

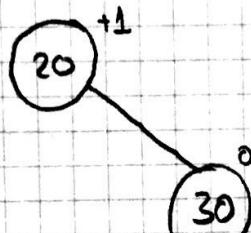
## 1-) AVL Tree Add

add: 20



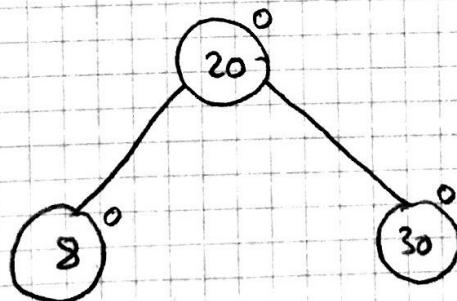
→ ilk gelen öge ve denge: 0.

add: 30



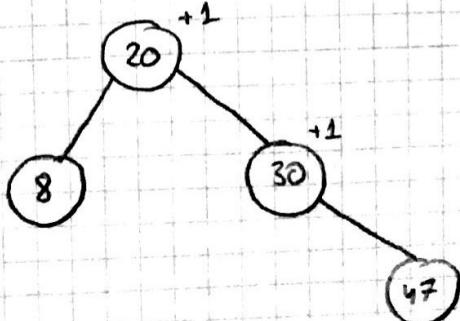
→ Eklelen ögenin denge: 0, faktet faktet denge: 1 oldu

add 8



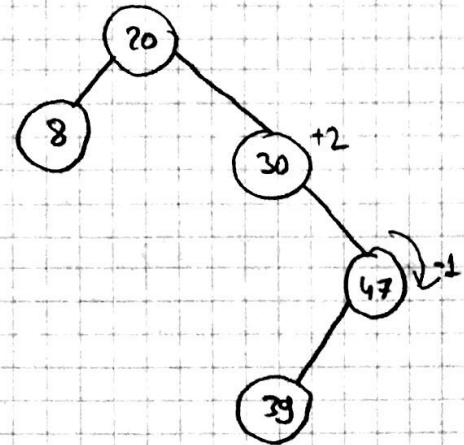
→ 8 geldi, ve 8'in denge sıfır, ordinden root'un da dengeyi 0 yaptı

add 47



→ 47' gelince 30'ın ve 20'ın denge: +1 oldu faktet hala kabul edilebilir sunular içerisinde

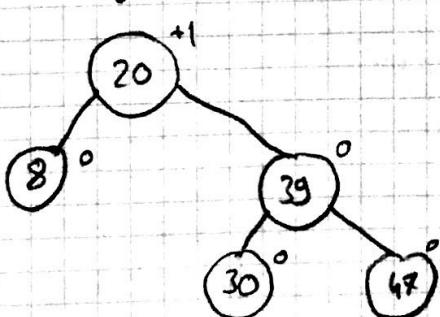
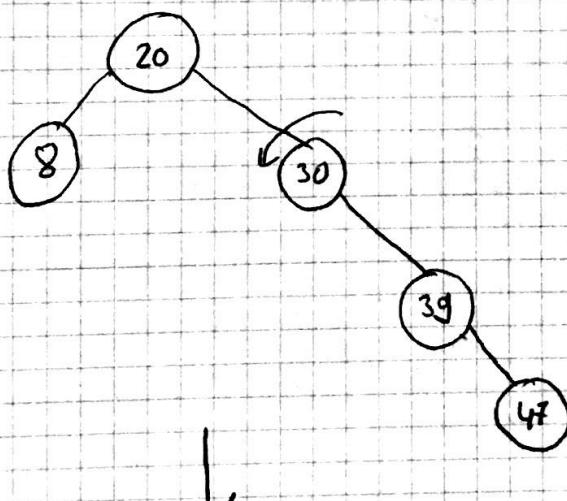
add : 39



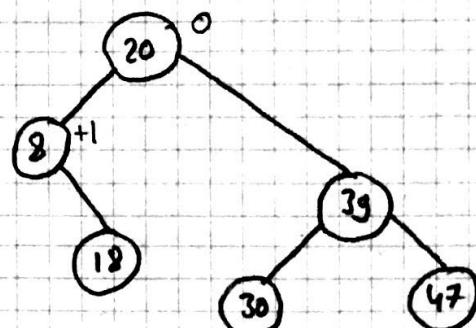
→ 39 gelince 3dn sağes: +2

old ve esinli sağlı, sağa R-L tree

oldugu için nce 47 arasında 30' etrafında  
dönme yapbit.

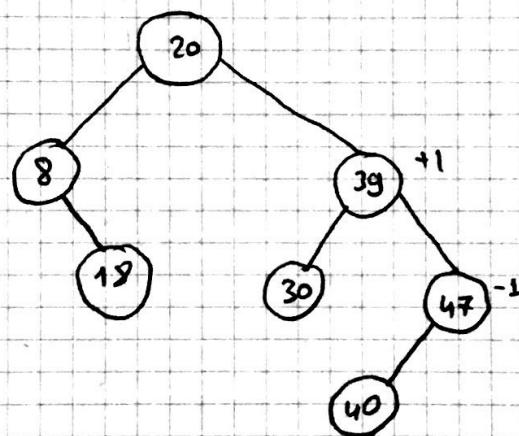


add 18:



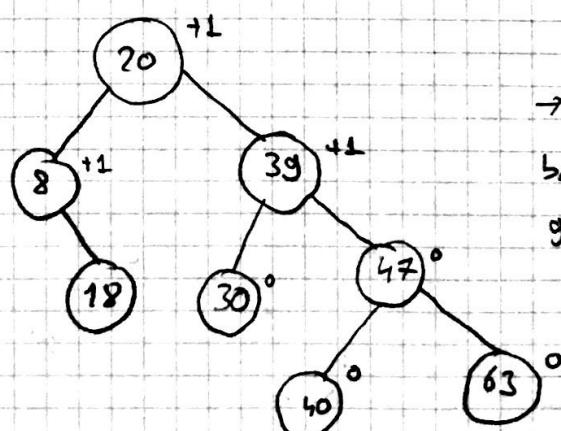
→ 18'in gelmesiyla bir dengeyi sınırları dışına çıkarmadı.

add 40:



→ 40'in eklenmesi dengeyi sınırları dışına çıkararak şekilde değiştirmedi.

