



TIB29 – Struktur Data dan Algoritma

U N I V E R S I T A S B U N D A M U L I A

PERINGATAN HAK CIPTA

Segala materi ini merupakan milik Universitas Bunda Mulia yang dilindungi oleh hak cipta.

Materi ini hanya untuk dipergunakan oleh mahasiswa Universitas Bunda Mulia dalam rangkaian proses perkuliahan.

Dilarang keras untuk mendistribusikannya dalam bentuk apapun.

Pelanggaran terhadap hak cipta ini dapat dikenakan sanksi hukum sesuai dengan perundang-undangan yang berlaku.

© Universitas Bunda Mulia

PERINGATAN HAK CIPTA

Segala materi ini merupakan milik Universitas Bunda Mulia yang dilindungi oleh hak cipta.

Dilarang keras untuk mengunduh dan atau merekam dan atau mendistribusikannya dalam bentuk apapun.

Materi ini hanya untuk dipergunakan oleh mahasiswa Universitas Bunda Mulia dalam rangkaian proses perkuliahan.

Pelanggaran terhadap hak cipta ini dapat dikenakan sanksi hukum sesuai dengan perundang-undangan yang berlaku

© 2024 Universitas Bunda Mulia



Multitary Tree

UNIVERSITAS BUNDA MULIA

Sub-CPMK

- Mahasiswa mampu menyusun data di dalam simpul-simpul multiary tree. (C3, A3)

Materi

1. Pengertian *Multiary Tree*
2. B-Tree



1. Pengertian *Multiary Tree*

1.1 *Multiary Tree*

- *Multiary Tree / N-ary Tree* adalah *Tree* dengan setiap simpul yang dapat memiliki anak lebih dari dua.
- *Multiary Tree* dapat diterapkan dengan cara:
 1. *Node* dengan sebuah ke anak pertama dan sebuah link ke saudara. Cara ini memiliki penyimpanan yang optimal.
 2. Sebuah *node* dengan banyak *Link* anak, cara ini memiliki keterbatasan jumlah anak dari suatu *node*, biasanya di implementasikan pada n-ary *tree*.

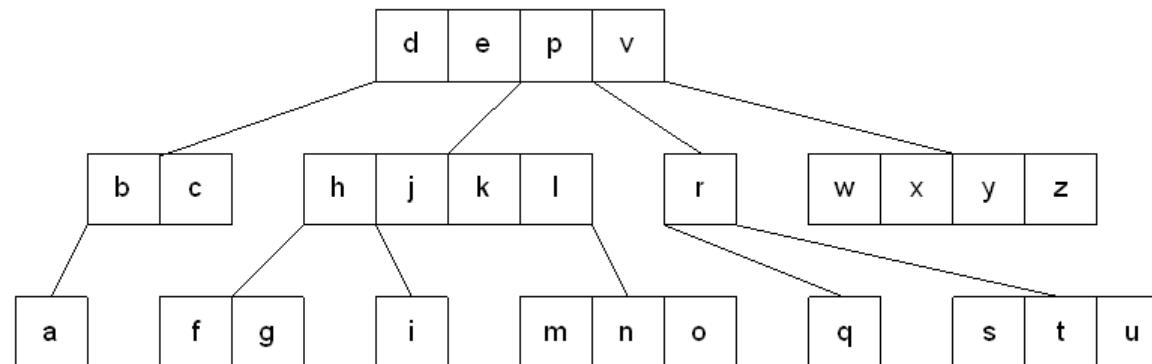
1.2 Multiway Tree

- *Multiway Tree* merupakan salah satu bentuk dari n-ary *Tree* / *multiary tree*.
- Salah satu varian dari *multiway tree* adalah *B-Tree* .
Catatan: harap diperhatikan *B-Tree* bukan *Binary Tree*, melainkan generalisasi dari *binary tree* untuk merepresentasikan *multiary tree* sehingga setiap *node* dapat memiliki anak lebih dari dua dengan setiap *key* di dalam *node* tersebut hanya memiliki dua anak agar lebih teratur dan mudah diakses.

