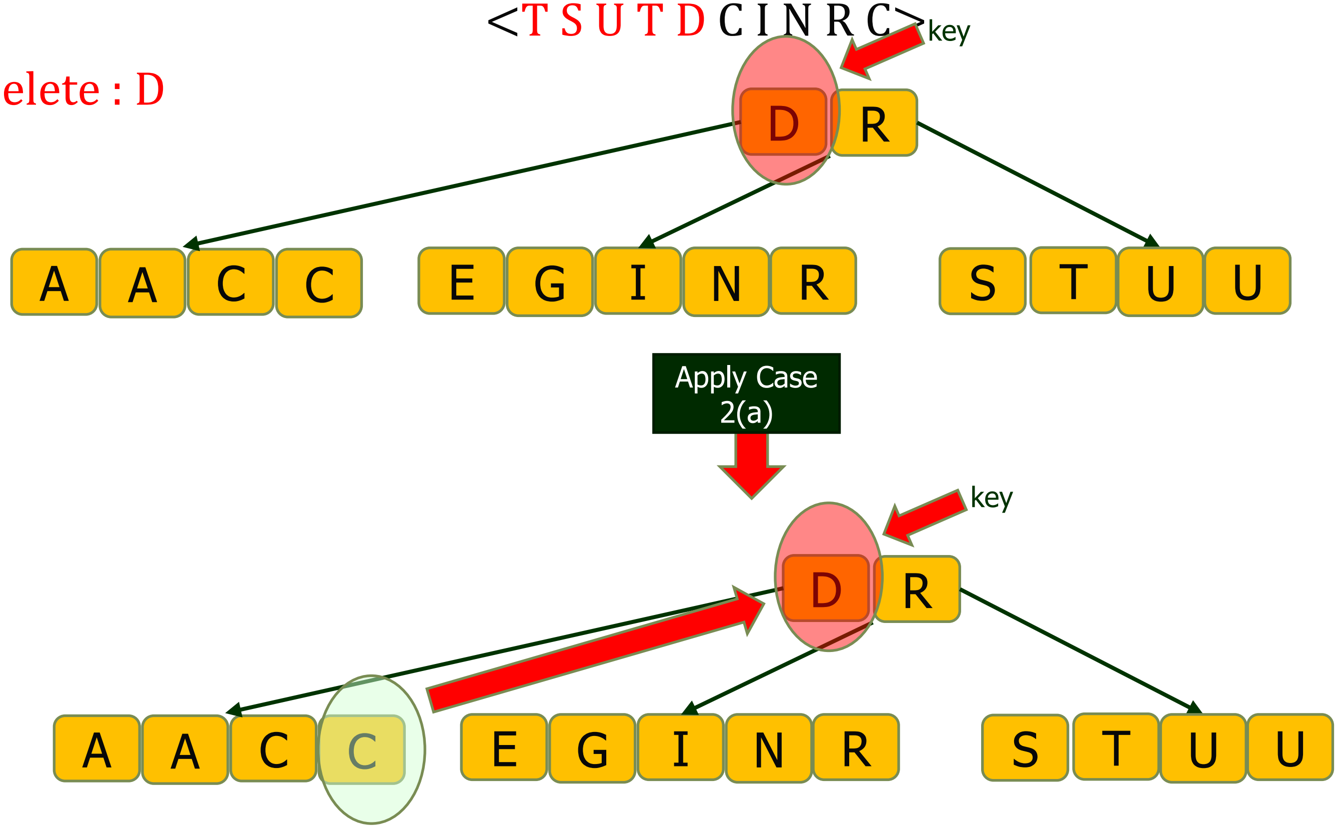
B Tree (Deletion)

Example 1:

Perform the deletion operation with the following data sequentially on the given B-Tree of degree 3.

<T S U T D C I N R C>

Delete : D



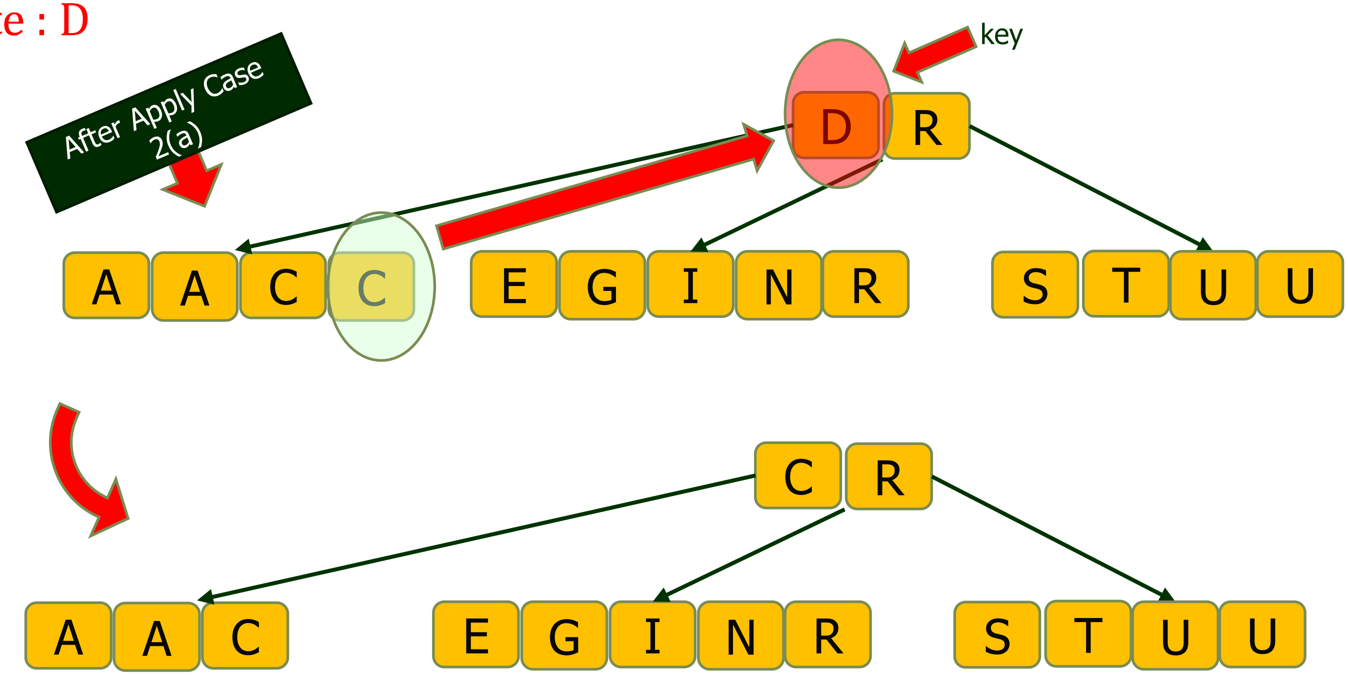
B Tree (Deletion)

Example 1:

Perform the deletion operation with the following data sequentially on the given B-Tree of degree 3.

<T S U T D C I N R C>

Delete : D



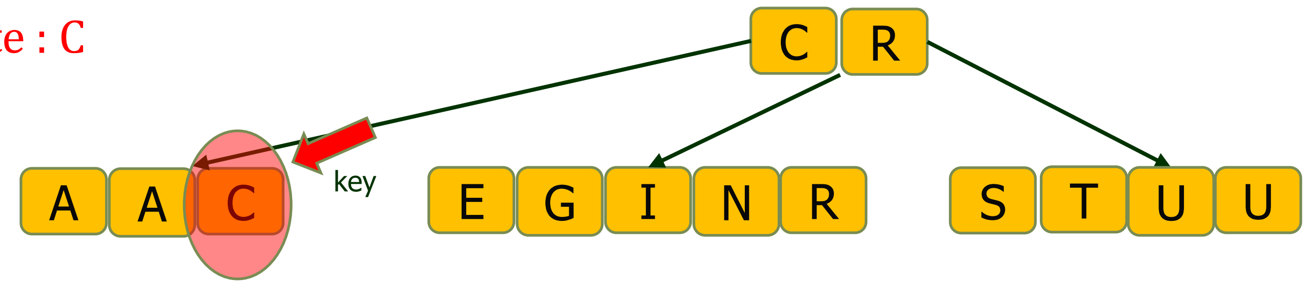
B Tree (Deletion)

Example 1:

Perform the deletion operation with the following data sequentially on the given B-Tree of degree 3.

<T S U T D C I N R C>

Delete : C



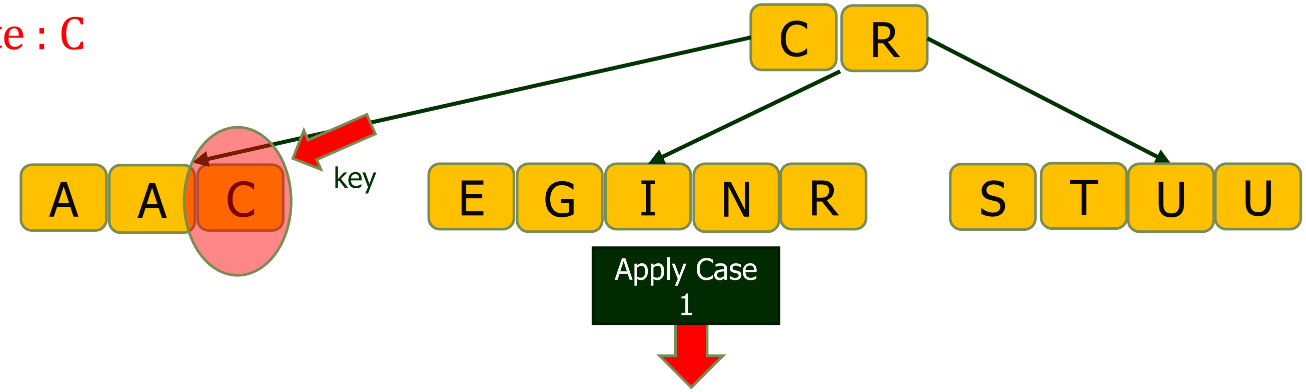
B Tree (Deletion)

Example 1:

Perform the deletion operation with the following data sequentially on the given B-Tree of degree 3.