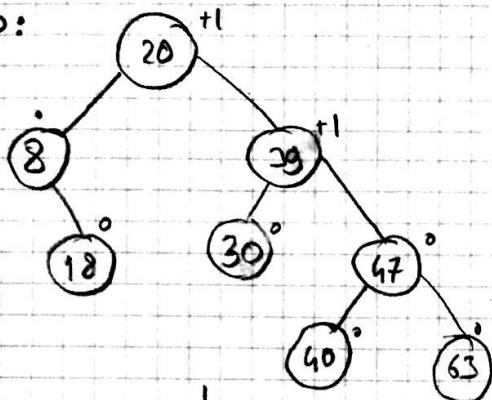
add 63:



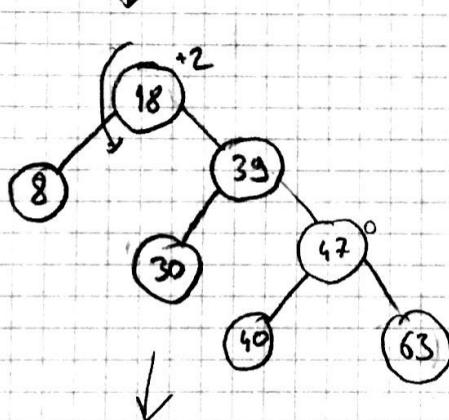
→ 63'in eklenmesi dengeyi bozmadı. Rotate işlemi gereklidir.

## AVL Tree Remove

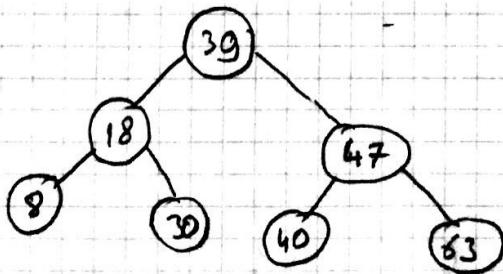
Remove 20:



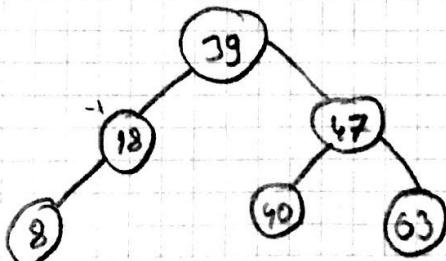
\* Normal BST remove işlemi gibi haldiriz. En büyük elemen gelir sağ oğuldan.



Root'un dengesi +2 olduğu için sola doğru dondurme yapınız.

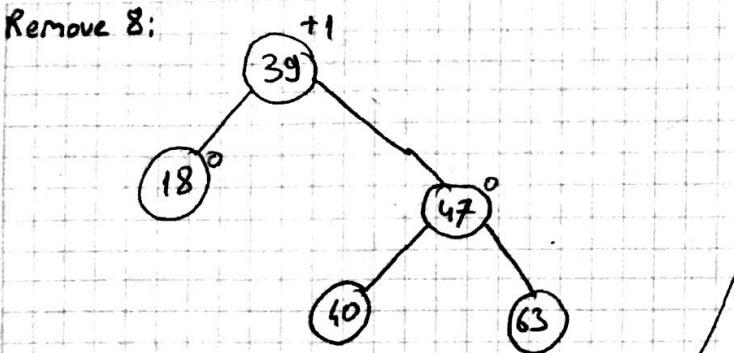


Remove 30:



30'un kaldırılması dengede hahangi bir sorun yaratırız.

Remove 8:

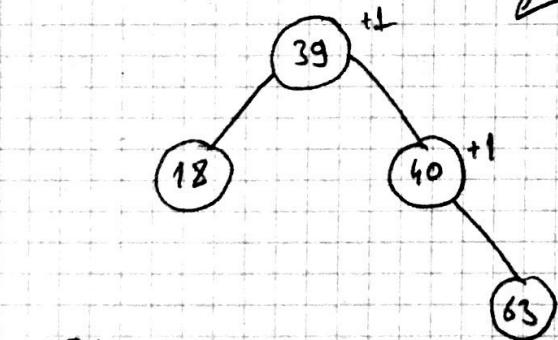


\* 8'in kaldırılması 20 + dengeye.

\* 47'in kaldırılması durumunda

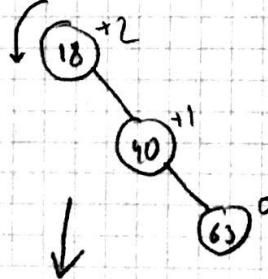
47'in yerine 40 yerler ve sadece  
40'ın değesi +1 olur. Dandarmeye gerek  
kalmaz.

Remove 47:



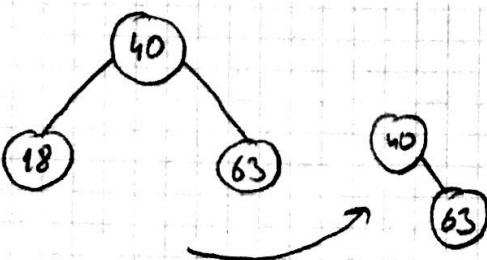
39'un kaldırılmasında 39 yerine 18 yer.  
Dengesi +2 olur. R-R oğus olduğu için  
sola dönürlüyor

Remove 39:

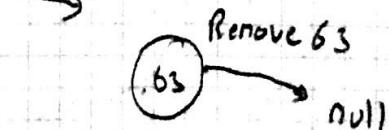


\* Bundan sonra, kaldırımlar  
değerde herhangi değişiklik  
yapma.

Remove 18:



Remove 40



## ② Red - Black Tree

20

→ 20 geldi. İlk gelen olduğu için siyah.

20

30

→ 30 geldi. Sağ left tree

null iken geldi kırımsız oldu

8

20

30

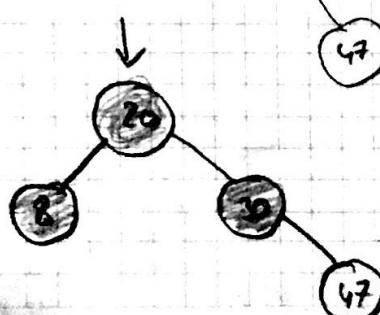
→ 8' left tree (20'nin) null olduğu  
yere geldiği için kırımsız.

