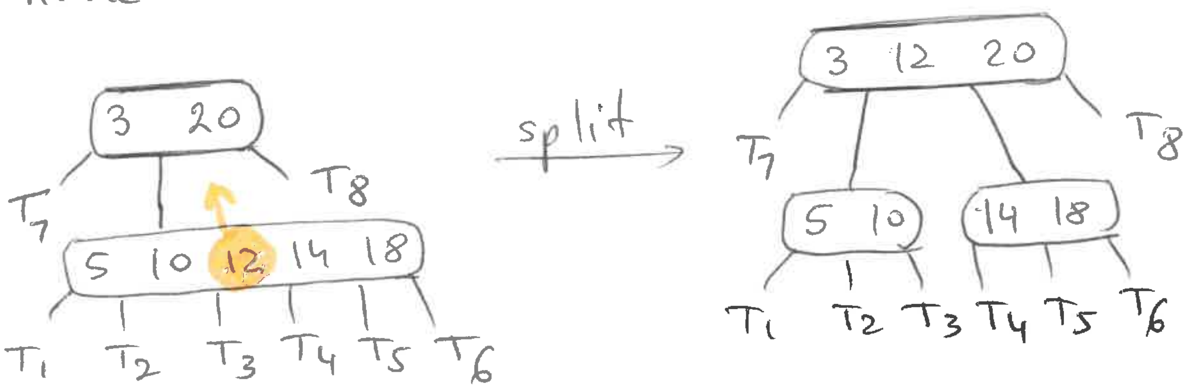
B-tree, insert operation

split operation – split a full node with 2t-1 Keys into two nodes with (t-1) Keys each, and move the median key to the parent node.

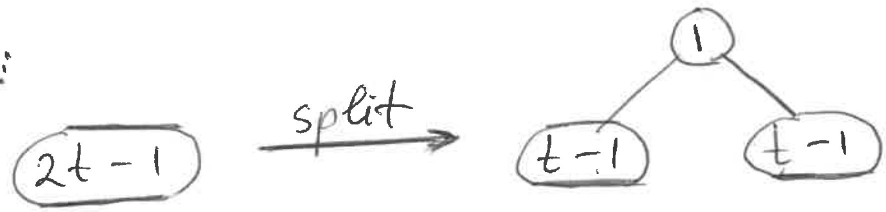
example

$t=3$

#Keys: 2..5



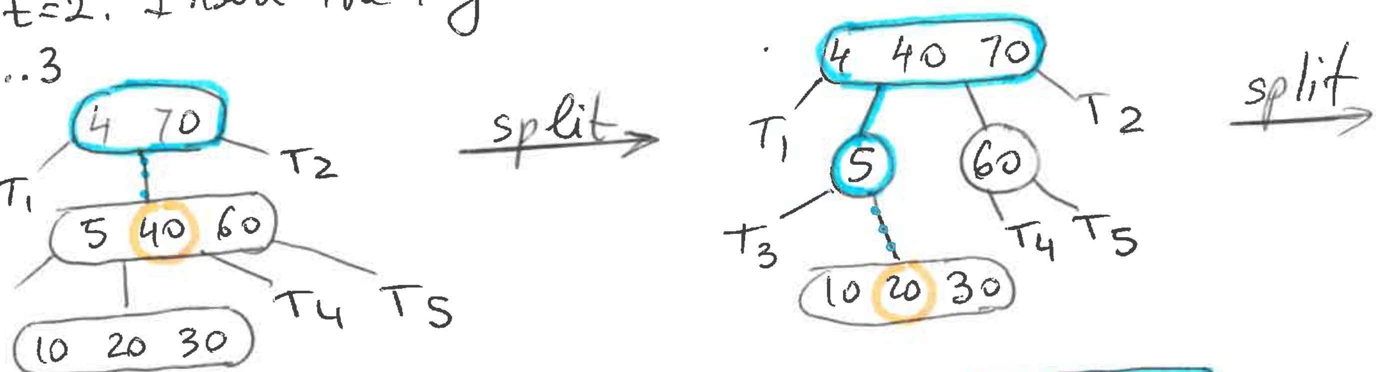
split:

