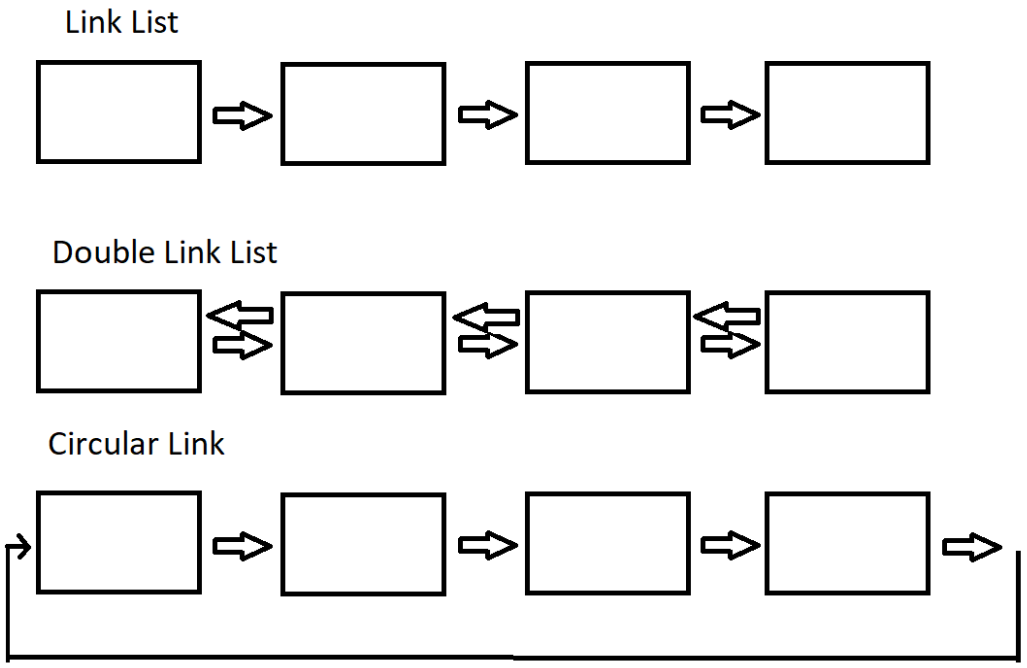
Link List

1.



2. Perbedaan utama dari Link Listed dan Array adalah Link Listed ukurannya sangat fleksibel bisa ditambah sampai berapapun, sedangkan array itu ukurannya bersifat statis berarti array memiliki batas maksimum untuk isinya cth a[8] berarti a hanya bisa menampung sebanyak 8 saja sedangkan untuk link list tidak ada

3.

Stack and Queue

1. Perbedaan utama stack dan queue adalah urutan masuk serta keluarnya data. Kalau Stack itu masuk pertama keluar terakhir berarti data yang bisa di pop atau dihapus atau keluar adalah data yang paling atas (terakhir masuk) cth stack 5 4 3 2 1 berarti 1 masuk paling pertama dan 5 masuk paling terakhir dan yang bisa dihapus hanya angka paling depan yaitu 5. Untuk queue kebalikannya yang pertama masuk pertama keluar. Cth 1 2 3 4 5 jadi 1 adalah yang pertama masuk dan 5 yang terakhir masuk dan yang bisa keluar adalah yang paling depan yaitu 1

2. Infix

Ini sistenya sama dengan perhitungan matematika biasa

3 + 3 + 4

Stack

- 3

Ketemu +

- 3
  3
- 6

Ketemu +

- 4
  6
- 10

Prefix

Prefix ini kita baca angkanya dimulai dari belakang, jadi kita cari dulu operator nya dari belakang lalu kalo udh ketemu kita cari 2 angka dibelakang operatornya dan urutannya itu +AB = A + B

+ 4 + 3 3

Stack

- 3
- 3

  3

Ketemu +

- 6
- 4
  6

Ketemu +

- 10

Postfix

Post fix ini kita baca angkanya dari depan, lalu kalo ketemu operator baru lakuin operasi dengan angka tersebut sesuai operatornya. Cth A B + C + Jadi A ketemu B lalu ketemu + maka A + B lalu ketemu C dan ketemu + lagi jadi Hasil A + B di + C itu hasilnya