1.2. Multiway Tree (Lanj.)

Contoh Multiway Tree

- *Multiway Tree* dengan *Order 5* yg memiliki 4 *key* dan 5 cabang
- Key d hanya memiliki satu anak kiri
- Key p memiliki dua: anak kiri (h j k l) dan anak kanan (r)
- Key r merupakan anak kanan dari key p dan sekaligus anak kiri dari key v



1.2 Multiway Tree (Lanj.)

1. Setiap *node/simpul* memiliki m anak dan $m-1$ *key/kunci*
2. *Key* pada setiap node/simpul disusun secara *ascending*
3. *Key* pada *first-i children* lebih kecil dari pada *key* ke i
4. *Key* pada *last m-i children* lebih besar dari pada *key* ke i



2. B-Tree

- Salah satu varian dari *multiway Tree* adalah *B-Tree*.
- *Size* pada setiap *node* dapat dibuat sebagai *size* dari *block*.
- Jumlah *key* pada setiap *node* dapat bervariasi tergantung ukuran *key*.
- *Size* dari sebuah *block* bervariasi untuk setiap *system*.

2.1 B-tree Order

B-tree dengan *order m* adalah sebuah sebuah *multiways search tree*

- Setiap *root* selain *leaf* memiliki paling tidak dua *subtree*
- Setiap *nonroot* dan setiap *each nonleaf* memiliki $k-1$ *keys* dan k *pointers* ke *subtrees* dimana $[m/2] \leq k \leq m$
- Setiap *leaf node* memiliki $k-1$ *keys where* $[m/2] \leq k \leq m$
- Semua *leaf* selalu pada *level* yang sama

2.2 Contoh B-Tree



2.3 Insertion

- Dibentuk secara *bottom to up*
- *Root* selalu berubah
- Jika ada sebuah ruang pada *leaf* tujuan, *node* dapat langsung disisipkan
- Jika semua *leaf* terisi
 - Bentuk *leaf* lain
 - *Keys* dibagi antara diantara *leaf* tersebut dan sebuah *key* diproposikan menjadi *parent*.
 - Dalam hal ini jika *parent* penuh, ulangi proses sampai mencapai *root* dan sebuah *root* dibentuk

2.4 Kondisi *Insertion*

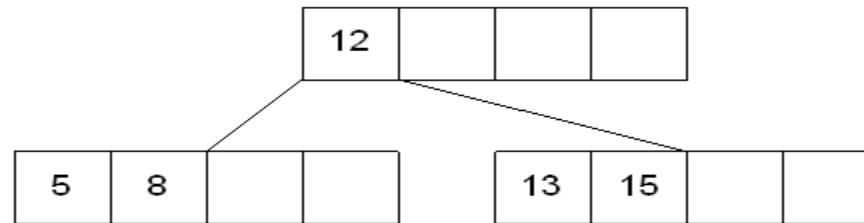
- Masih terdapat ruang untuk menempatkan sebuah *leaf*
- *Leaf* yang akan ditempatkan bisa jadi sudah penuh
- *Root* dari B-tree sudah penuh

2.5 Menyisipkan *Key* pada B-Tree

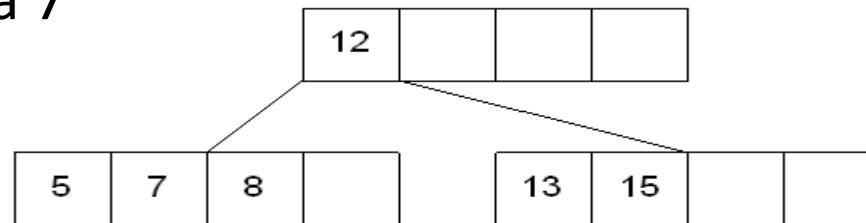
Tiga situasi yang umum terjadi

- Key di tempatkan pada *leaf* yang masih memiliki ruang (dengan tetap menjaga *ascending order*)
- *Leaf* yang akan diisi *key* sudah penuh
- *Special case* terjadi jika *root B-tree* sudah penuh