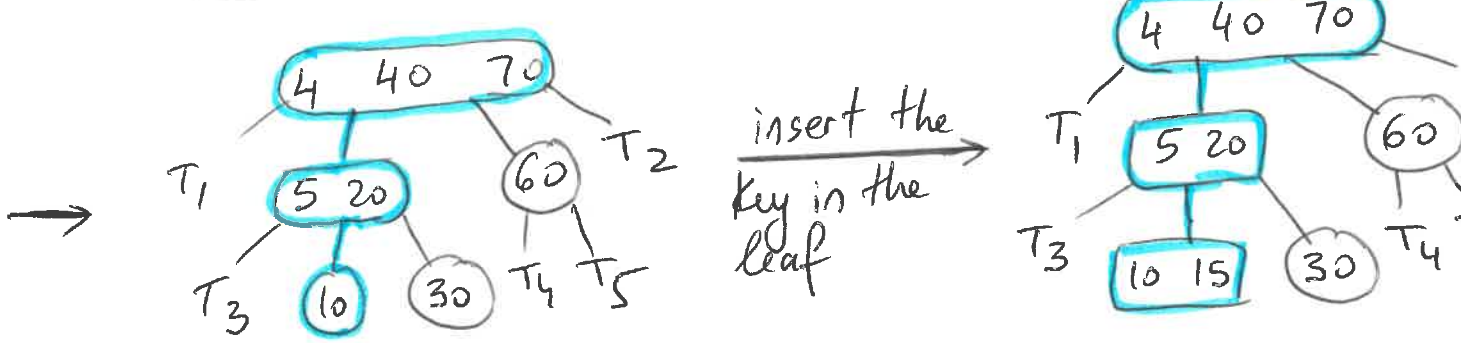


example

Let $t=2$. Insert the Key 15 into the B-tree below.

#Keys: 1..3

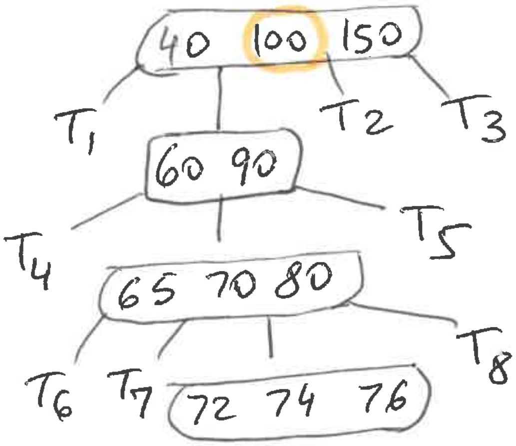




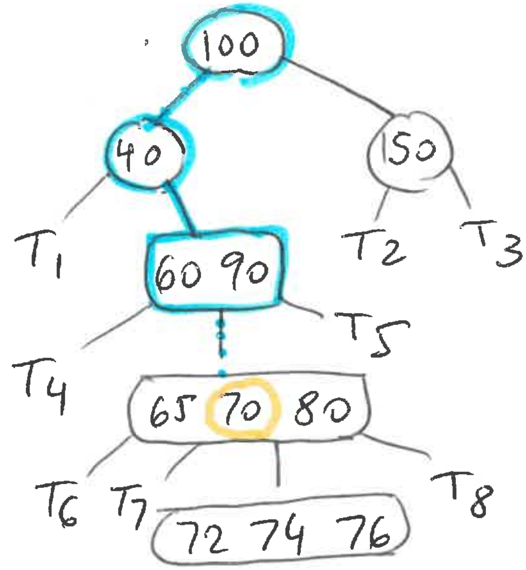
example

Let $t=2$. Insert the key 73 into the B-tree below.