<T S U T D C I N R C>

Delete : C



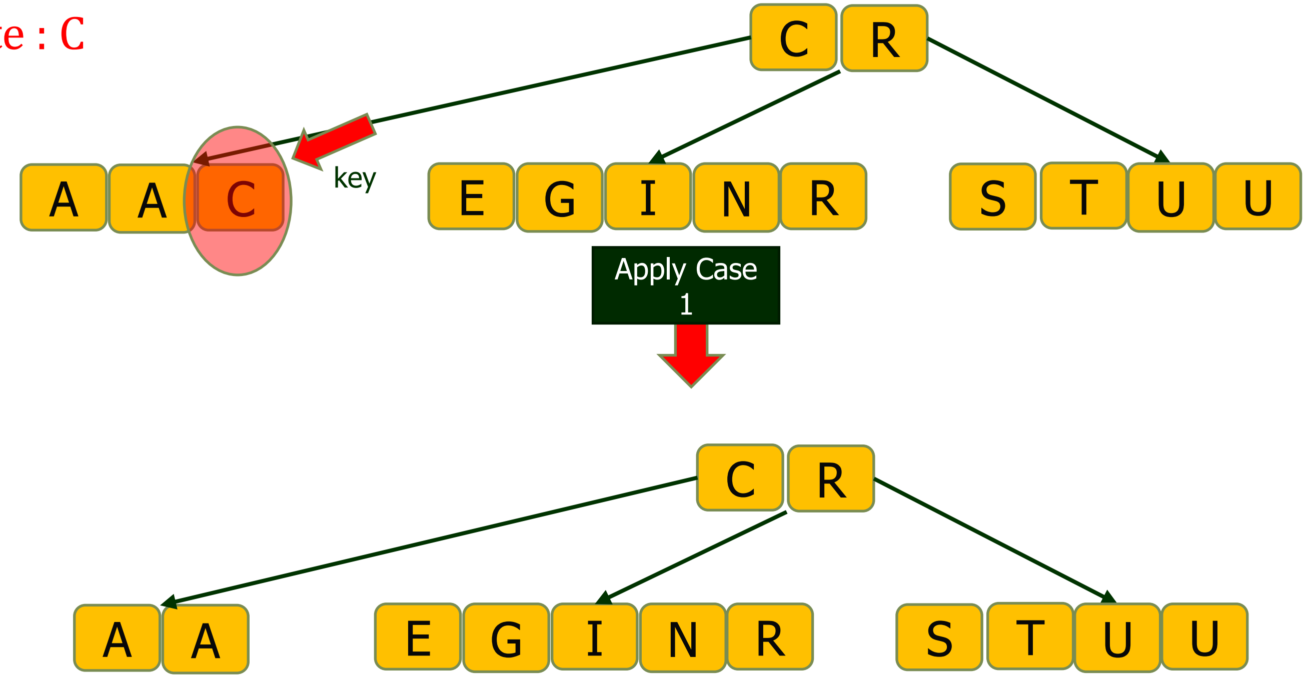
B Tree (Deletion)

Example 1:

Perform the deletion operation with the following data sequentially on the given B-Tree of degree 3.

<T S U T D C I N R C>

Delete : C



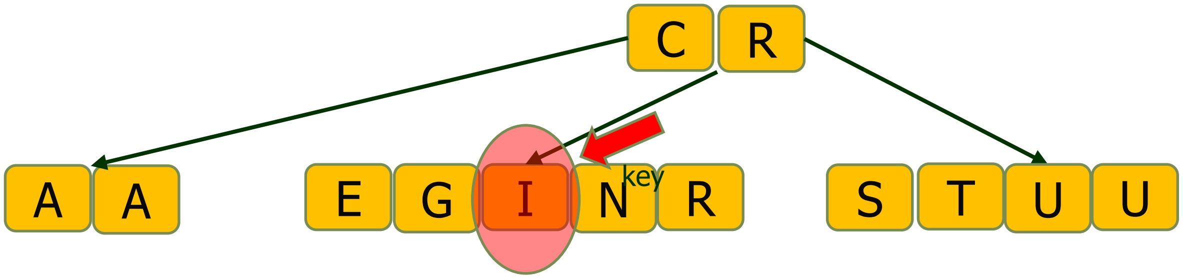
B Tree (Deletion)

Example 1:

Perform the deletion operation with the following data sequentially on the given B-Tree of degree 3.

<T S U T D C I N R C>

Delete : I



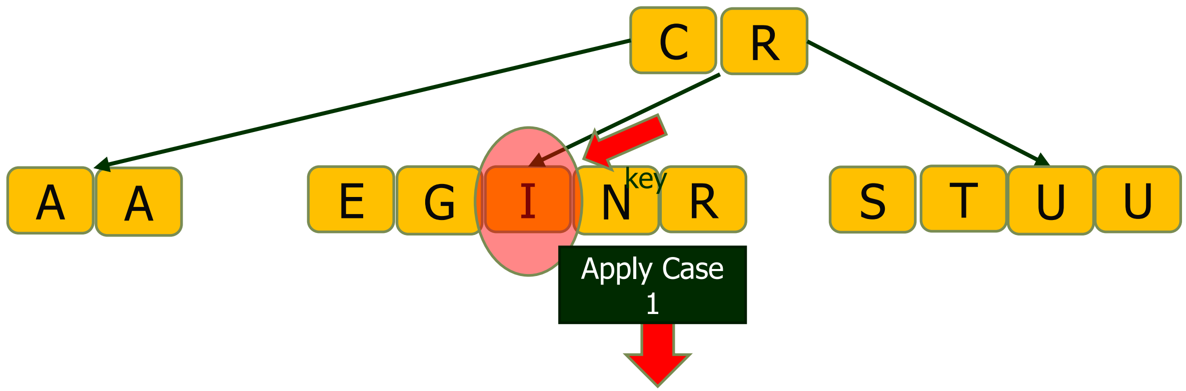
B Tree (Deletion)

Example 1:

Perform the deletion operation with the following data sequentially on the given B-Tree of degree 3.

<T S U T D C I N R C>

Delete : I



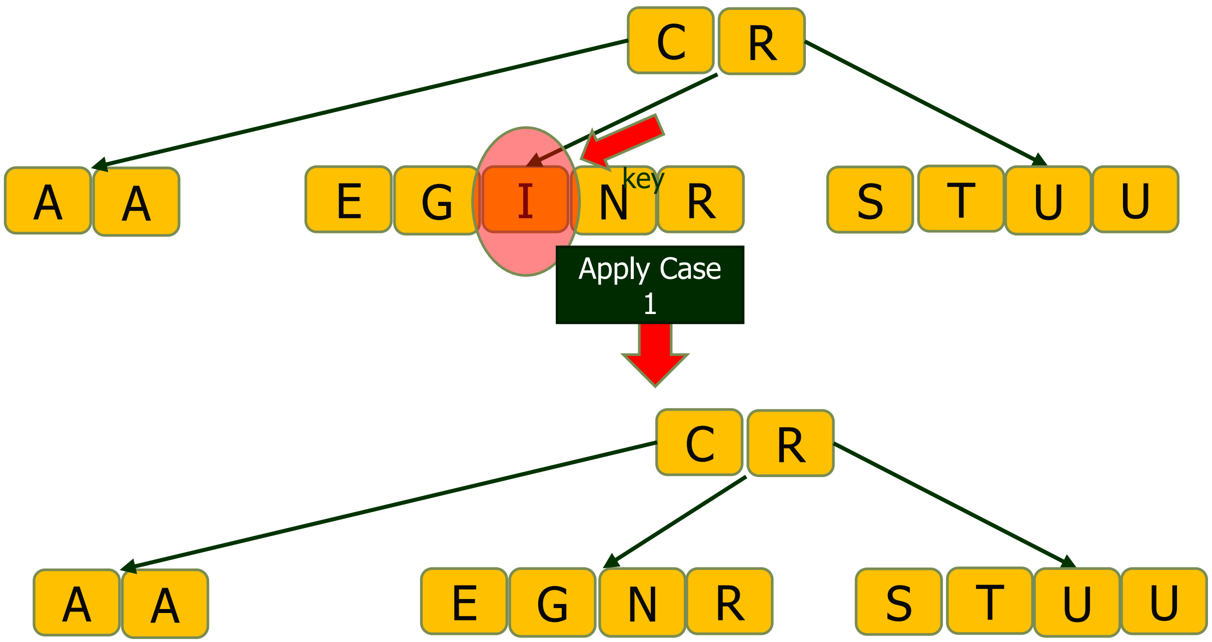
B Tree (Deletion)

Example 1:

Perform the deletion operation with the following data sequentially on the given B-Tree of degree 3.

<T S U T D C I N R C>

Delete : I



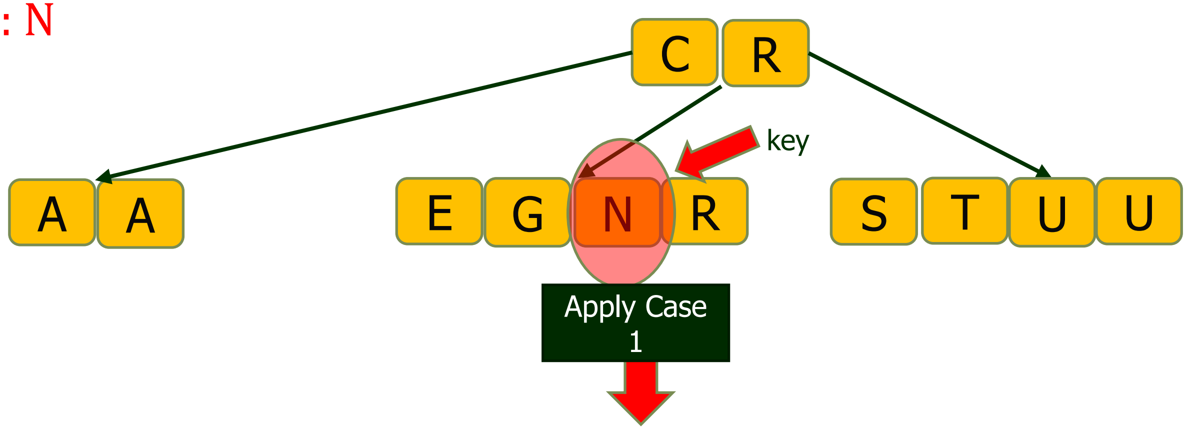
B Tree (Deletion)

Example 1:

Perform the deletion operation with the following data sequentially on the given B-Tree of degree 3.

<T S U T D C I N R C>

Delete : N



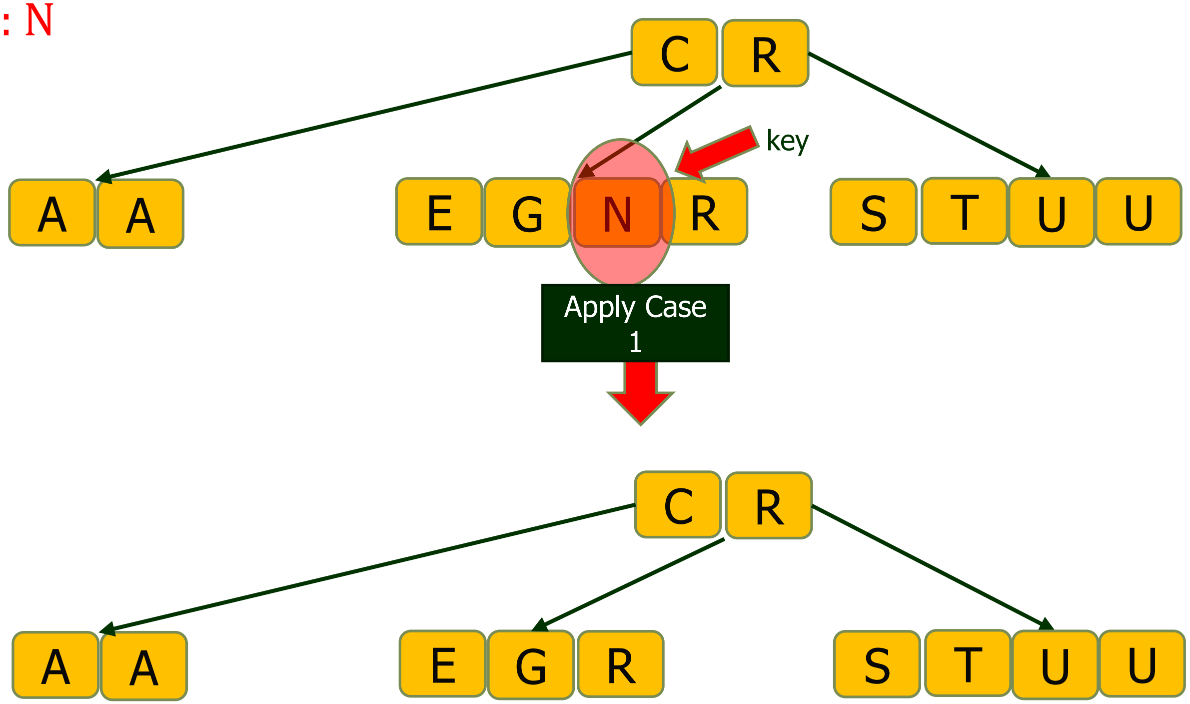
B Tree (Deletion)

Example 1:

Perform the deletion operation with the following data sequentially on the given B-Tree of degree 3.