2.6. Contoh Penyisipan

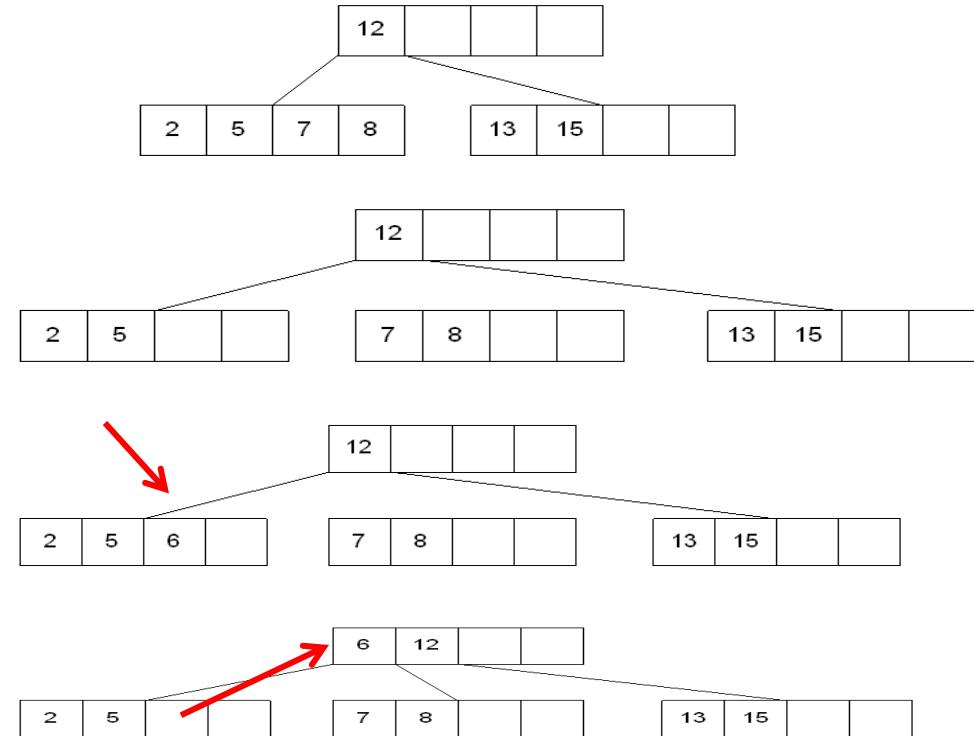


Menyisipkan data 7



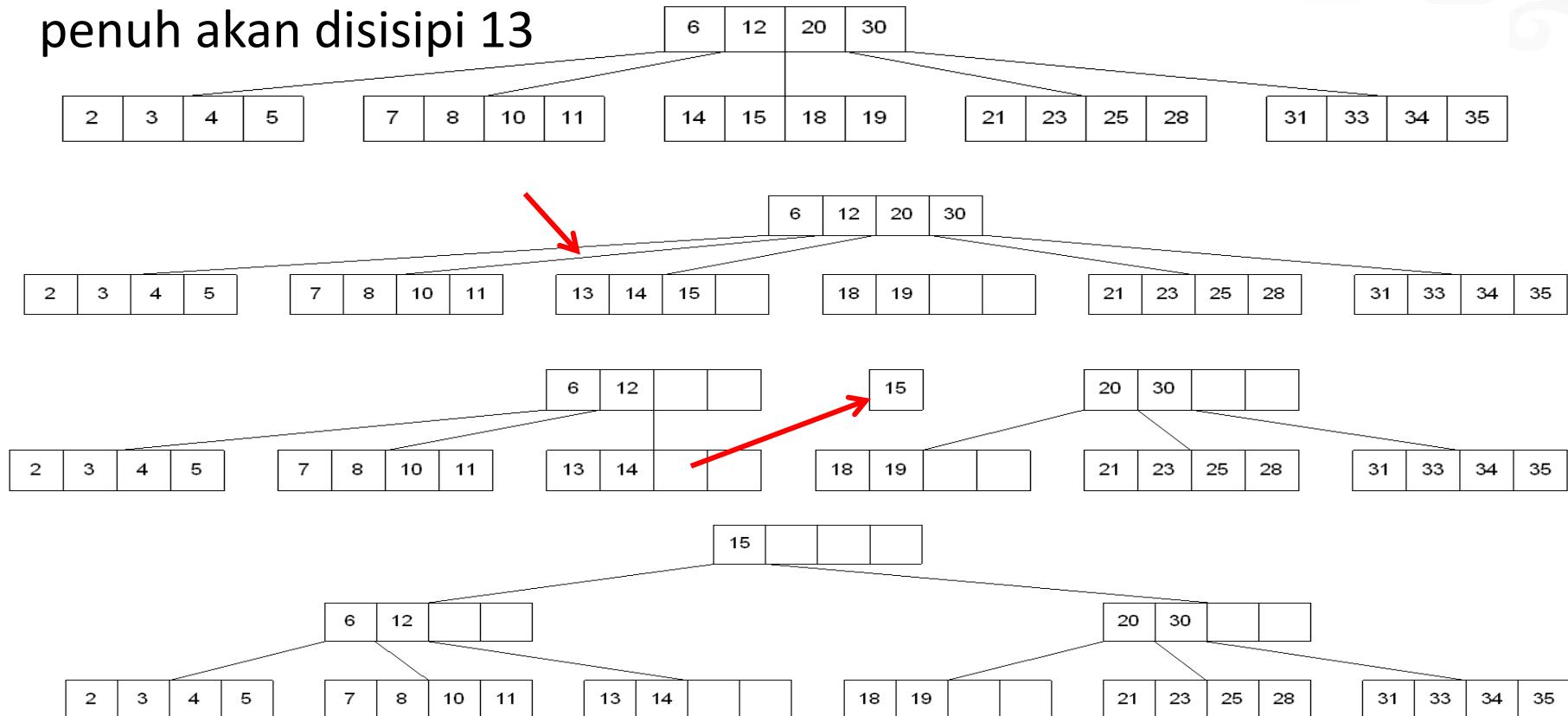
2.6 Contoh Penyisipan (Lanj.)