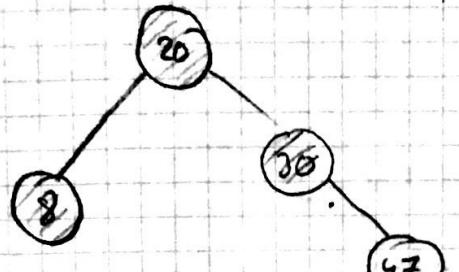
20

8

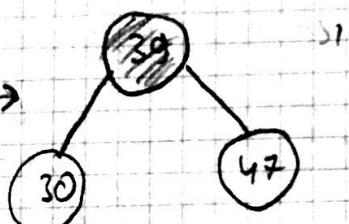
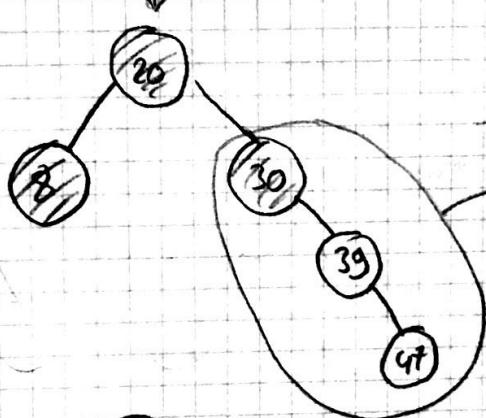
30

→ ebeveyn kırımsız ve kardeşlerde kırımsız  
dolayısıyla grand parenti kırımsız ve uncle  
ile parenti siyah yapılıyor.  
Ardından root'u siyah'a çeviriliyor.

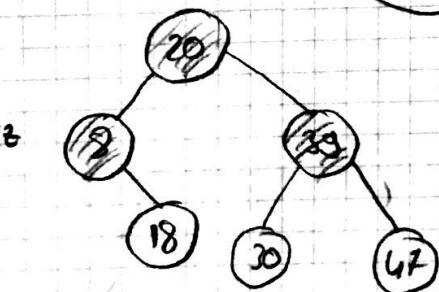




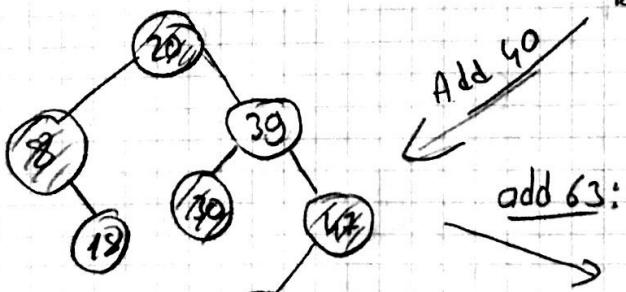
\* İki kırmızı yan yana olduğu için ve türdeş  
siyah olduğu için 39 ile 47 yer değiştir.



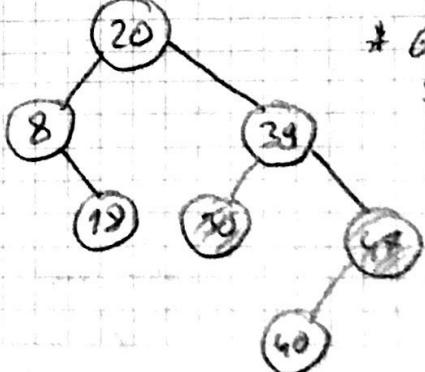
\* 18 sorusuz eklendi.



- 60 kirmizi olarak eklendi, parent'i kirmizi  
bu bir sonun parentin kardesi de kirmizi  
oldugu için parent ve uncle siyah grandparent  
kirmizi oldu.



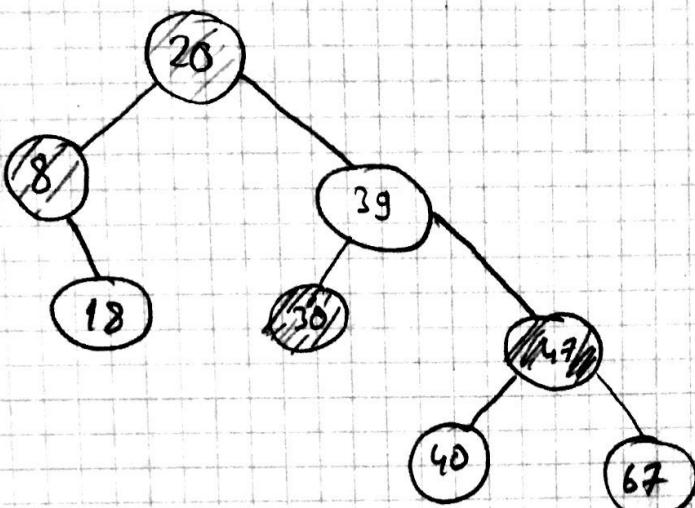
add 63:



\* G'de son yok.

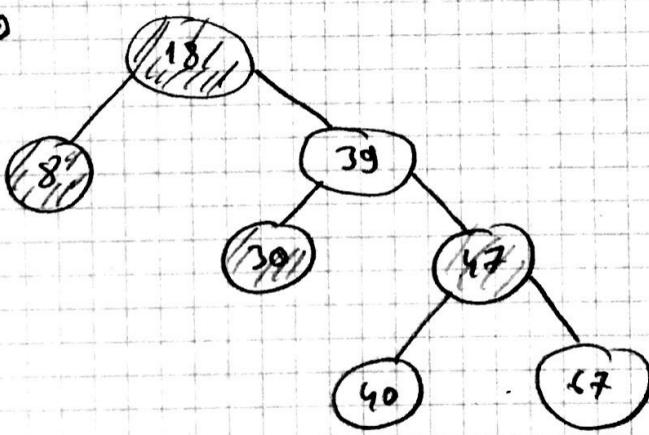
## R-B Remove

remove 20



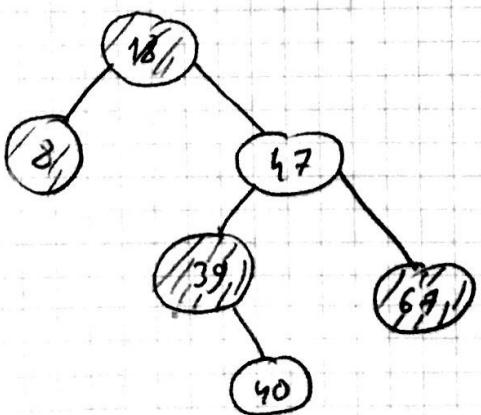
\* 20 silindiğen sol en büyük odağı yeldi.

remove 30



\* 30 silindi, sadece null kubbi, doldurme yapılıncaya 39'ın 67'nin soluna, 40 39'un sağına yattı. Parent kubusları birbirini olunca grand parent kümeleri parent ve uncle sadece oldular.

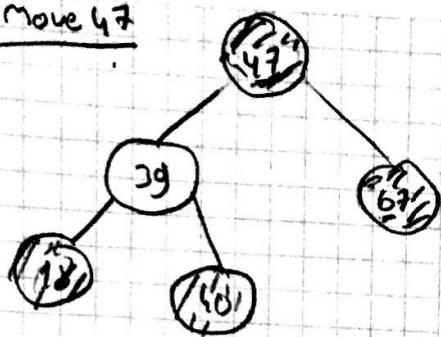
remove 8



\* 8 talksıca 18'in solu null oldu, sola döndü, 18'in sağı 39, root 47 oldu.