<T S U T D C I N R C>

Delete : N



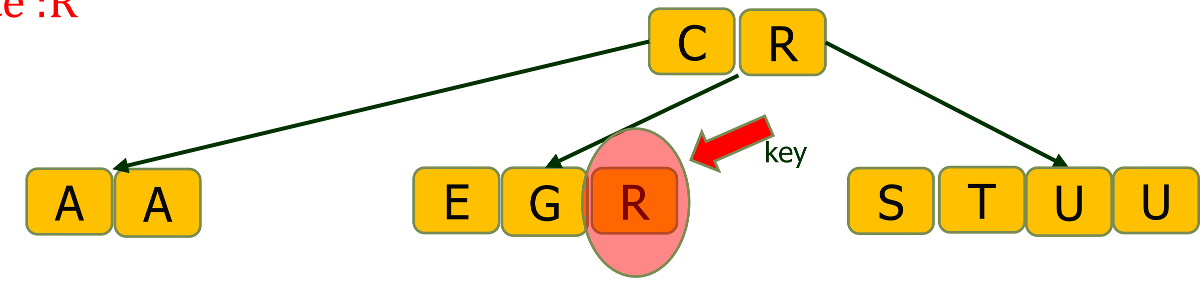
B Tree (Deletion)

Example 1:

Perform the deletion operation with the following data sequentially on the given B-Tree of degree 3.

<T S U T D C I N R C>

Delete :R



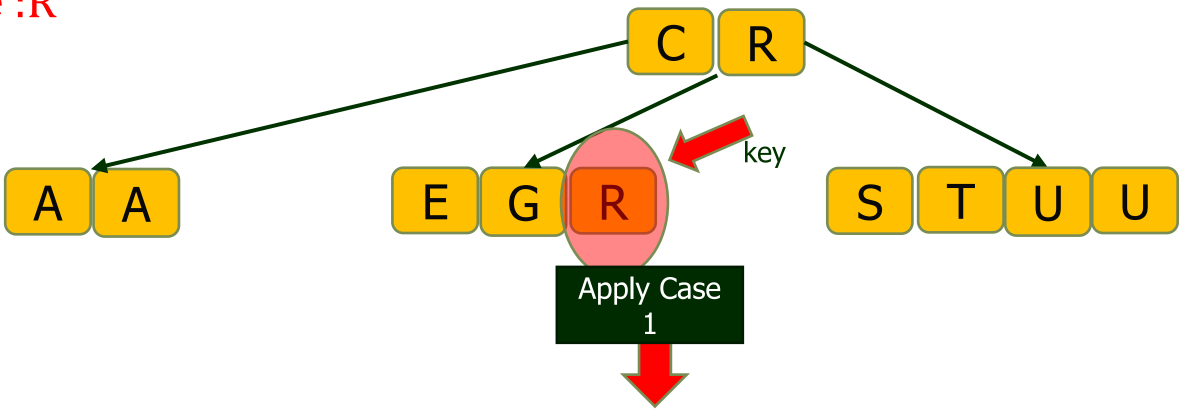
B Tree (Deletion)

Example 1:

Perform the deletion operation with the following data sequentially on the given B-Tree of degree 3.

<T S U T D C I N R C>

Delete :R



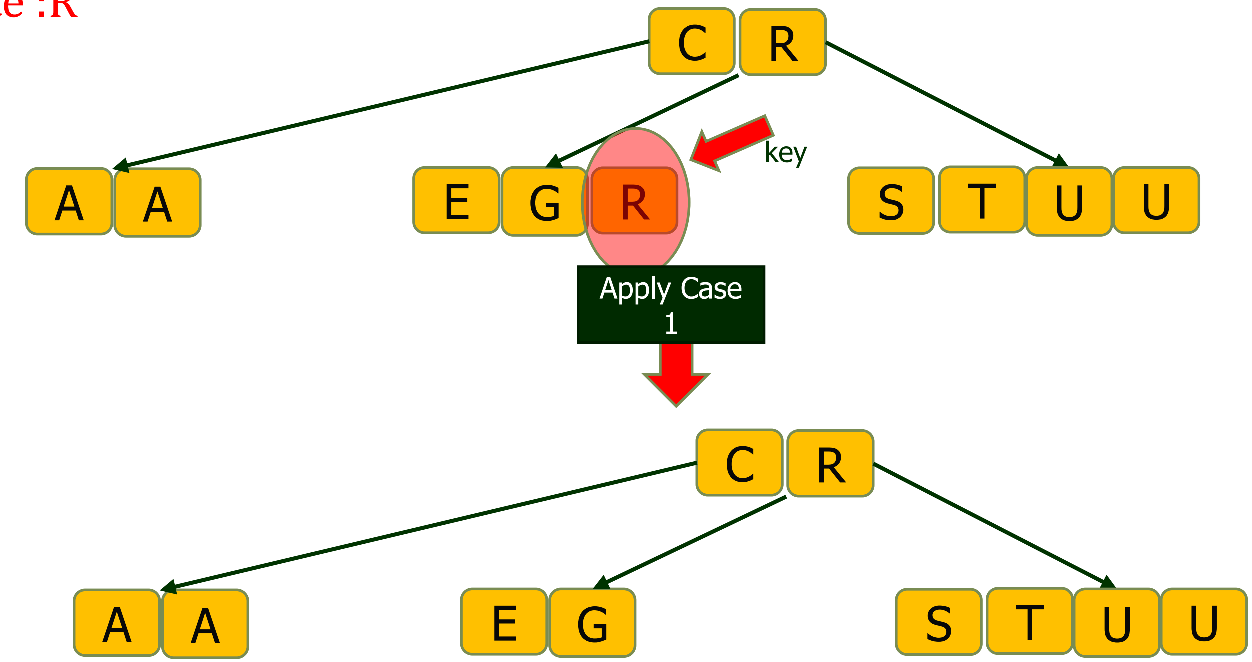
B Tree (Deletion)

Example 1:

Perform the deletion operation with the following data sequentially on the given B-Tree of degree 3.

<T S U T D C I N R C>

Delete :R



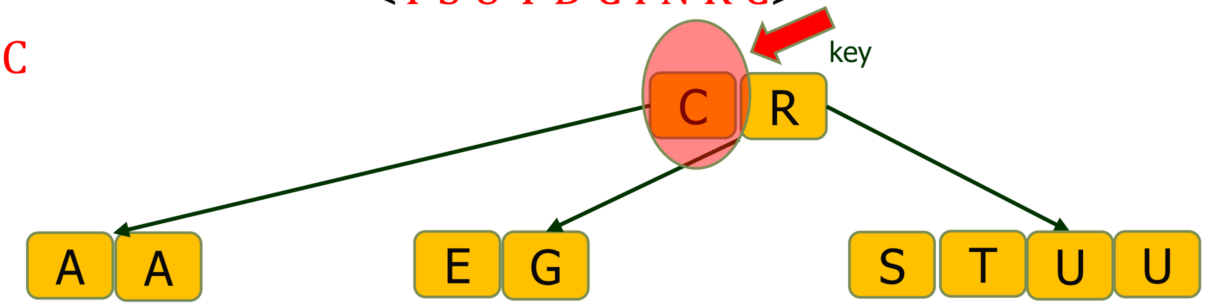
B Tree (Deletion)

Example 1:

Perform the deletion operation with the following data sequentially on the given B-Tree of degree 3.

<T S U T D C I N R C>

Delete :C



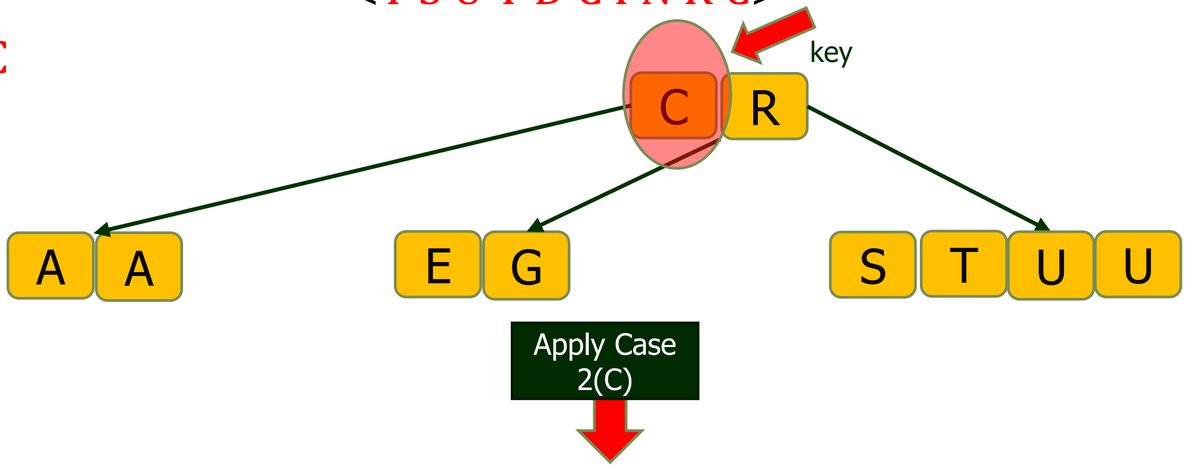
B Tree (Deletion)

Example 1:

Perform the deletion operation with the following data sequentially on the given B-Tree of degree 3.

<T S U T D C I N R C>

Delete :C



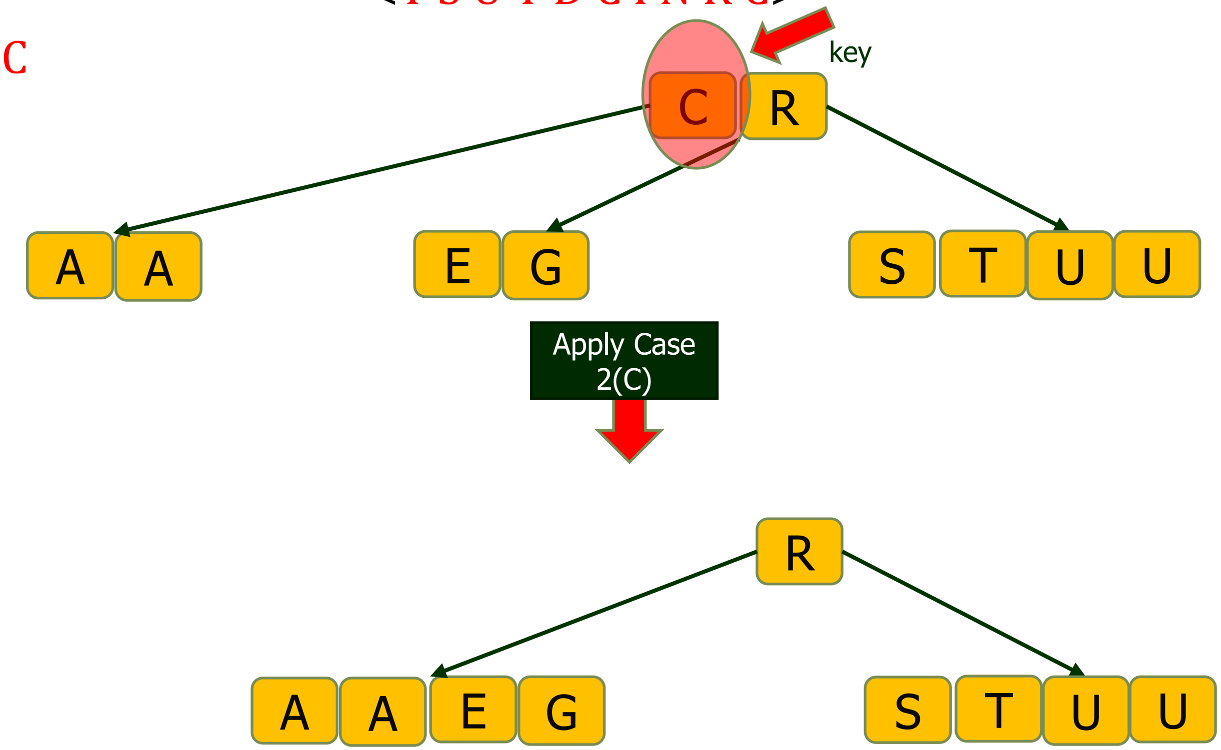
B Tree (Deletion)