Menyisipkan 6 Pada
tree berikut ini



2.6 Contoh Penyisipan (Lanj.)

Tree yang sudah penuh akan disisipi 13



2.7 Menghapus Key dari B-Tree

Ada dua kasus

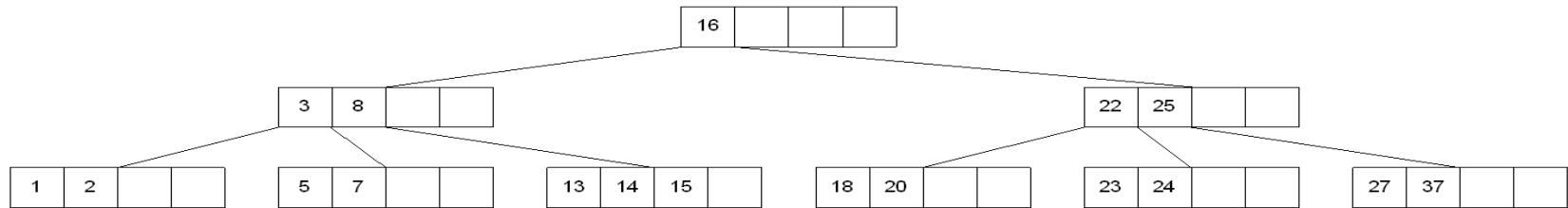
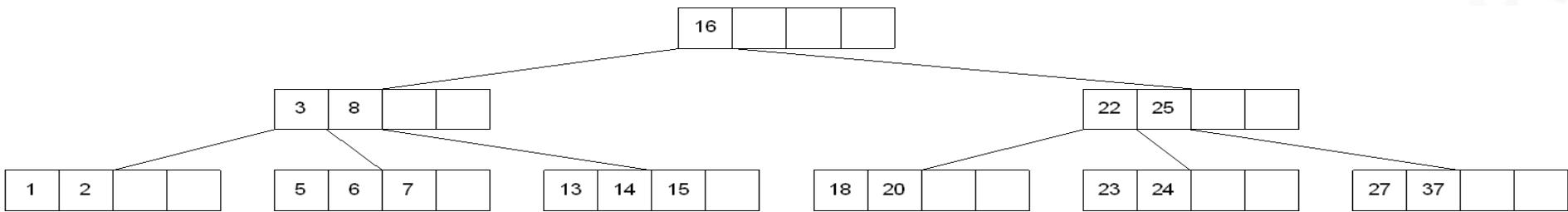
- Hapus key dari *leaf node*
- Hapus key dari *nonleaf node*
 - *Similar dengan deletingByCopying()*