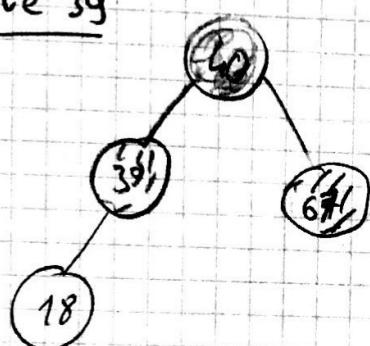
Ardından 18 kumizi, 39 sağını 40 kumizi kulanı kural kesildi. 18 sola döndü, krodaları sadece parenti kumizi yaptı.

remove 47



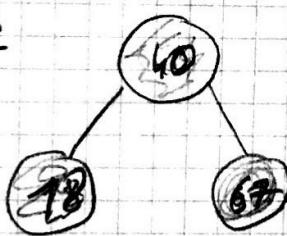
47'in yerine 60 yeri,  
39 ile 18'in enini değiştirdi  
Yeni gelenler sozlenmesin diye

remove 39



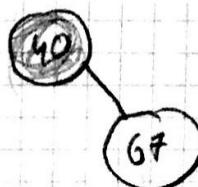
\* 39 yerine 18 yeri. Rengi  
siyah oldu

remove 18



\* 18 kalkınca yapsa, boşluksen düz  
67 kırızbıya döndü

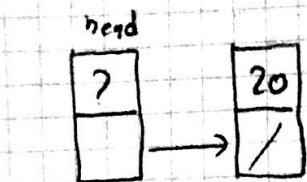
remove 40



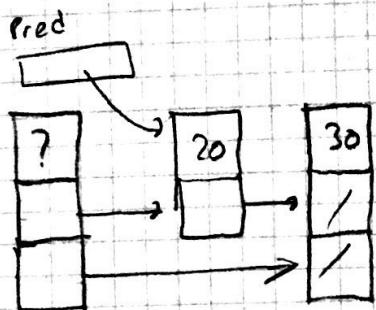
remove 67



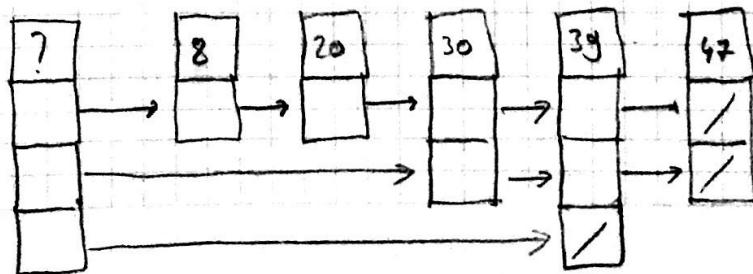
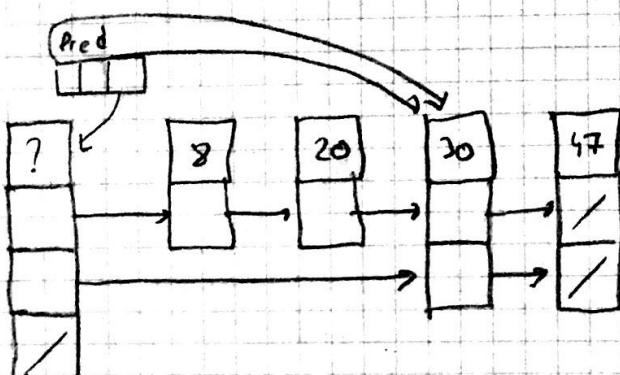
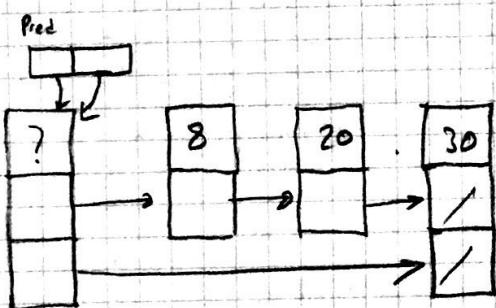
## Skip List

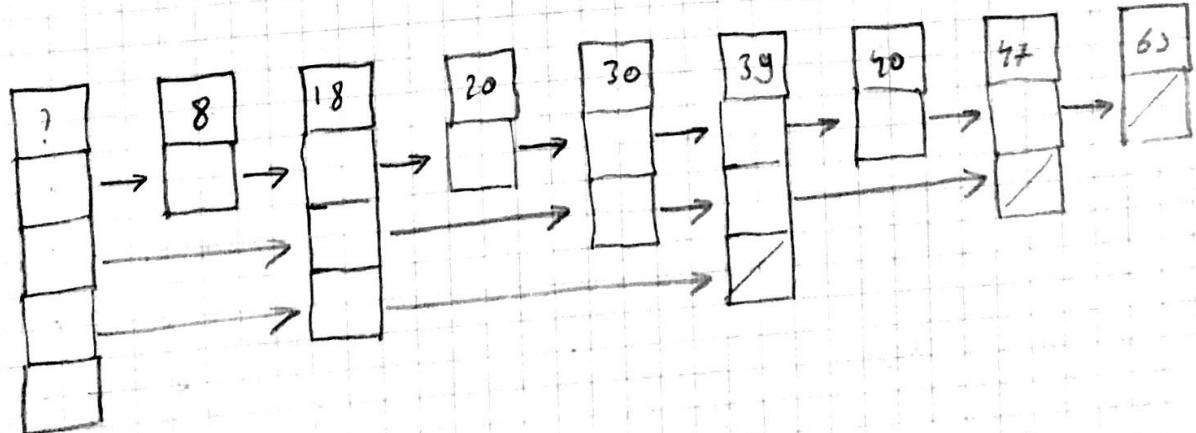
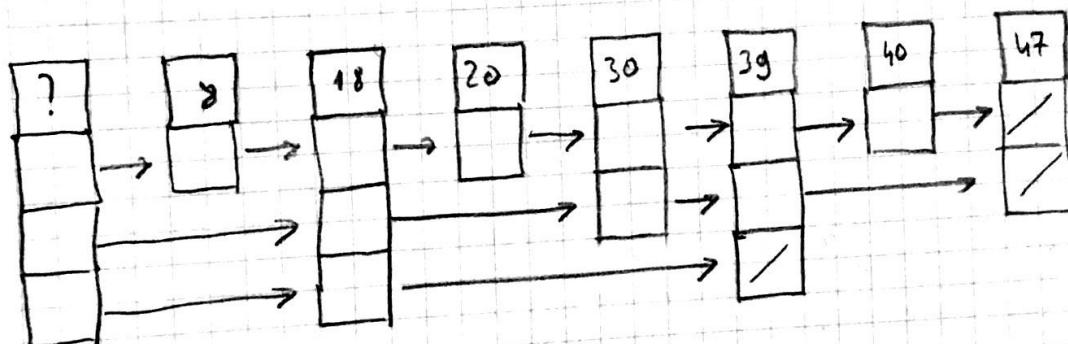
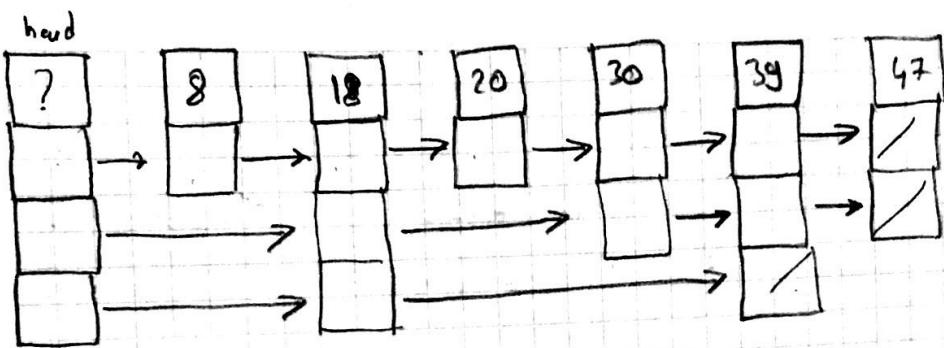