3 3 + 4 +

Stack

- 3

- 3
  3

Ketemu +

- 6

- 4
- 6

Ketemu +

- 10

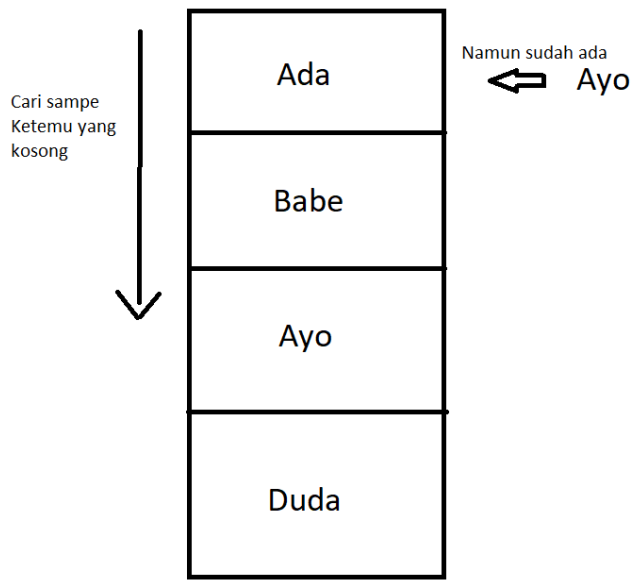
Hashing dan Hash Table

1. Hash table itu adalah cara penyimpanan suatu data baik itu berupa string ataupun angka dan penyimpannya didasarkan pada sesuatu misal urut abjad.

Hash function ini adalah rumus atau cara kita memasukkan data yang ada ke dalam hash table. Rumus ini ditetapkan berdasarkan penyimpanan dari hash table jadi misal hash tablenya itu urut abjad maka hash functionnya dibuat agar data yang masuk ke hash table sesuai urutan abjadnya

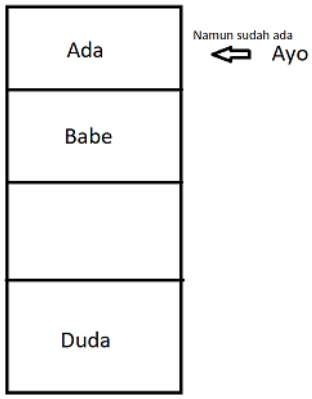
Collision ini saat data yang masuk ke hash table sudah ditempati oleh data yang lain. Cth misal di hash table 1 ada kata Ada dan data yang mau dimasukkan ke hash table berdasarkan huruf pertama dan urut abjad. Jadi saat Ayo keluar dari hash function maka dia diarahkan ke hash table 1 tapi sudah ada yang menempati yaitu Ada maka kondisi inilah yang disebut collision

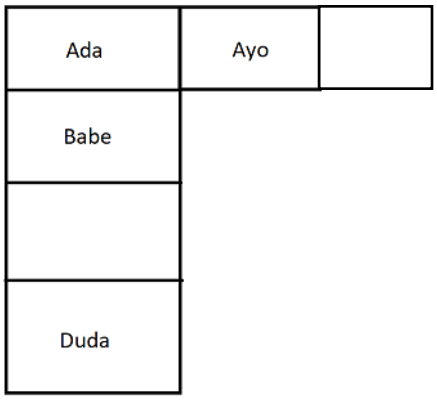
2. Method yang pertama Linear Probing



Jadi Linear probing ini memiliki sistem bahwa apabila suatu data mau dimasukkan ke hash table dan di hash table tersebut sudah ada isinya maka data yang mau dimasukkan tersebut di letakkan di tempat yang kosong. Jadi di loop mulai dari hash table tempat seharusnya sampai ketemu hash table yang kosong.

Methode Chaining



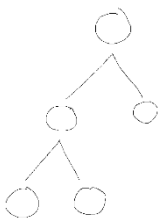


Jadi Methode chaining ini apabila data yang mau dimasukkan ke hashtable namun di hashtable tempatnya sudah terisi maka data yang dimasukkan ini diletakkan di tempatnya namun di samping