2.8 Menghapus Dari Leaf

Kondisi

- Jika jumlah *key* setelah dihapus lebih besar atau sama dengan $m/2$
 - Hapus *key*
 - Sisipkan *key* dari sebelah *key* yang dihapus untuk mengisi sel yang kosong

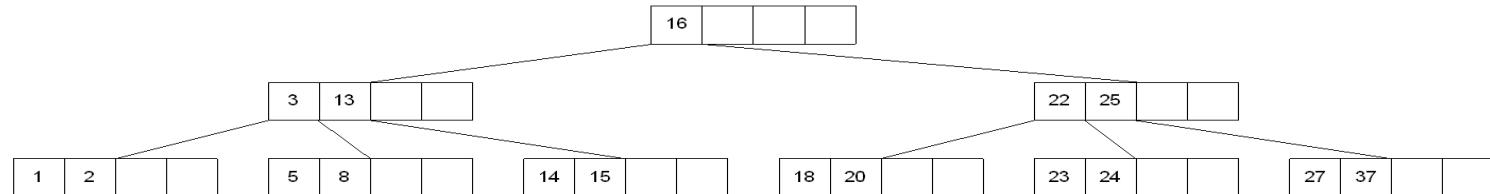
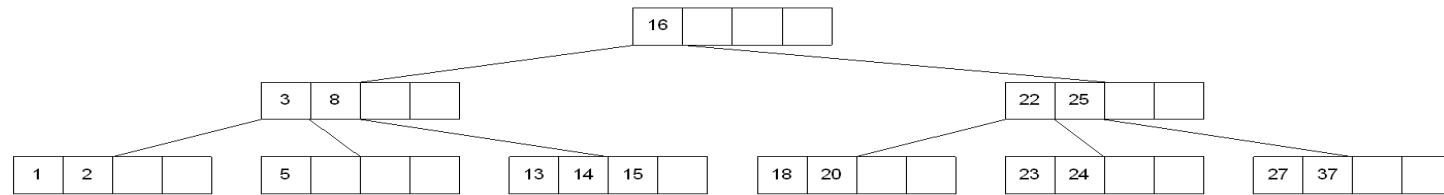
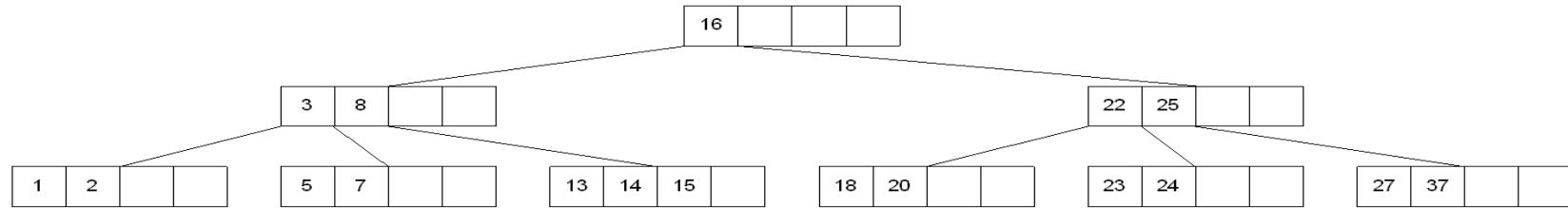
2.8 Menghapus Dari Leaf (Lanj.)



2.8 Menghapus Dari *Leaf* (Lanj.)

- Jumlah *key* setelah penghapusan kurang dari $m/2$
 - Jika jumlah *key* pada kiri atau kanan *sibling* cukup untuk mengisi kekurangan *node* target
 - Ambil *key* dari *parent* dan *insert* menjadi *node*.
 - Ambil *key* dari kanan atau kiri *sibling* untuk mengisi *key* pada *parent node*