1. tane sayı olduğunda  $2^0 = 1$  olduğundan  
dolayı kendi层级上 duruyor.

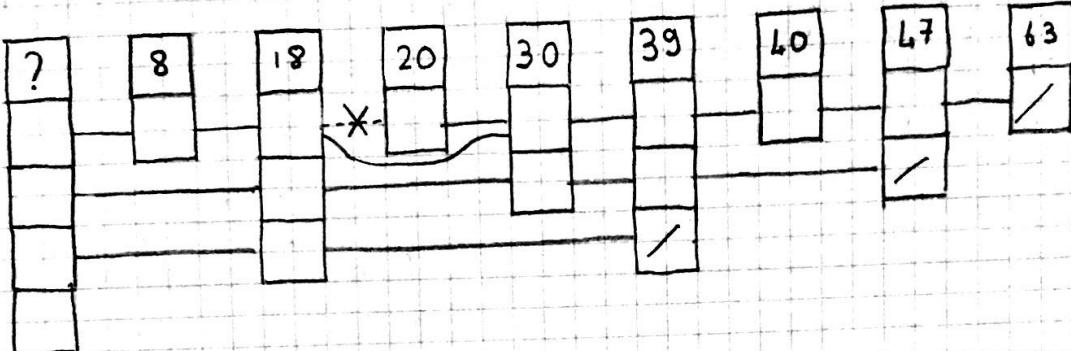


2. sayı geldiğinde 1'in  $2^1 = 2$  olduğundan  
dolayı head yenilendi!

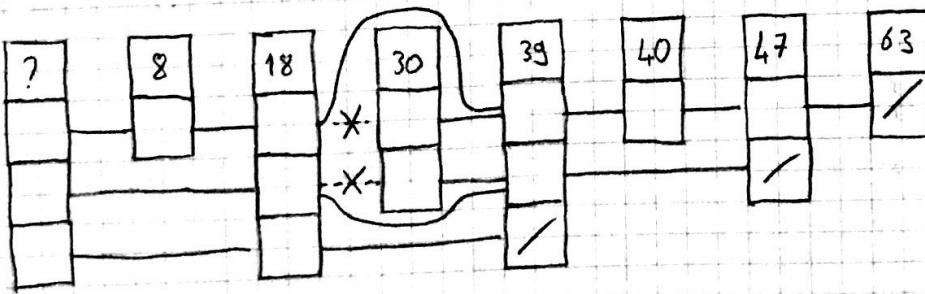




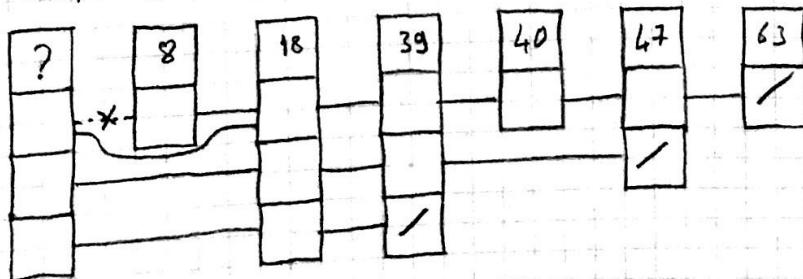
## Skip List Remove



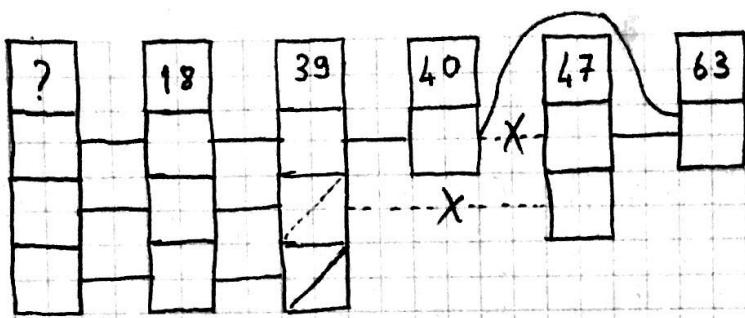
\* Önce 3. seviyeden başlayıp bir sonraki linkin 20 olup olmadığını kontrol ediyoruz. 20'den büyükse, bir seviye aşağıda devam ediyoruz. Burada 3. ve 2. seviyede 18'den sonra 20'den büyük sayı geliyor fakat 1. seviyede 18 den sonra 20 gelmiyor. Burada 18'in 1. seviye linkinin 20'nin 2. seviye linkinin gösterdiği yeri gösteriyoruz.



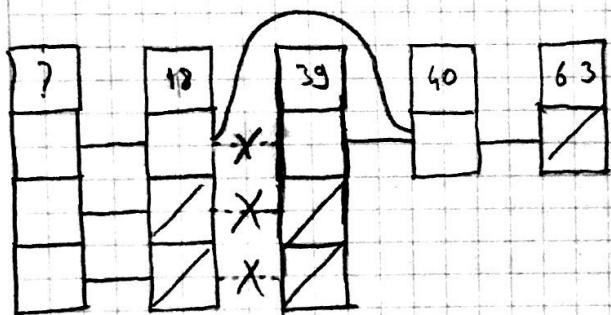
\* 3. seviyeden başlayıp sırayla 30 için arama yapılıyor. 18'in 3. seviyesinin linki 20'den büyüklerken 2. seviyesine bakılır, 30'u gösterdiği için 2. seviyesinin linki artik 30'un linkini göstererek şekilde ayarlanır. Ardından 1. seviyesi için bakılır, Ona da 30'un 1. seviyesinin gösterdiği yeri göstererek şekilde ayarlanır.