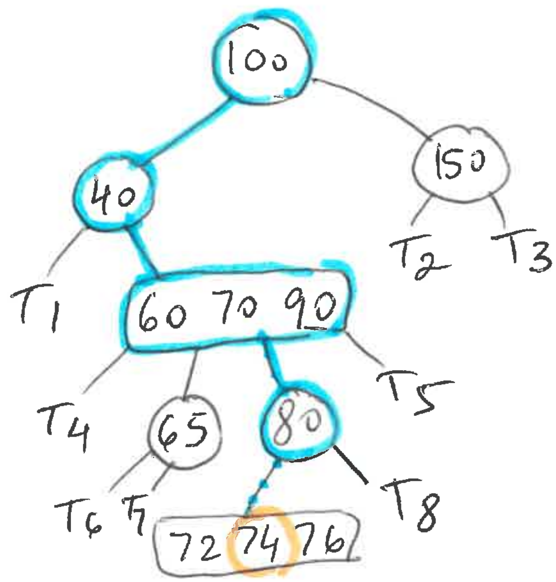
#keys: 1..3



split

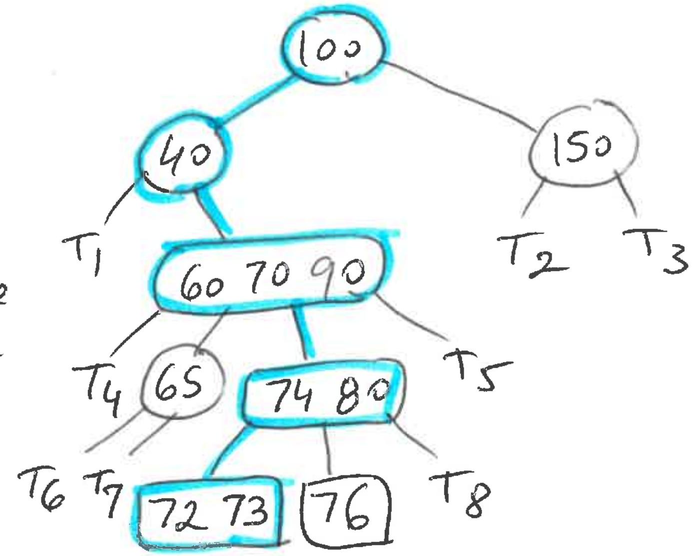


split



split

insert the key in the leaf



Splitting a node in the B-tree

Input: x – nonfull internal node (in main memory)
index i such that $\begin{cases} x.c_i \text{ is a full child of } x \\ x.c_i \text{ is in the main memory} \end{cases}$

Output: split node $x.c_i$ around its median key $x.c_i.key_t$

B-TREE-SPLIT-CHILD(x, i)

1. z = ALLOCATE-NODE()
2. y = $x.c_i$
3. z.leaf = y.leaf
4. z.n = t – 1

$\Theta(t)$

5. **for** j = 1 to t –1
6. $z.key_j = y.key_{j+t}$

$\Theta(t)$

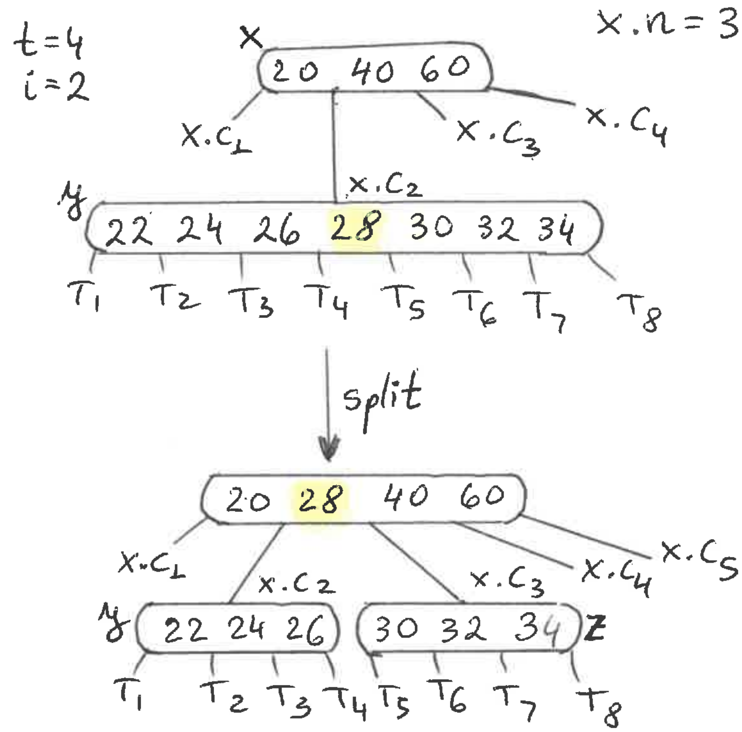
7. **if not** y.leaf
8. **for** j = 1 to t
9. $z.c_j = y.c_{j+t}$
10. y.n = t – 1