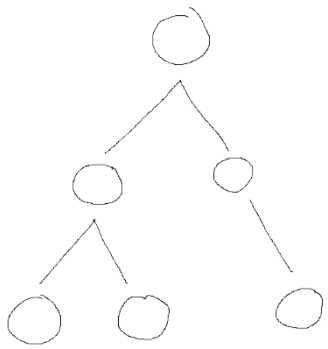
Binary Search Tree

1. 5 tipe binary tree

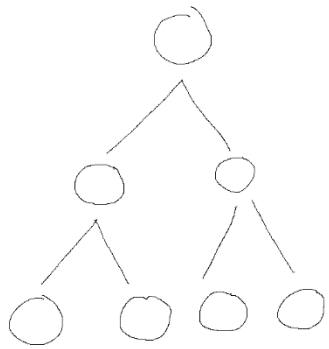
- Full
  Hanya boleh punya 2 anak atau tidak ada anak sama sekali. Tidak boleh punya 1 anak saja



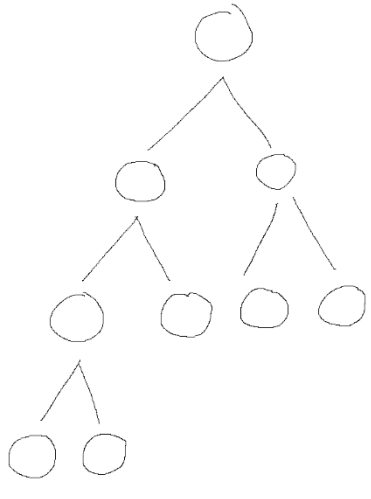
- Complete
  Semua harus keisi dengan node jadi apabila belum sampai di level terbawah maka harus punya minimal 1 anak



- Perfect
  Semua parent harus memiliki 2 anak



- Balanced
  Apabila tinggi dari sisi kiri dan sisi kanan maksimal minus 1

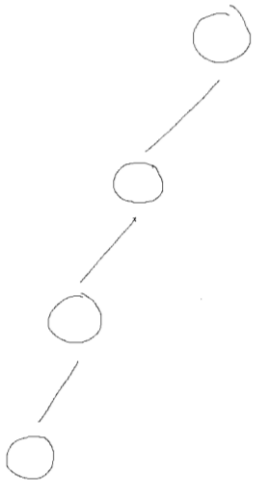


Sisi kiri tingginya 4

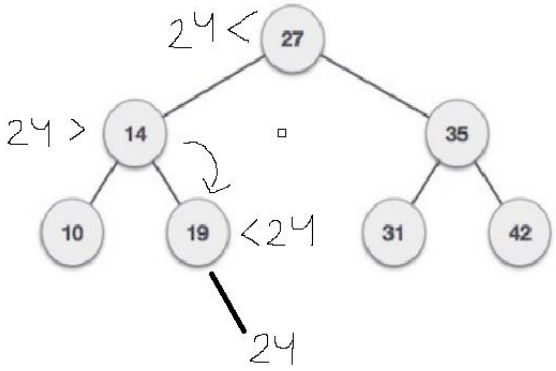
Sisi kanan tingginya 3

Jadi selisihnya 1

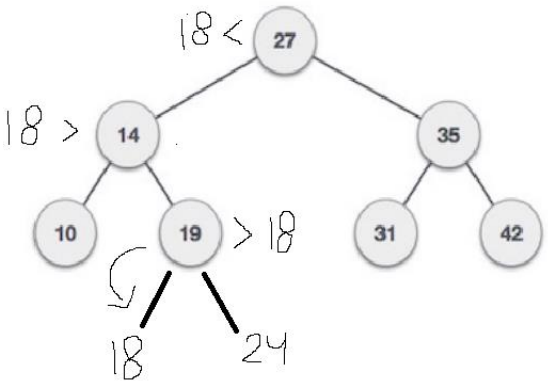
- Degenerate
  Apabila Node nya itu hanya di satu sisi saja



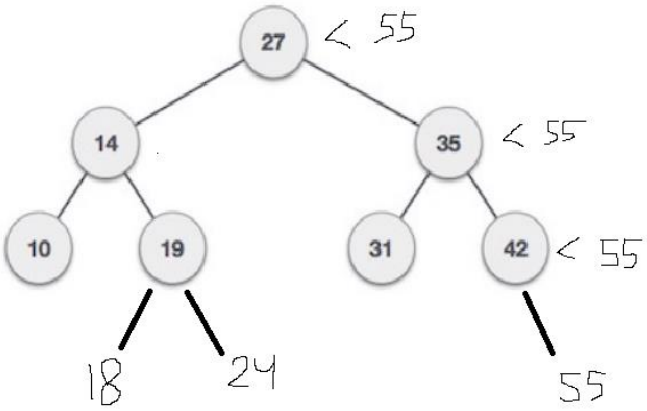
2. Simulasi insert 24, 18 55



Jadi pertama dia cek dulu dia lebih kecil atau besar dari pada root. Karena 24 lebih kecil maka dia kekiri
Lalu di cek dengan anak dari root karena 24 > 14 maka dia ke kanan dari parent
Lalu dibandingkan lagi dengan anak dari parent karena 24 > 19 maka letak 24 adalah dibawah kanan dari 19