Example 1:

Perform the deletion operation with the following data sequentially on the given B-Tree . of degree 3

<T S U T D C I N R C>

Delete :C



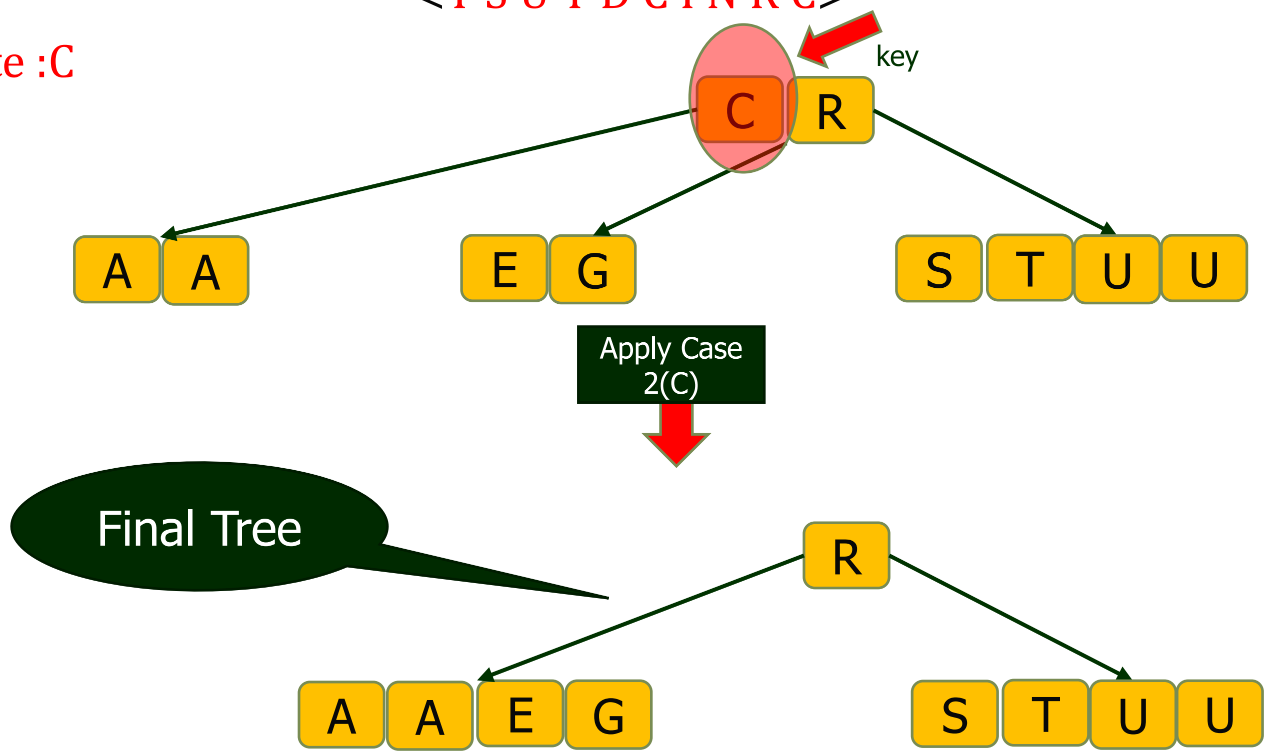
B Tree (Deletion)

Example 1:

Perform the deletion operation with the following data sequentially on the given B-Tree of degree 3.

<T S U T D C I N R C>

Delete :C

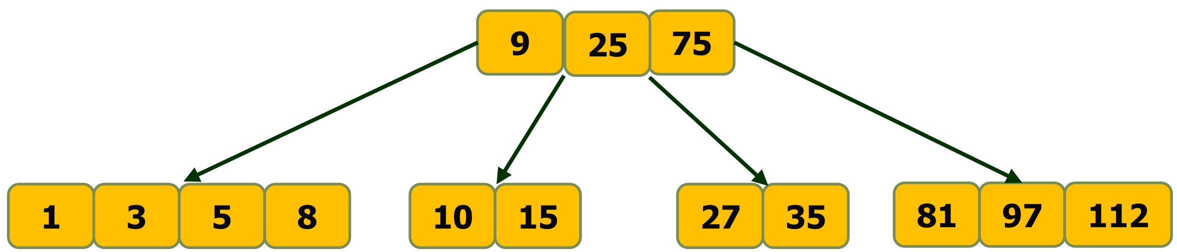


B Tree (Deletion)

Example 2:

Perform the deletion operation with the following data sequentially on the given B-Tree of order 5.

<25 75 27 5 15 81 97 112 10 >

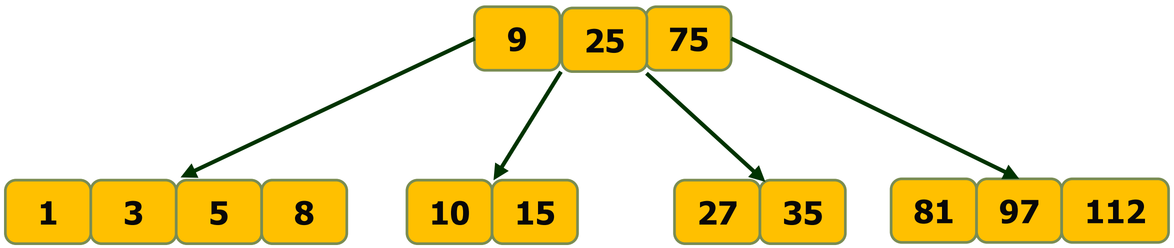


B Tree (Deletion)

Example 2:

Perform the deletion operation with the following data sequentially on the given B-Tree of order 5.

<25 75 27 5 15 81 97 112 10 >



If order =5 then

Maximum Key=m-1 = 5-1=4

Minimum Key=$\left\lceil \frac{m}{2} \right\rceil - 1$= 3-1=2

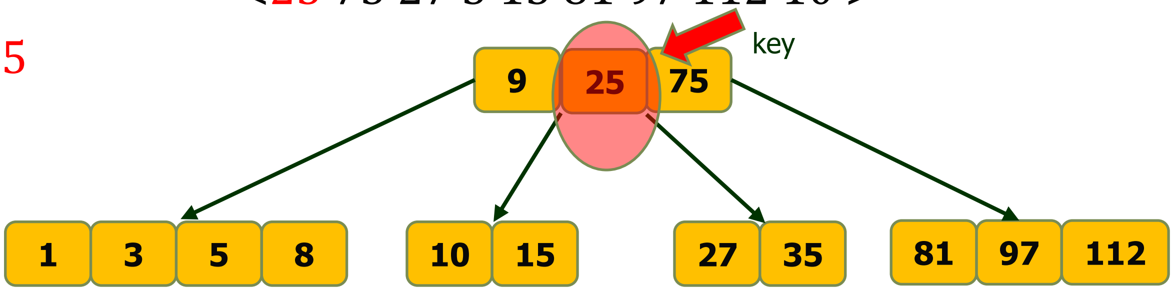
B Tree (Deletion)

Example 2:

Perform the deletion operation with the following data sequentially on the given B-Tree of order 5.

<25 75 27 5 15 81 97 112 10 >

Delete :25



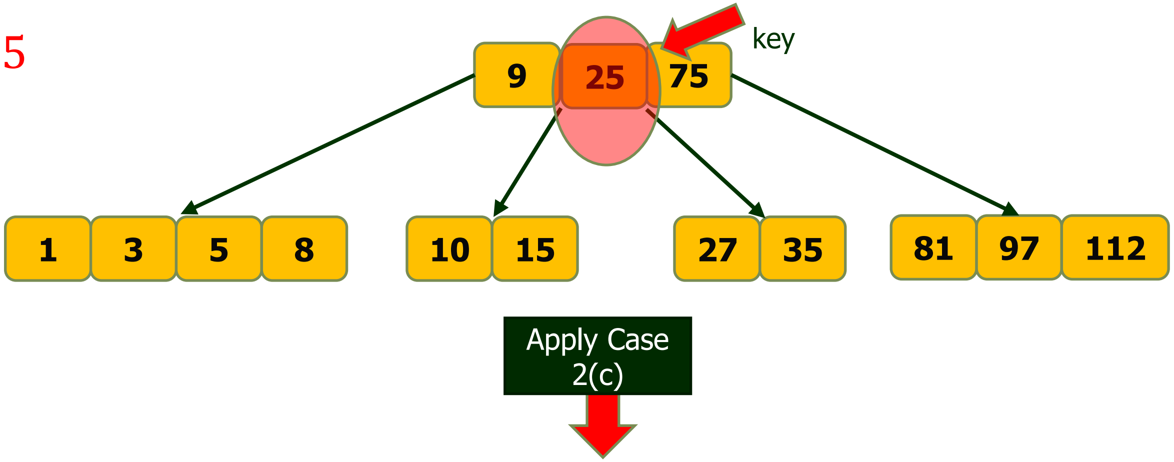
B Tree (Deletion)

Example 2:

Perform the deletion operation with the following data sequentially on the given B-Tree of order 5.