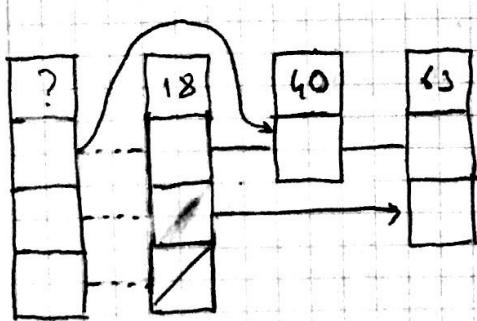
+ Head'in 3. ve 2. seviyesinin linki, 8'den büyük bir sayı gösterdiğinde 1. seviyesine inince 8'i buluyorsa. Bu yozdur. head'in birinci seviyesi 8'den bir sonraki yeri göstererek şekilde ayarlıyoruz.



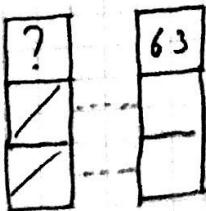
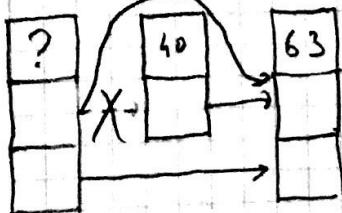
\* 39'un 3. seviyesi null olduğu için  
2. seviyeye geçtilik, 47 olduğu için  
47'nin gösterdigini göstererek yani null,  
39'in birinci dergesi 40'i işaret ettiği için  
40'a geçtilik, 40'in ikinci dergisi 47'si gösterdiği  
için, artık 47'nin gösterdigini yani, 63'ü  
gösterir.



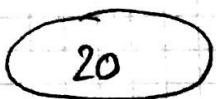
► 18'in 3. ve 2. seviyesi 39'u gösterdiğii  
için ve onda null olduğu için artık 18'inki  
de null olur. 1. seviyeye geçince 39'ya  
40'ı göstericek. (39'un gösterdig.)



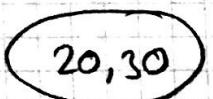
\* Head'in 3. ve 2. elementi 18'i gösterdiğii  
için onda 18'in gösterdigini yani null'ı.



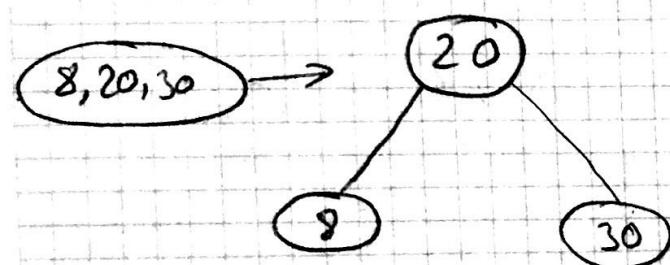
## 2-3 Tree



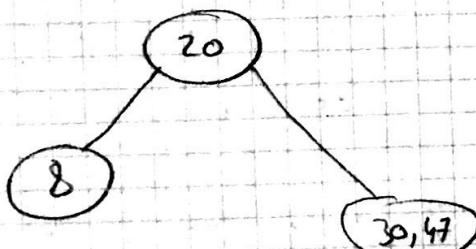
→ 20 eklendi



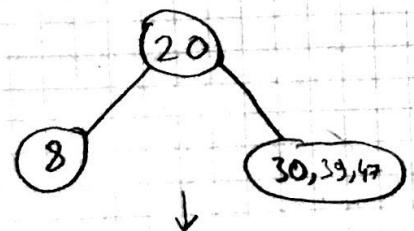
→ 30 yanına geldi



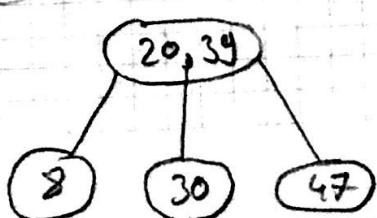
→ 3'ü sona olursa ortaca değer  
ust node'a gitti sağları sola  
ve sağa yükseli

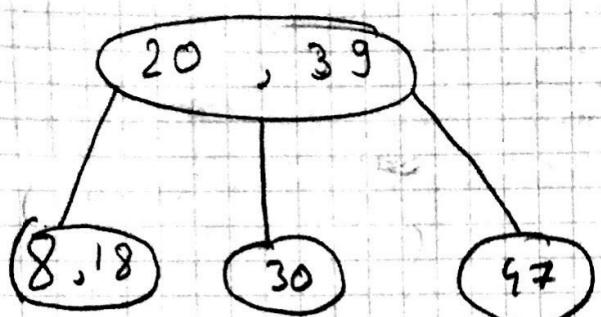


→ 47 30'ın sağına yerleştirili

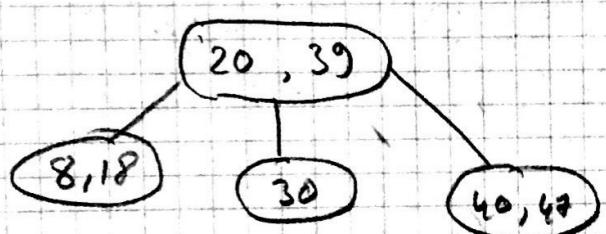


\* 39'ye 3'ü oldu burası 12'n  
olduğu için 39 ortaca eklen olsun  
1cm yukarı çıkarılı

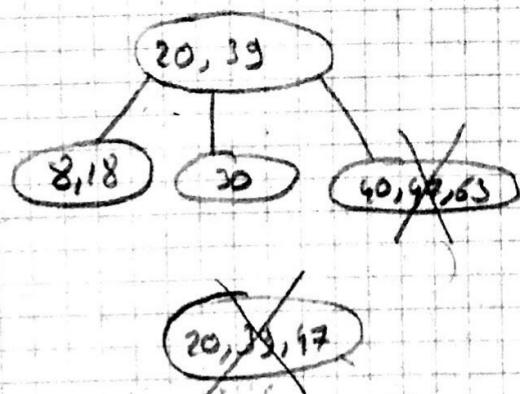




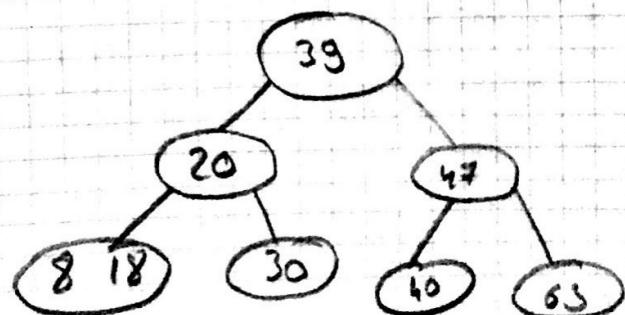
- 18 geldi, 8'in yanına yerlesti.