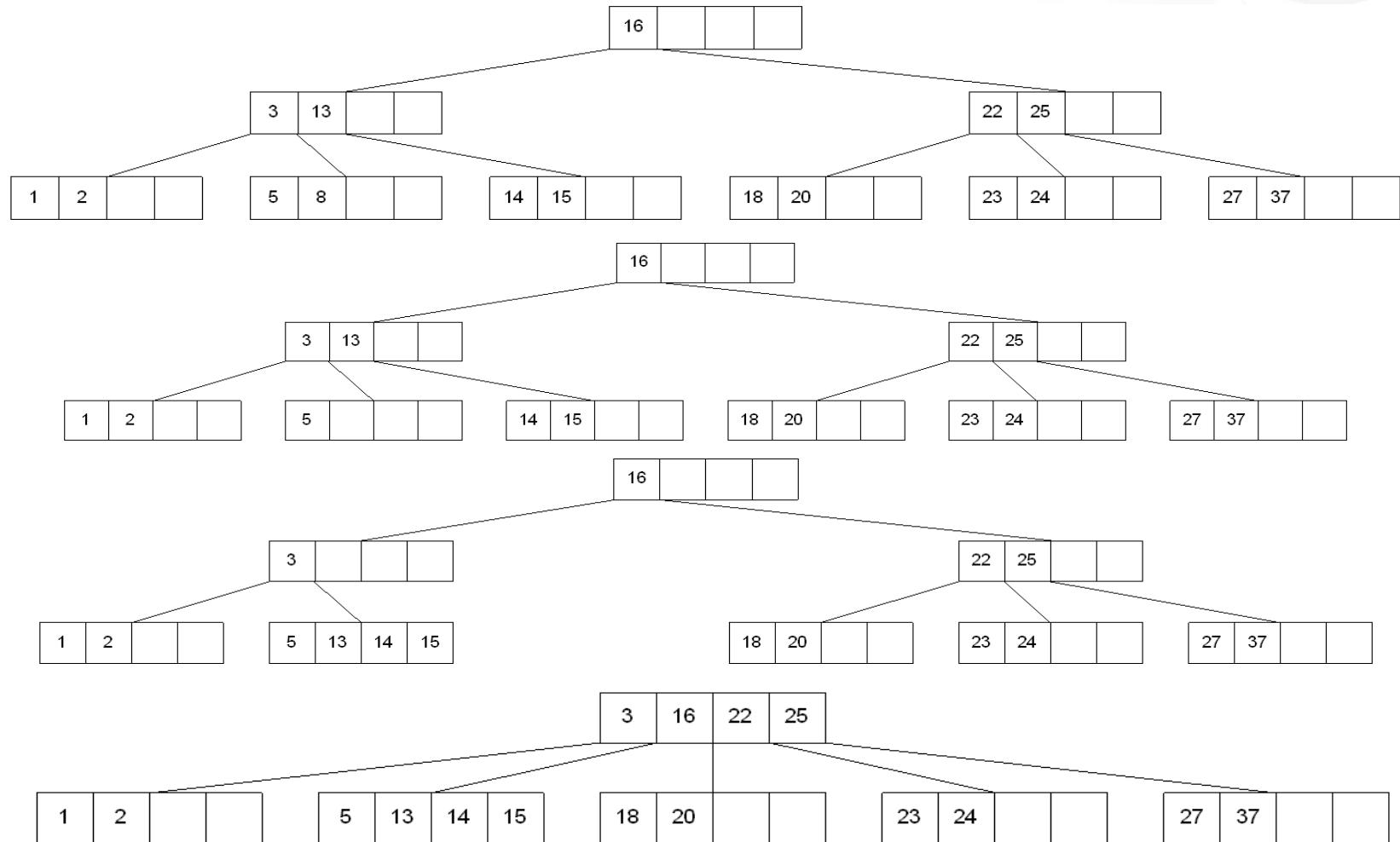
2.8 Menghapus Dari Leaf (Lanj.)



2.8 Menghapus Dari *Leaf* (Lanj.)

- Jika jumlah *key* setelah penghapusan kurang dari $m/2$ (cont.)
 - Jika jumlah *key* pada kiri atau kanan *sibling* tidak cukup untuk mengisi kekurangan sebagai *target node*
 - *Merge key sibling* dari kiri atau kanan, dengan mengambil *key* dari *parent node* meletakkan diantara nya
 - Geser *key* sebelah kanan *parent node* untuk mengisi sel kosong setelah *key* tersebut diambil untuk menggantikan anaknya.
 - Jika jumlah *key* pada *parent* kurang dari $m/2$, dapat diisi dengan proses untuk mengisi *node* kosong pada proses yang sama dengan menghapus *non leaf*