<25 75 27 5 15 81 97 112 10 >

Delete :25



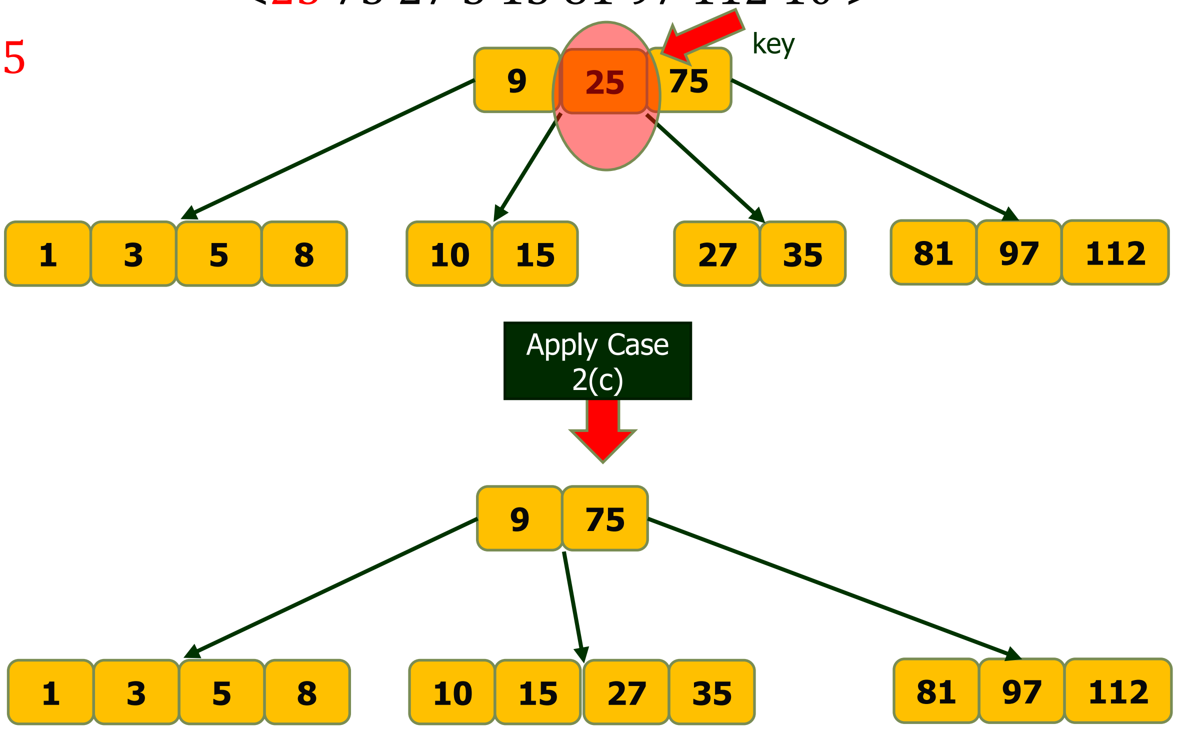
B Tree (Deletion)

Example 2:

Perform the deletion operation with the following data sequentially on the given B-Tree of order 5.

<25 75 27 5 15 81 97 112 10 >

Delete :25



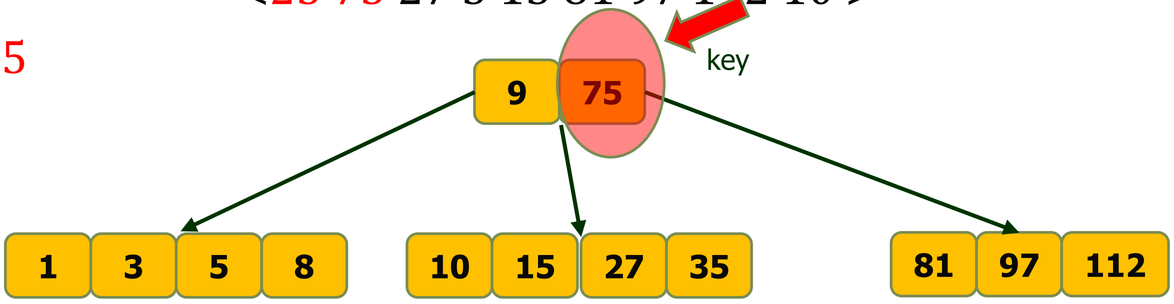
B Tree (Deletion)

Example 2:

Perform the deletion operation with the following data sequentially on the given B-Tree of order 5.

<25 75 27 5 15 81 97 112 10 >

Delete :75



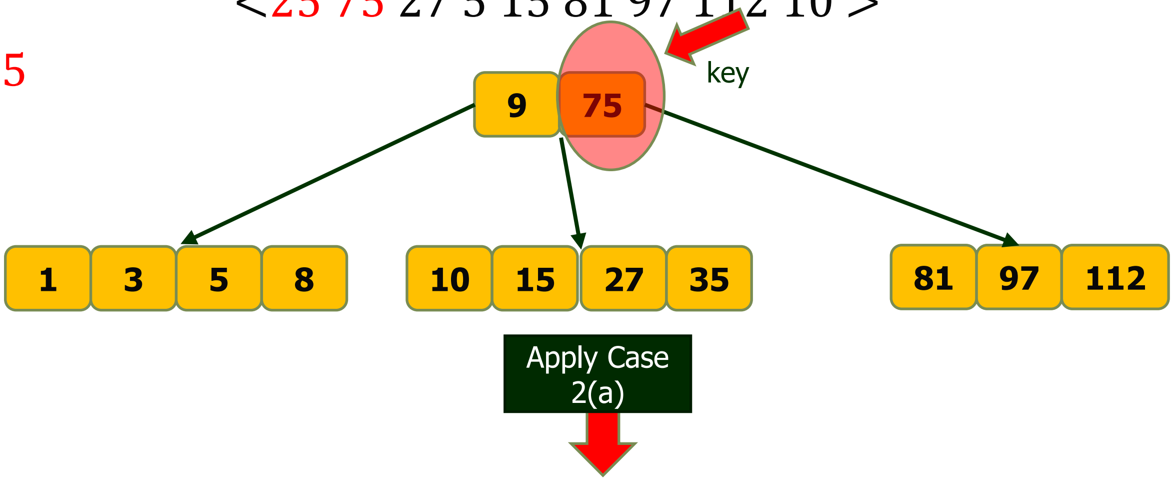
B Tree (Deletion)

Example 2:

Perform the deletion operation with the following data sequentially on the given B-Tree of order 5.

<25 75 27 5 15 81 97 112 10 >

Delete :75



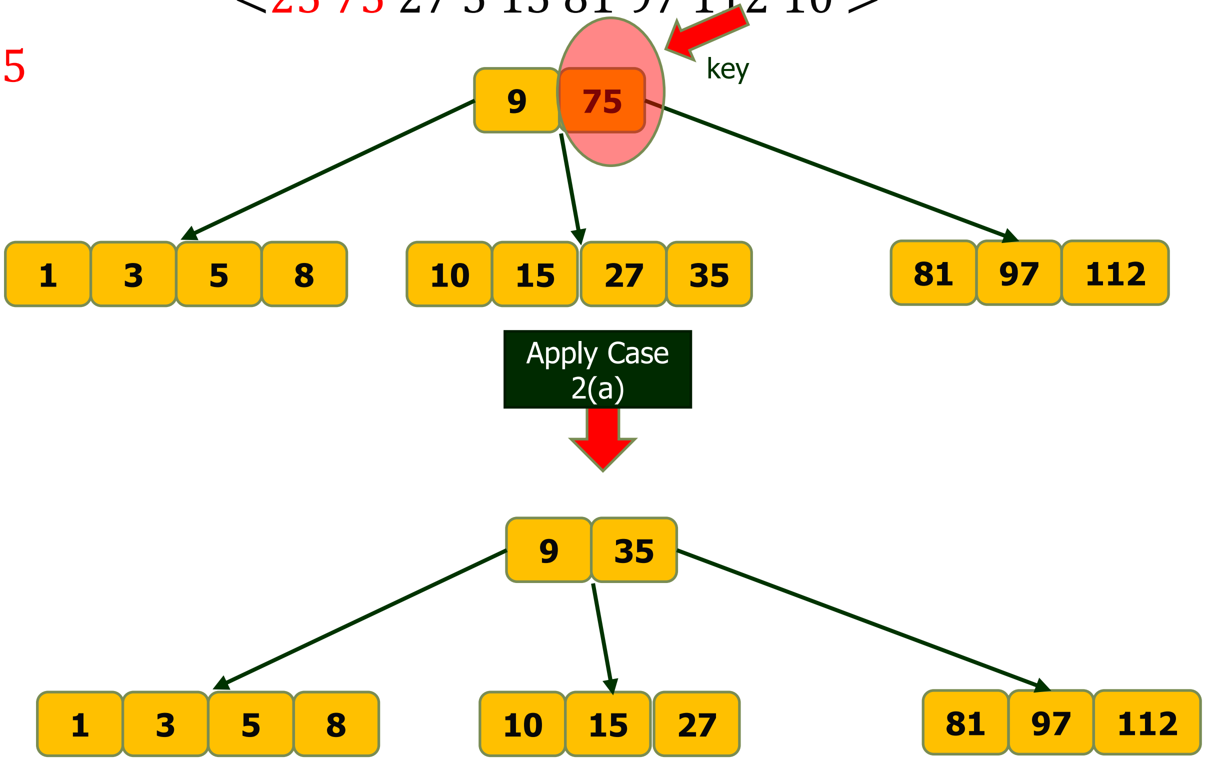
B Tree (Deletion)

Example 2:

Perform the deletion operation with the following data sequentially on the given B-Tree of order 5.

<25 75 27 5 15 81 97 112 10 >

Delete :75



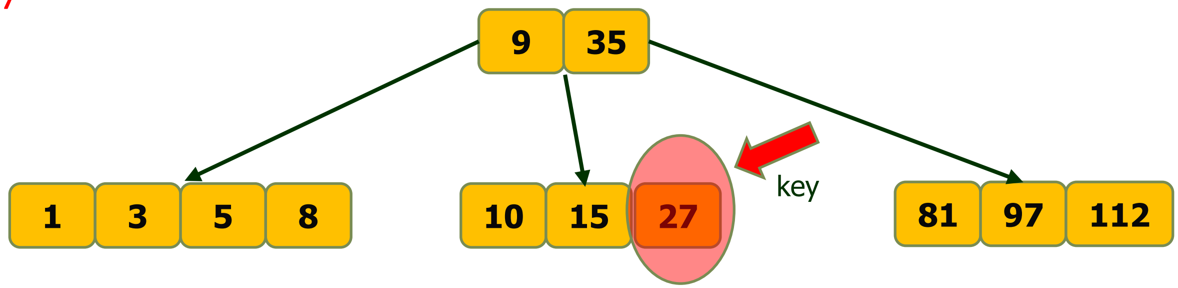
B Tree (Deletion)

Example 2:

Perform the deletion operation with the following data sequentially on the given B-Tree of order 5.

<25 75 27 5 15 81 97 112 10 >

Delete :27



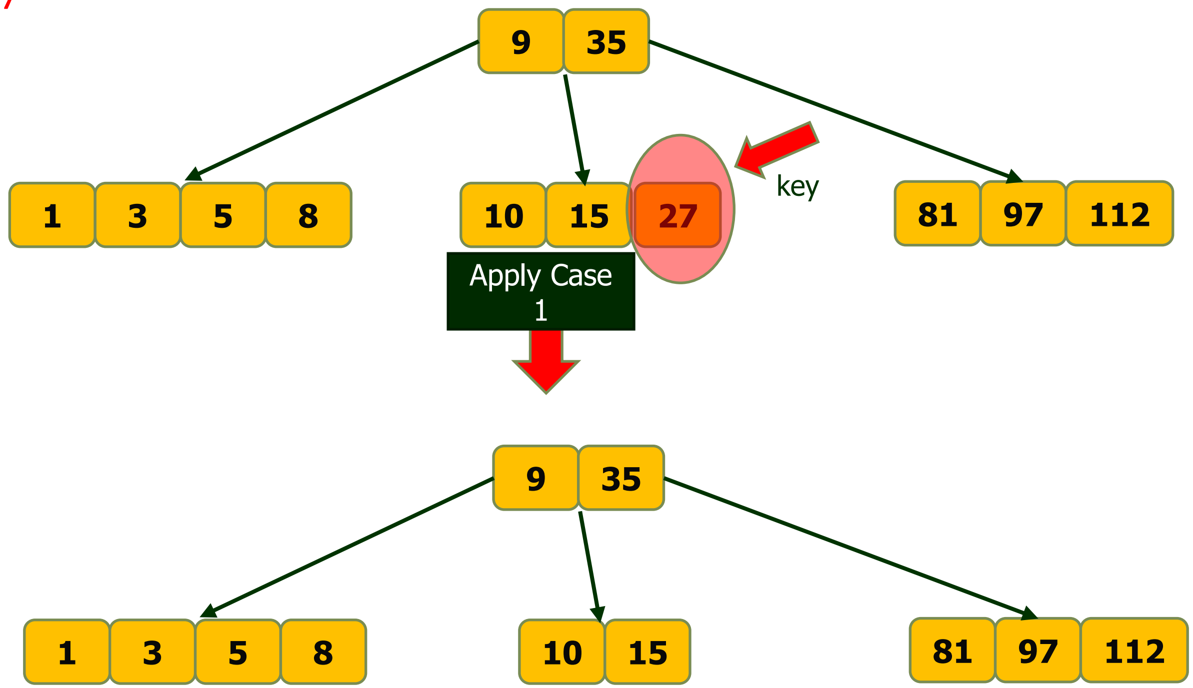
B Tree (Deletion)

Example 2:

Perform the deletion operation with the following data sequentially on the given B-Tree of order 5.