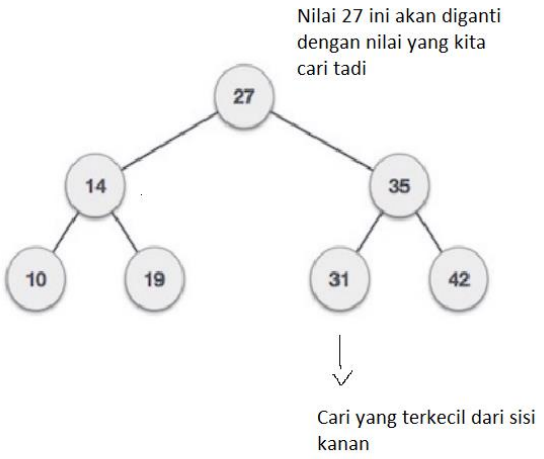


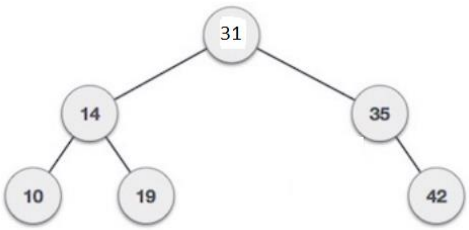
Jadi pertama dia cek dulu dia lebih kecil atau besar dari pada root. Karena 18 lebih kecil maka dia kekiri
Lalu di cek dengan anak dari root karena 18 > 14 maka dia ke kanan dari parent
Lalu dibandingkan lagi dengan anak dari parent karena 18 < 19 maka letak 18 adalah dibawah kiri dari 19

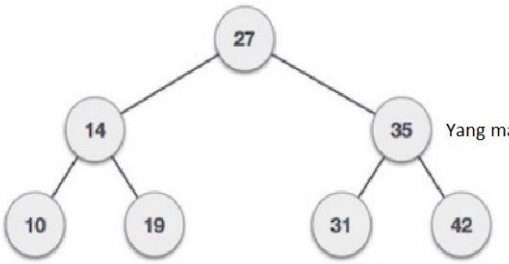


Jadi pertama dia cek dulu dia lebih kecil atau besar dari pada root. Karena 55 lebih besar dari 27 maka dia ke kanan
Lalu di cek dengan anak dari root karena 55 > 35 maka dia ke kanan dari parent
Lalu dibandingkan lagi dengan anak dari parent karena 55 > 42 maka letak 55 adalah dibawah kanan dari 42

3. Hapus 27 35 42

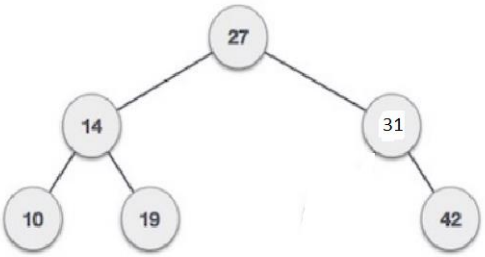


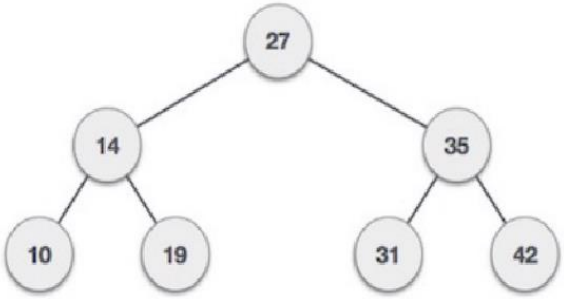




Yang mau dihapus

Cari anak terkecil dari parentnya. Jadi nanti nilai ini yang akan gantiin yang akan dihapus. Lalu node ini akan dihapus





Karena 42 adalah leaf maka jika mau hapus 42 tinggal hapus saja dan tidak ada perubah lain di tree nya