2.8 Menghapus Dari Leaf (Lanj.)



2.9. Menghapus Dari *Node Non-leaf*

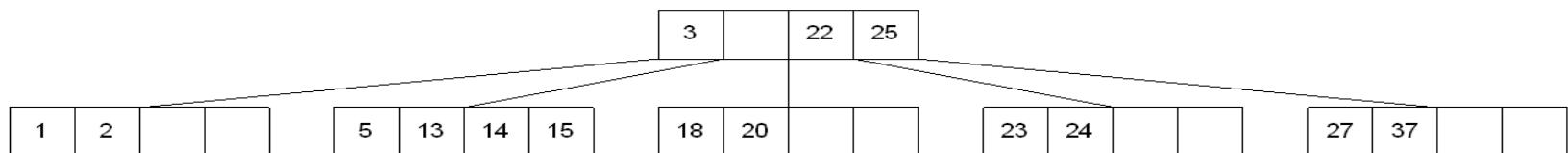
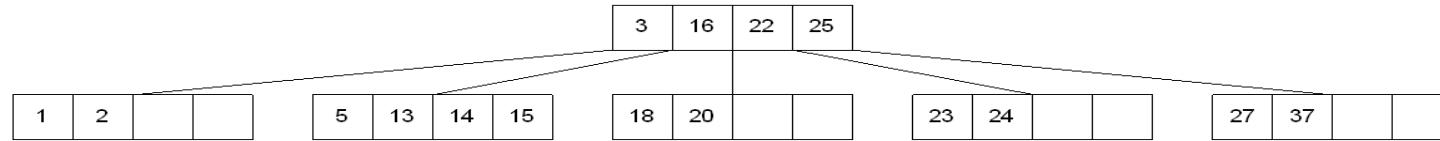
- Key yang lebih besar dari yang dihapus dipindah ke kiri untuk menggantikan yang sudah dihapus
- Jumlah *key* pada *leaf* harus lebih besar sama dengan $[m/2]$

2.9 Menghapus Dari *Node Non-leaf* (Lanj.)

- Penghapusan dari *node non-leaf* lebih sedikit prosesnya daripada menghapus dari *leaf*. *Deletion from a non-leaf node is reduced to deleting a key from a leaf*
- Key yang dihapus direplace dengan *immediate predecessor* nya (dapat juga dengan *successor*) yang dapat ditemukan pada sebuah *leaf*

2.9 Menghapus Dari Node Non-leaf (Lanj.)

contoh mengganti simpul terhapus dengan *predesesor*



Isi sel terhapus dengan *predesesor*

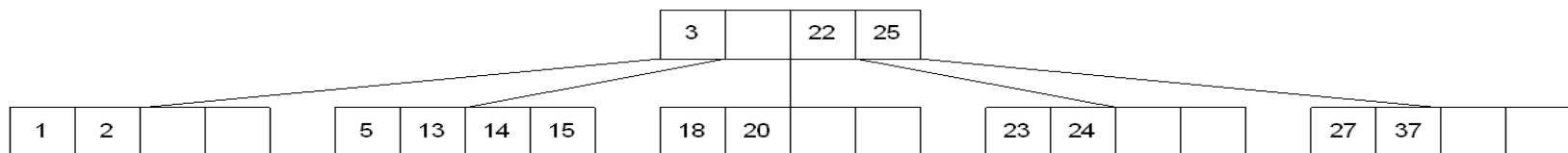
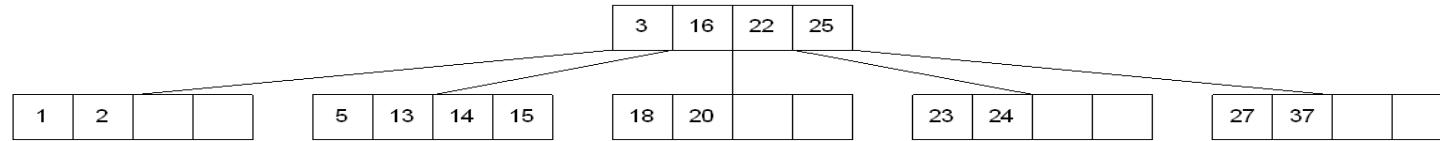