<25 75 27 5 15 81 97 112 10 >

Delete :27



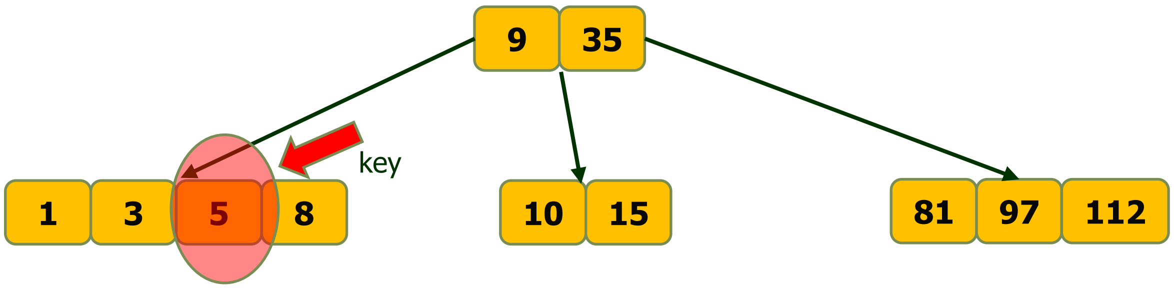
B Tree (Deletion)

Example 2:

Perform the deletion operation with the following data sequentially on the given B-Tree of order 5.

<25 75 27 5 15 81 97 112 10 >

Delete :5



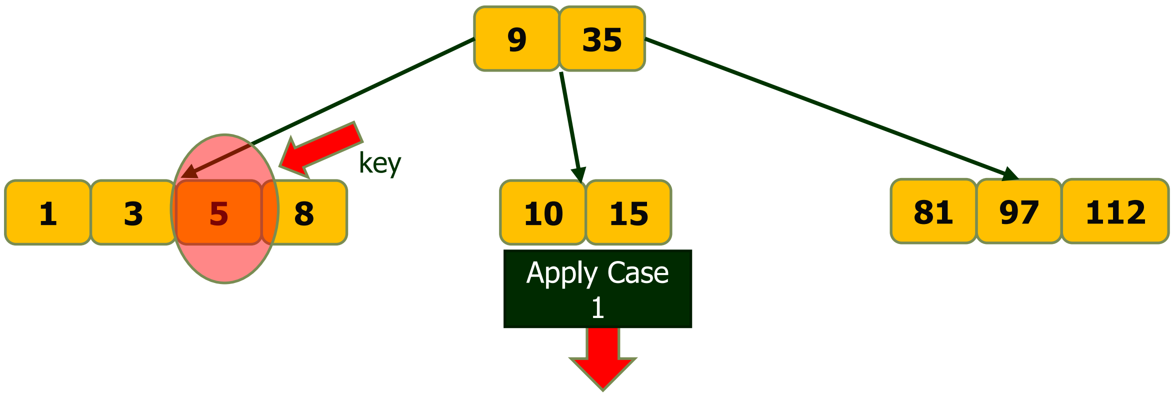
B Tree (Deletion)

Example 2:

Perform the deletion operation with the following data sequentially on the given B-Tree of order 5.

<25 75 27 5 15 81 97 112 10 >

Delete :5



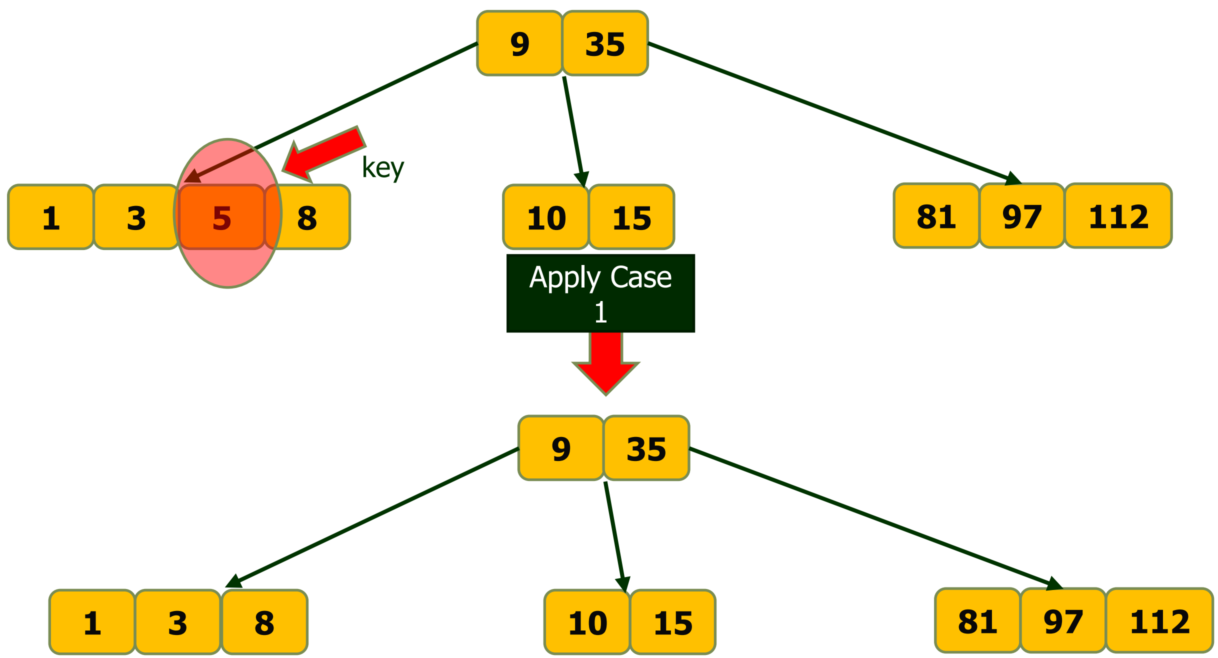
B Tree (Deletion)

Example 2:

Perform the deletion operation with the following data sequentially on the given B-Tree of order 5.

<25 75 27 5 15 81 97 112 10 >

Delete :5



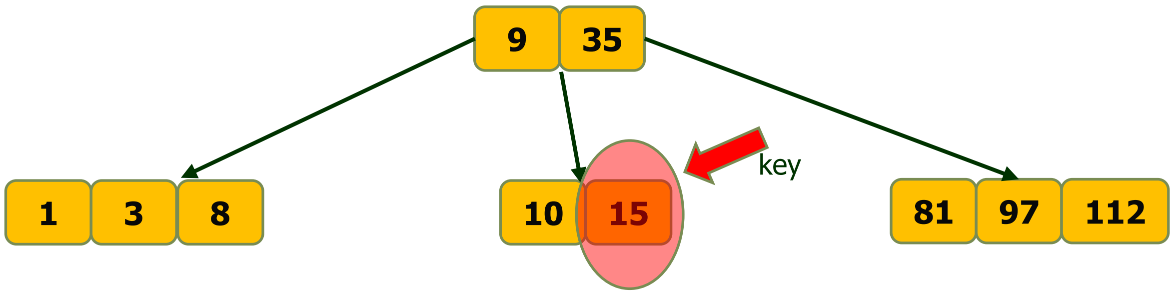
B Tree (Deletion)

Example 2:

Perform the deletion operation with the following data sequentially on the given B-Tree of order 5.

<25 75 27 5 15 81 97 112 10 >

Delete :15



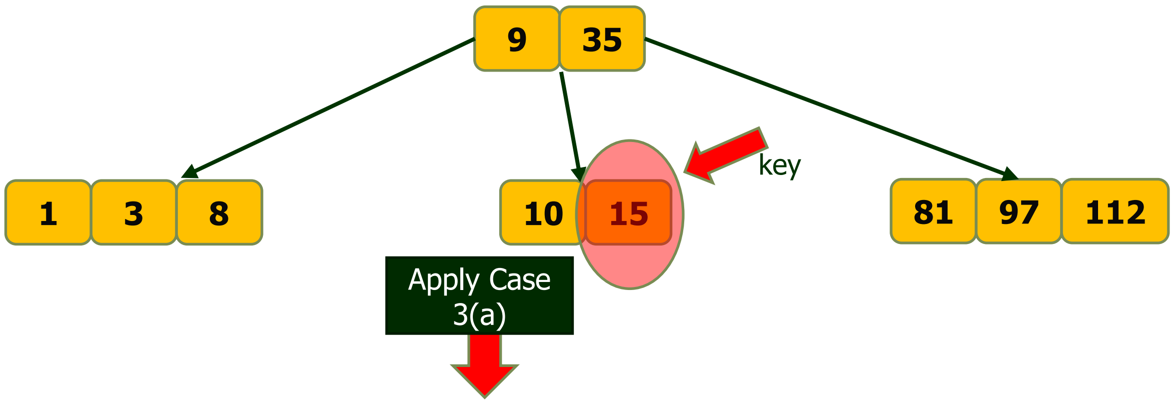
B Tree (Deletion)

Example 2:

Perform the deletion operation with the following data sequentially on the given B-Tree of order 5.

<25 75 27 5 15 81 97 112 10 >

Delete :15



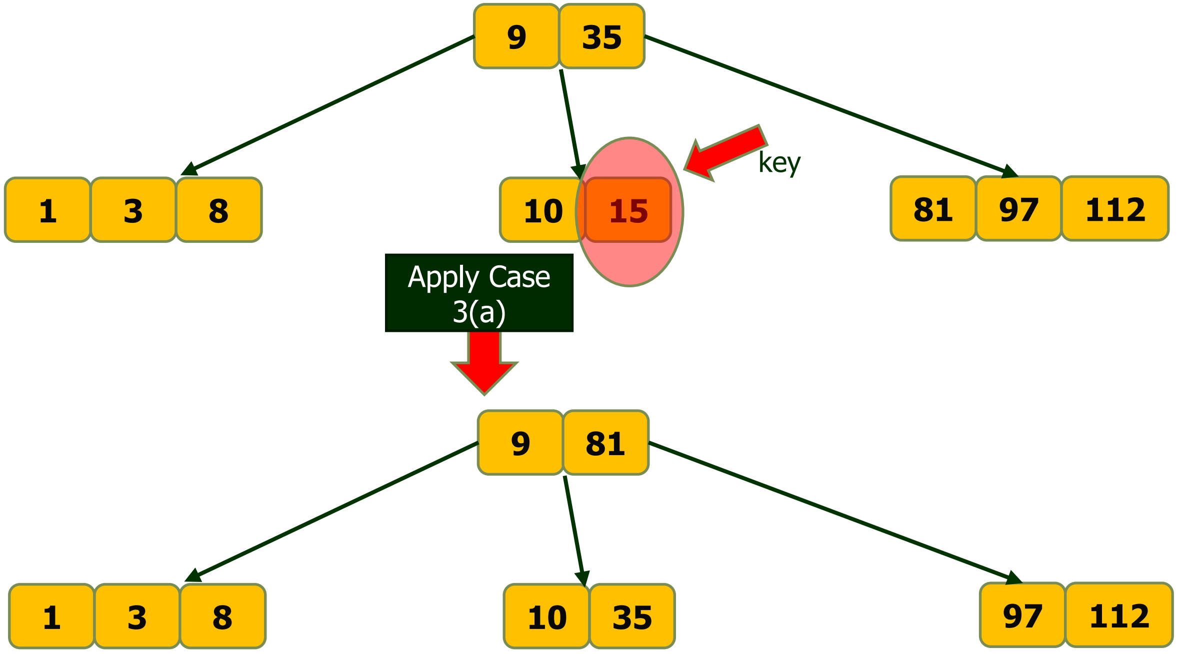
B Tree (Deletion)

Example 2:

Perform the deletion operation with the following data sequentially on the given B-Tree of order 5.