$\Theta(t)$

11. **for** j = x.n+1 **downto** i+1
12. $x.c_{j+1} = x.c_j$
13. $x.c_{i+1} = z$

$\Theta(t)$

14. **for** j = x.n **downto** i
15. $x.key_{j+1} = x.key_j$
16. $x.key_i = y.key_t$
17. x.n = x.n + 1
18. DISK-WRITE(y)
19. DISK-WRITE(z)
20. DISK-WRITE(x)



lines 5..6, j=1..3
$\begin{cases} z.key_1 = y.key_5 \\ z.key_2 = y.key_6 \\ z.key_3 = y.key_7 \end{cases}$

lines 7..9, j=1..4
$\begin{cases} z.c_1 = y.c_5 \\ z.c_2 = y.c_6 \\ z.c_3 = y.c_7 \\ z.c_4 = y.c_8 \end{cases}$

lines 11..12, j=4..3
$\begin{cases} x.c_5 = x.c_4 \\ x.c_4 = x.c_3 \end{cases}$
$x.c_3 = z$

lines 14..15, j=3..2
$\begin{cases} x.key_4 = x.key_3 \\ x.key_3 = x.key_2 \end{cases}$
$x.key_2 = y.key_4$

RT = $\Theta(t)$

$\Theta(1)$ disk operations

If the root node r is <u>full</u>, then split r and a new node s becomes the root.

```
B-TREE-INSERT(T, k)
r = T.root
if r.n == 2t – 1
     s = ALLOCATE-NODE()
     T.root = s
      s.leaf = FALSE
      s.n = 0
      s.c_1 = r
      B-TREE-SPLIT-CHILD(s, 1)        Θ(t)
      B-TREE-INSERT-NONFULL(s, k)
else B-TREE-INSERT-NONFULL(r,k)
```

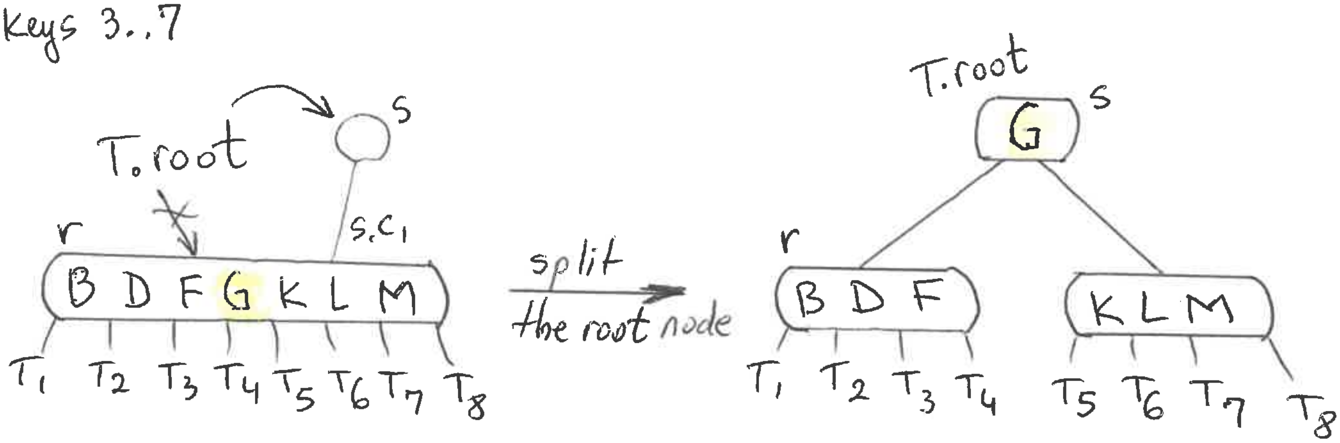
$RT = \Theta(t \cdot h) = \Theta(t \cdot \log_t n)$