immediate predesesor dihapus



2.9 Menghapus Dari Node Non-leaf (Lanj.)

contoh mengganti simpul terhapus dengan suksesor



Isi sel terhapus dengan suksesor

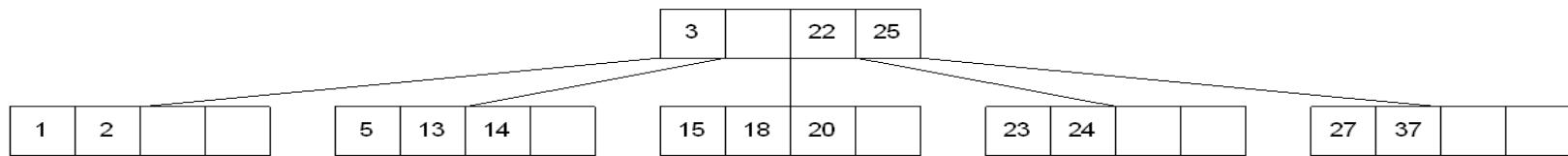
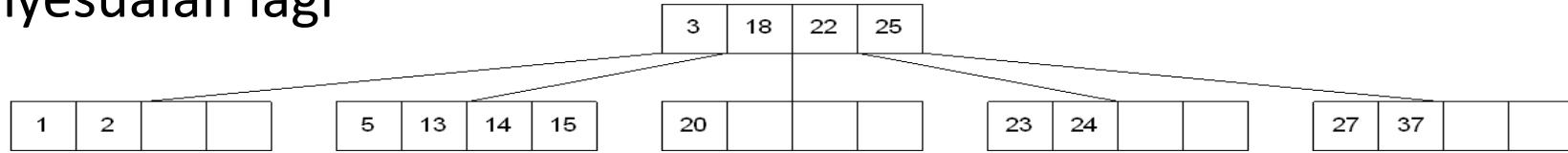


Suksesor dihapus sehingga
data 20 geser ke kiri



2.9 Menghapus Dari Node Non-leaf (Lanj.)

Jika kondisi menjadi tidak seimbang, maka perlu dilakukan penyesuaian lagi



Catatan:

- Contoh-contoh disadur dari:

<http://faculty.cs.niu.edu/~freedman/340/340notes/340multi.htm>

Contoh Record Untuk B-Tree dengan C++

```
Struct RecBTree
{
    int element[5];
    RecBTree *child[6];
}

RecBtree *Root, *Cursor;
```

Contoh Record Untuk B-Tree dengan Python

```
class RecBTree:  
    def __init__(self):  
        self.element = [0] * 5  
        self.child = [None] * 6  
  
# Creating instances of the structure  
Root = RecBTree()  
Cursor = RecBTree()
```

Ringkasan

- N-Ary *tree* atau *Multiary Tree* merupakan *tree* yang dapat memiliki maksimum n anak pada setiap *node* nya.
- *Multiway Tree* merupakan salah satu bentuk dari n-ary *Tree*
- Salah satu varian dari *multiway Tree* adalah *Btree*.

PERINGATAN HAK CIPTA

Segala materi ini merupakan milik Universitas Bunda Mulia yang dilindungi oleh hak cipta.

Dilarang keras untuk mengunduh dan atau merekam dan atau mendistribusikannya dalam bentuk apapun.

Materi ini hanya untuk dipergunakan oleh mahasiswa Universitas Bunda Mulia dalam rangkaian proses perkuliahan.

Pelanggaran terhadap hak cipta ini dapat dikenakan sanksi hukum sesuai dengan perundang-undangan yang berlaku

© 2024 Universitas Bunda Mulia

PERINGATAN HAK CIPTA

Segala materi ini merupakan milik Universitas Bunda Mulia yang dilindungi oleh hak cipta.

Materi ini hanya untuk dipergunakan oleh mahasiswa Universitas Bunda Mulia dalam rangkaian proses perkuliahan.

Dilarang keras untuk mendistribusikannya dalam bentuk apapun.

Pelanggaran terhadap hak cipta ini dapat dikenakan sanksi hukum sesuai dengan perundang-undangan yang berlaku.

© Universitas Bunda Mulia



Terimakasih

TUHAN Memberkati Anda

Teady Matius Surya Mulyana (tmulyana@bundamulia.ac.id)