<25 75 27 5 15 81 97 112 10 >

Delete :15



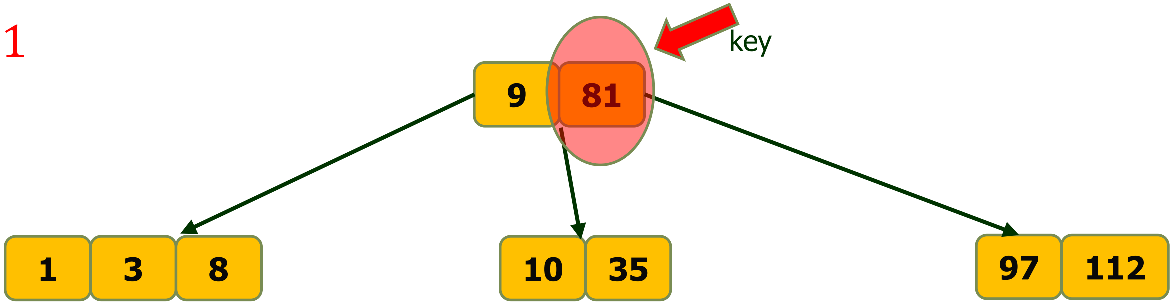
B Tree (Deletion)

Example 2:

Perform the deletion operation with the following data sequentially on the given B-Tree of order 5.

<25 75 27 5 15 81 97 112 10 >

Delete :81



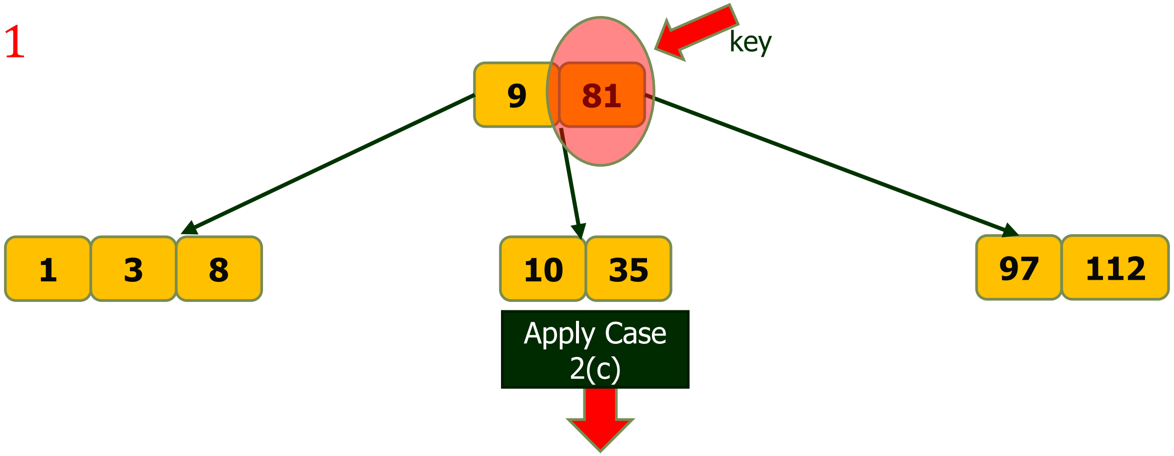
B Tree (Deletion)

Example 2:

Perform the deletion operation with the following data sequentially on the given B-Tree of order 5.

<25 75 27 5 15 81 97 112 10 >

Delete :81



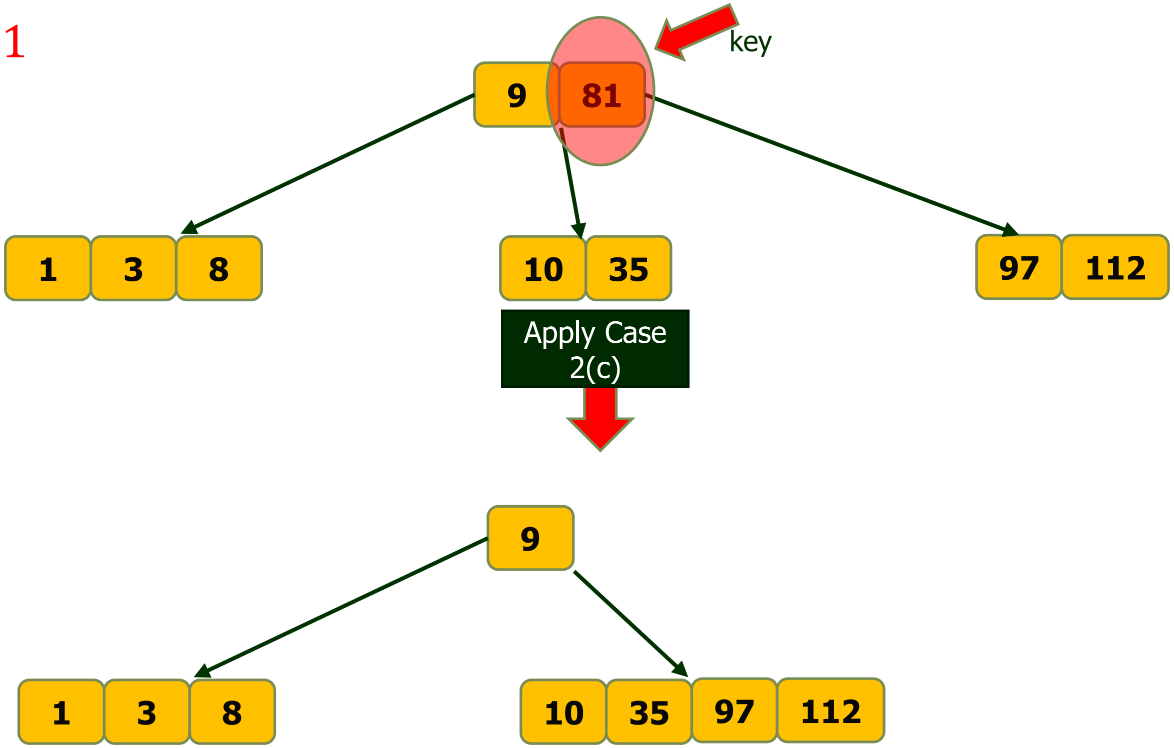
B Tree (Deletion)

Example 2:

Perform the deletion operation with the following data sequentially on the given B-Tree of order 5.

<25 75 27 5 15 81 97 112 10 >

Delete :81



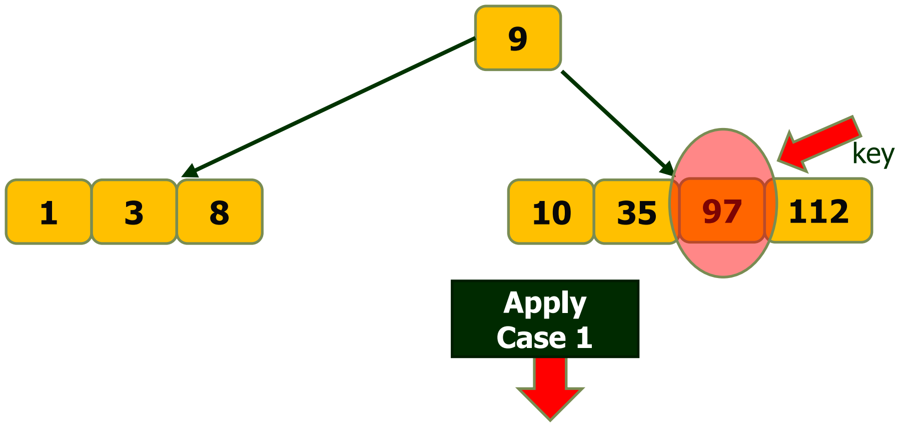
B Tree (Deletion)

Example 2:

Perform the deletion operation with the following data sequentially on the given B-Tree of order 5.

<25 75 27 5 15 81 97 112 10 >

Delete :97



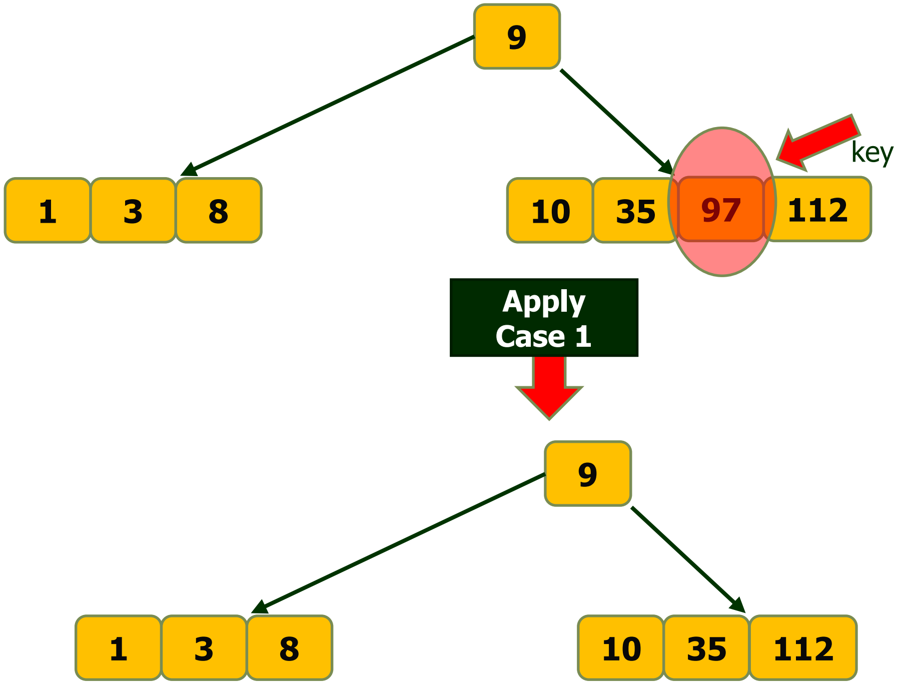
B Tree (Deletion)

Example 2:

Perform the deletion operation with the following data sequentially on the given B-Tree of order 5.

<25 75 27 5 15 81 97 112 10 >

Delete :97



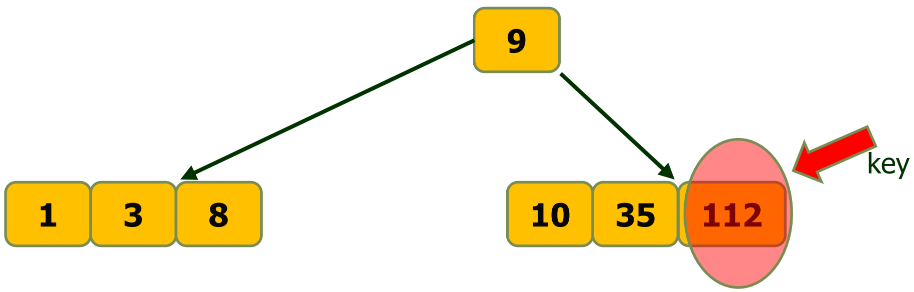
B Tree (Deletion)

Example 2:

Perform the deletion operation with the following data sequentially on the given B-Tree of order 5.