if t = constant $\Rightarrow \boxed{RT = \Theta(\lg n)}$

<u>example</u>

$t = 4$

keys 3..7



Insert key k into the subtree rooted at the nonfull node x.

B-TREE-INSERT-NONFULL(x, k)

```
1. i = x.n
2. if x.leaf
```

$\Theta(t)$

```
3.     while i ≥ 1 and k < x.key_i
4.         x.key_{i+1} = x.key_i
5.         i = i – 1
```

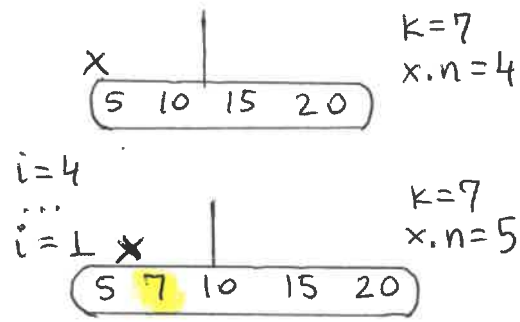
```
6.     x.key_{i+1} = k
7.     x.n = x.n + 1
8.     DISK-WRITE(x)
9. else
```

$\Theta(t)$

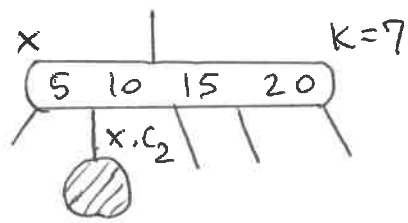
```
10.     while i ≥ 1 and k < x.key_i
11.         i = i – 1
```

```
12.     i = i + 1
13.     DISK-READ(x.c_i)
14.     if x.c_i.n == 2t – 1
15.         B-TREE-SPLIT-CHILD(x, i)
16.         if k > x.key_i
17.             i = i + 1
18.     B-TREE-INSERT-NONFULL(x.c_i, k)
```

x is a leaf node



x is NOT a leaf node



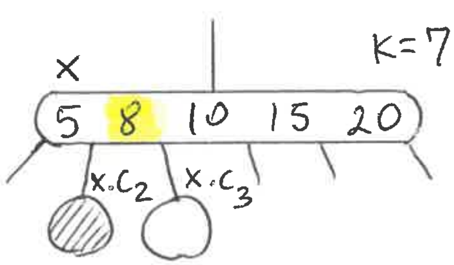
i=4

...

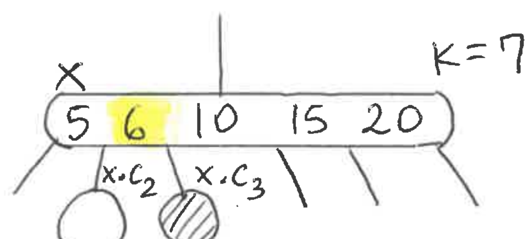
i=1 after while loop

line 12 ⇒ i=2

if $x.c_i$ is full (lines 14..17)



i=2



line 16: if $k > x.key_i$ ⇒ $i = i+1$

our case ⇒ $i = 3$

lines 1..17 have RT= $\Theta(t)$