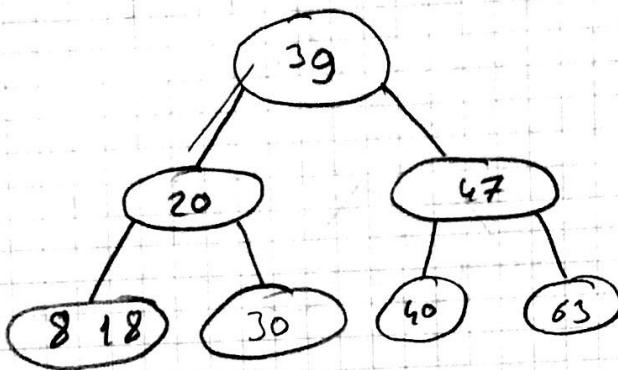
- 40 geldi. 40'in sağını yerlesti.



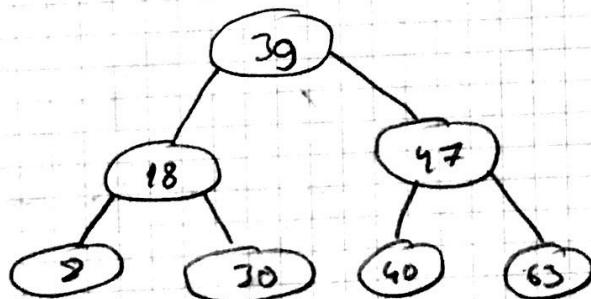
→ 63 geldi sonralı node oluşturdu, oradan  
daha 39 value çıktı, oradan da sonralı  
node oluşturdu orhancı daha 39 tekrar  
iste çıktı. 20 ve 47 diye söyleyişildi.



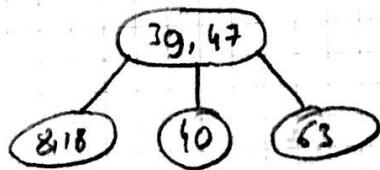
## 2-3 Tree Remove



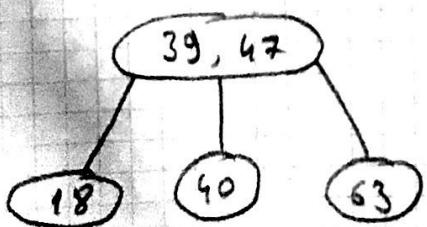
\* 20'yi silerken sol oğlunun  
en büyükü 18'i yerine yapsak  
yapı bozulur.



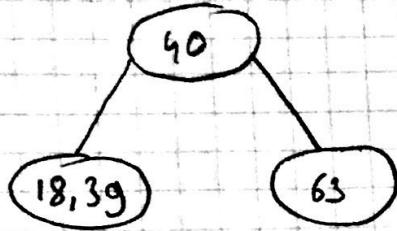
\* 30'u silerken, 8 ile 18  
birleşti, ardından 47'eden yanına  
gikerek, 3 Node'a düştü.



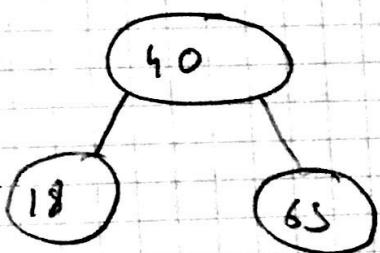
\* 8'in silinmesi yapımı bozmazca



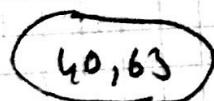
\* 47 siliniken sol en büyükü  
40'ı yerine geteri yapısı bozulur.  
39' 40'dan büyük olduğunu için  
sağ alt eşiği 18'er.



\* 39'un silinmesi yapıldı, bozmadı



\* 18 silinince 2 node kaldığı için  
bozmadı

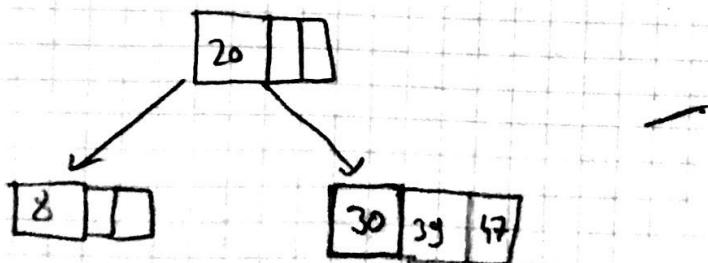
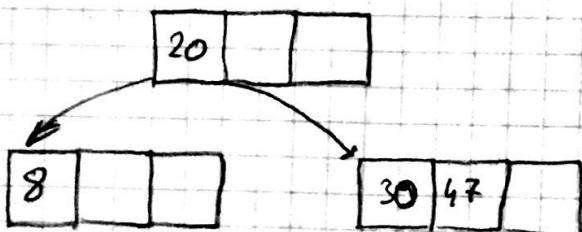
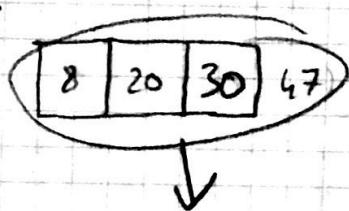
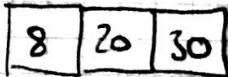
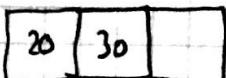