<25 75 27 5 15 81 97 112 10 >

Delete :112



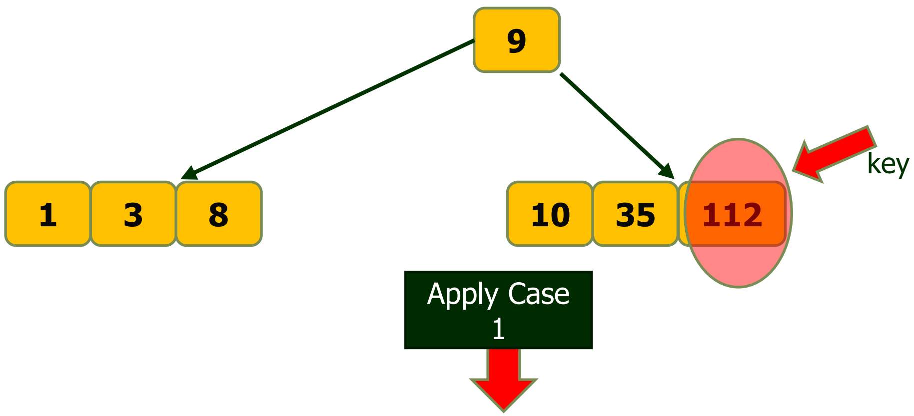
B Tree (Deletion)

Example 2:

Perform the deletion operation with the following data sequentially on the given B-Tree of order 5.

<25 75 27 5 15 81 97 112 10 >

Delete :112



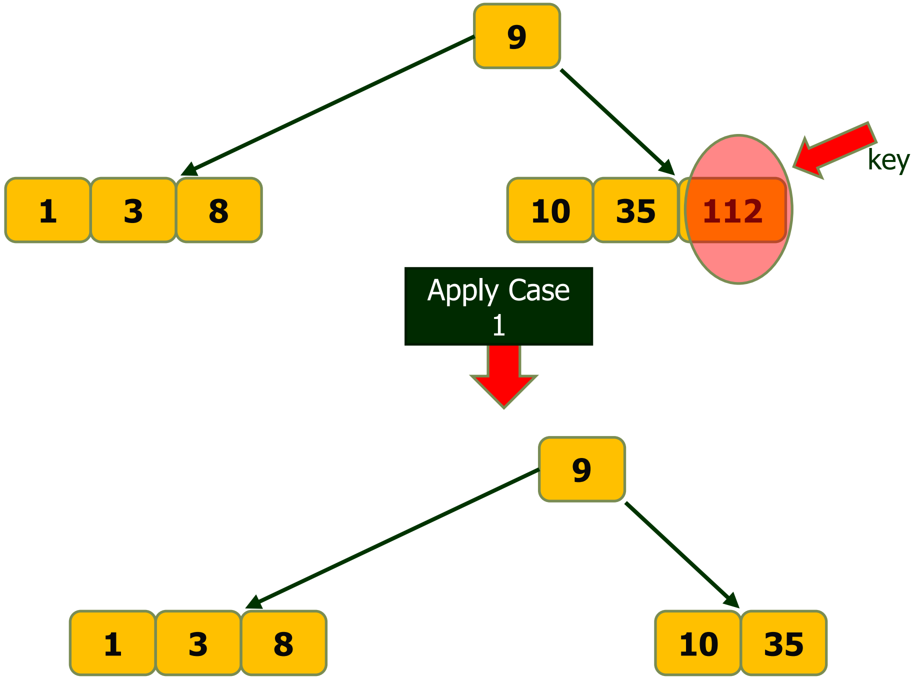
B Tree (Deletion)

Example 2:

Perform the deletion operation with the following data sequentially on the given B-Tree of order 5.

<25 75 27 5 15 81 97 112 10 >

Delete :112



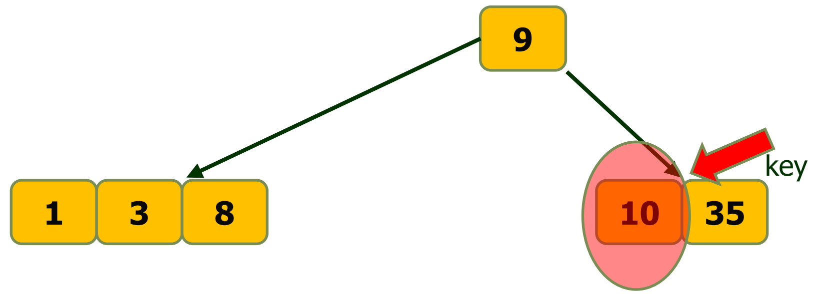
B Tree (Deletion)

Example 2:

Perform the deletion operation with the following data sequentially on the given B-Tree of order 5.

<25 75 27 5 15 81 97 112 10 >

Delete :10



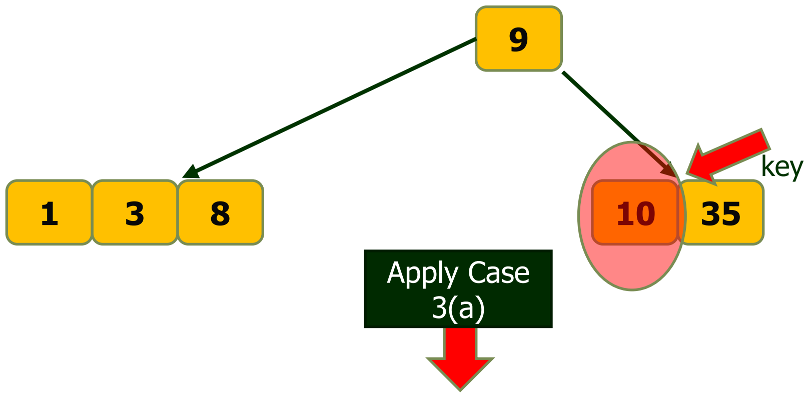
B Tree (Deletion)

Example 2:

Perform the deletion operation with the following data sequentially on the given B-Tree of order 5.

<25 75 27 5 15 81 97 112 10 >

Delete :10



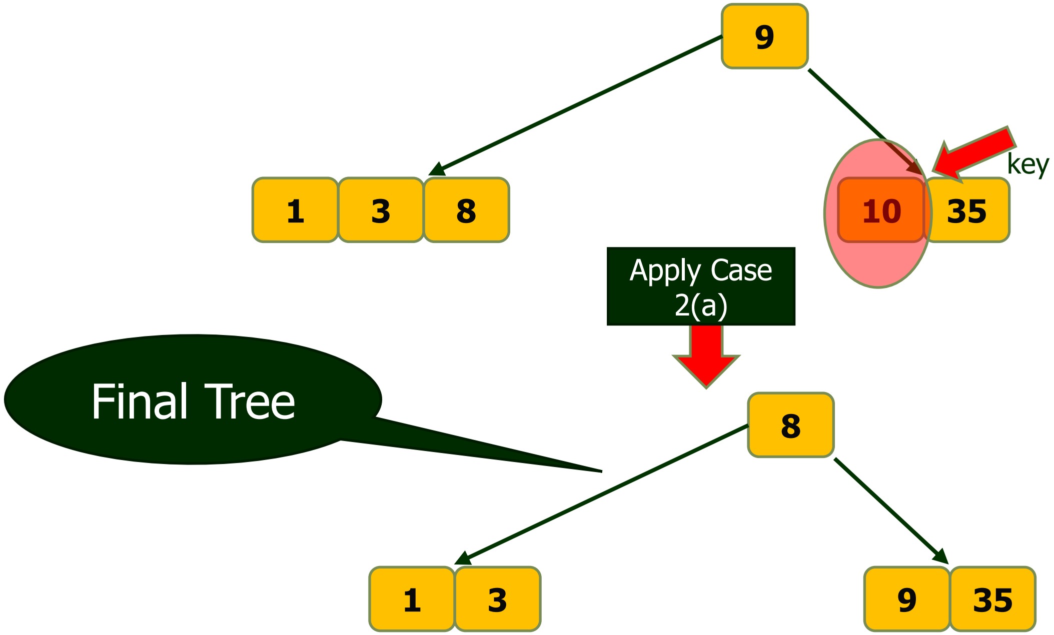
B Tree (Deletion)

Example 2:

Perform the deletion operation with the following data sequentially on the given B-Tree of order 5.

<25 75 27 5 15 81 97 112 10 >

Delete :10



B Tree (Deletion)

Theorem:

If $n \geq 1$, then for any n- key B-Tree T of height h and minimum degree $t \geq 2$, then $h \leq \log_t \frac{n+1}{2}$.

Proof:

The root of B-Tree contains at least one keys and all other nodes contain at least t-1 keys.

Thus T, whose height is h, has at least 2 nodes at depth 1. 2t nodes at depth 2, at least $2t^2$ node at depth 3, and so on until it has at least $2t^{h-1}$nodes.

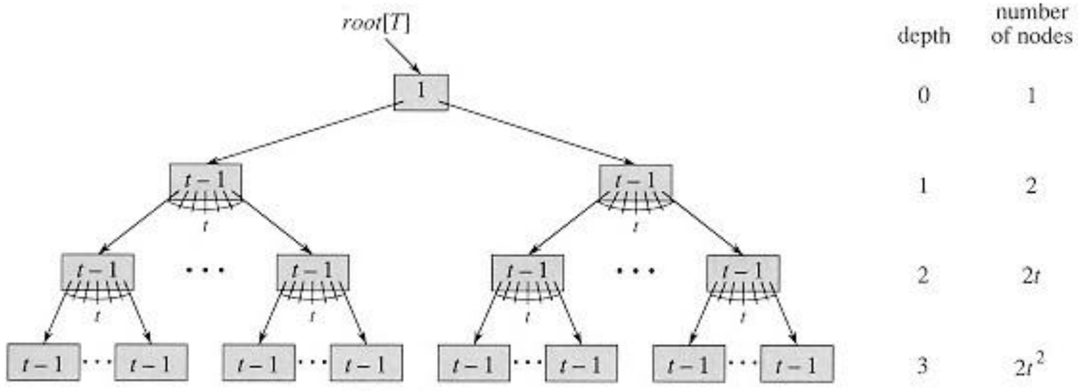


Figure A B-tree of height 3 containing a minimum possible number of keys. Shown inside each node x is n[x].

B Tree (Deletion)

Hence the total number of elements at depth are

1 element at depth 0, 2(t-1) elements at depth 1,

2t(t-1) elements at depth 2, $2t^2(t-1)$ elements at depth 3, and so on.

Hence,

$$n \geq 1 + 2(t-1) + 2t(t-1) + 2t^2(t-1) + \cdots + 2t^{h-1}(t-1)$$

$$n \geq 1 + (t-1)\sum_{i=1}^{h} 2t^{i-1}$$

$$n \geq 1 + 2(t-1)\frac{t^h - 1}{(t-1)}$$

$$n \geq 1 + 2t^h - 2$$

$$n \geq 2t^h - 1$$

$$n + 1 \geq 2t^h \implies t^h = \frac{n+1}{2}$$

Apply log both side

$$\log t^h = \log\frac{n+1}{2} \implies h = \log_t\left(\frac{n+1}{2}\right)$$ proved.

Thank u