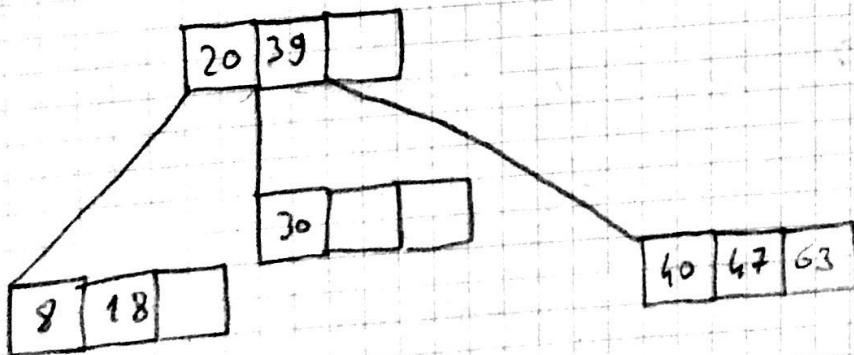
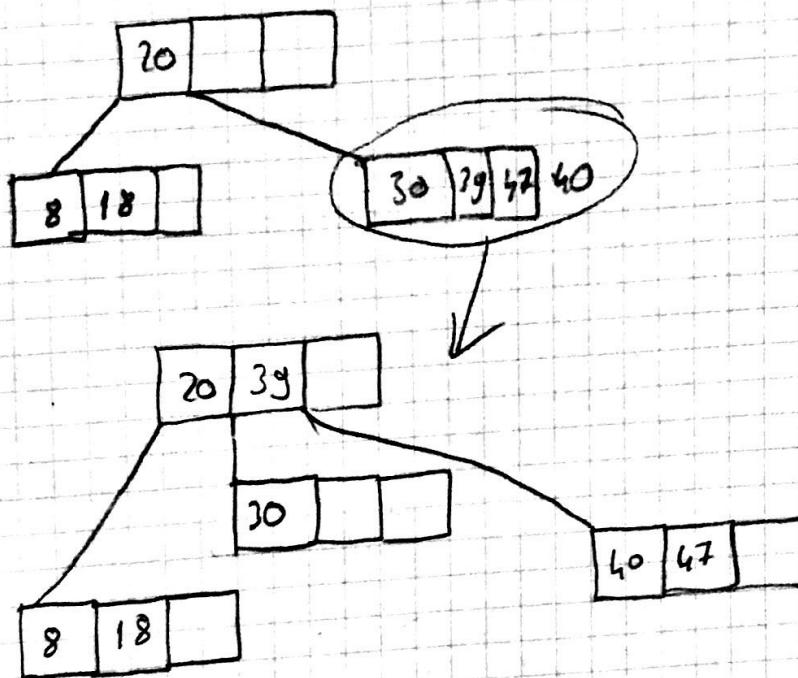
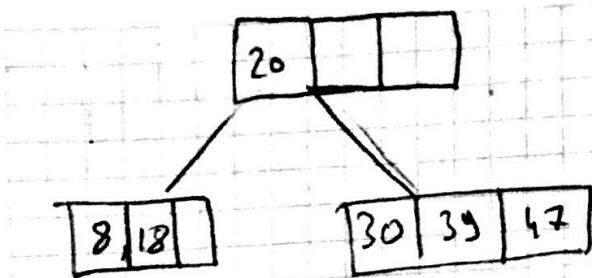
\* 40 silinince 63 tek kalmış.  
Ardından 63 silinince boş kalmış.



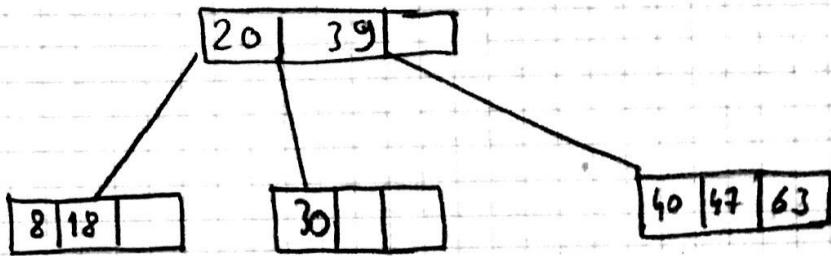
## B - Tree

$$M=4 \quad \text{keys} = M-1 = 3$$

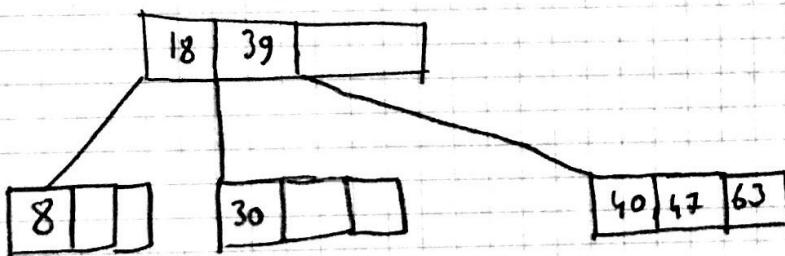




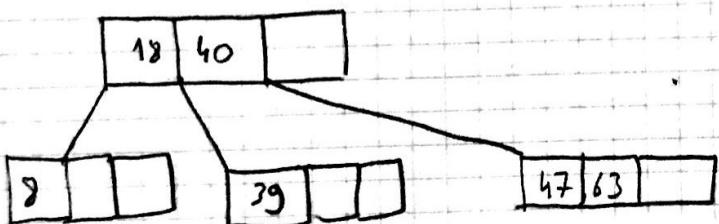
## B Tree with order 4 Remove



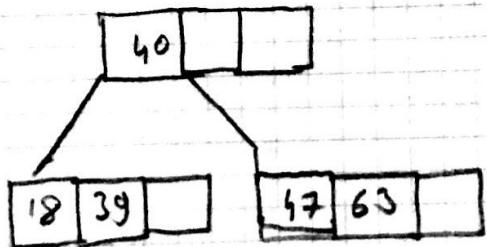
\* 20 siliniken yerine  
en büyük sol değer 18 gelir.



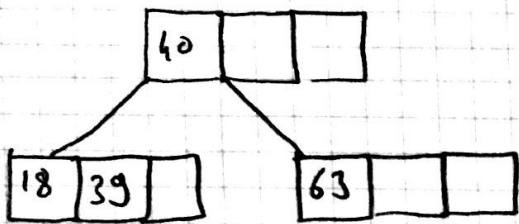
\* 30 siliniken 30'un yerine  
bir büyük 39 gelir onun  
yerine ise 2 büyük 40  
gelir.



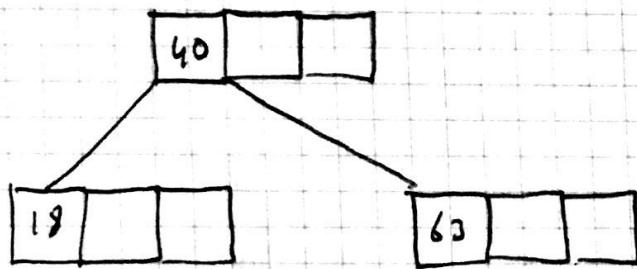
\* 8 silinince yerine bir büyük  
18 geçince root Lek'e dots  
3 çocuk yerine 2 çocuk olmalı  
39' 18' in young olsun



\* 47'in silmesi yapılmaz



\* 39'un da silinecek olası  
yapayı bozma.



\* 18'in silinecek olası yapayı  
bozacağı için 40 ve 63'un  
yeri yerler

