

Veritabanı Yönetim Sistemleri

Dr. Öğr. Üyesi Ahmet Arif AYDIN

B+ Tree

Dosya Yapıları

- Page (sayfa)
- Files layer (dosya katmanı)
- Disk space manager (disk alan yöneticisi)
- Buffer manager (ara bellek yöneticisi)

Index yapıları

- Hash based indexing
- Tree based indexing

B+ Ağaç Yapısı

- Esnek ve dinamik bir yapısı bulunmaktadır.
- Her bir **düğüm** disk üzerinde bir **sayfa**dır.
- İki adet düğüm çeşidi bulunmaktadır:
 - Yaprak Düğümler (leaf nodes)
 - Yaprak olamayan düğümler (non-leaf nodes)

B+ Ağaç Yapısı: Yaprak Düğümler

```
struct YaprakNode {  
    vector<Key> keys;  
    vector<Value> values;  
    PagePointer sonraki_sayfa;  
}
```



B+ Ağaç Yapısı: Yaprak Düğümler

```
struct YaprakNode {  
    vector<Key> keys;  
    vector<Value> values;  
    PagePointer sonraki_sayfa;  
}
```

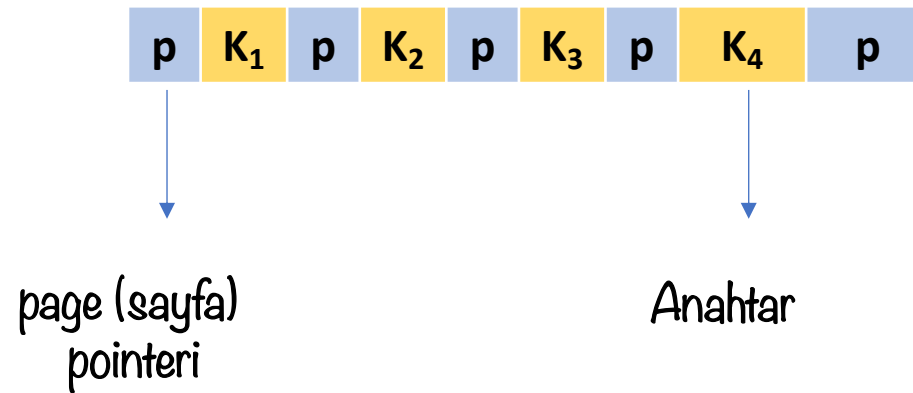


Verinin kaydedildiği sayfalar birbirine bağlı **linkedlist** ile tanımlanabilir.

Düğümelerde bulunan anahtarlar (key) sıralıdır.
Yapraklarda (leaf) bulunan değerlerde sıralıdır.

B+ Ağaç Yapısı: Yaprak Olmayan Düğüm

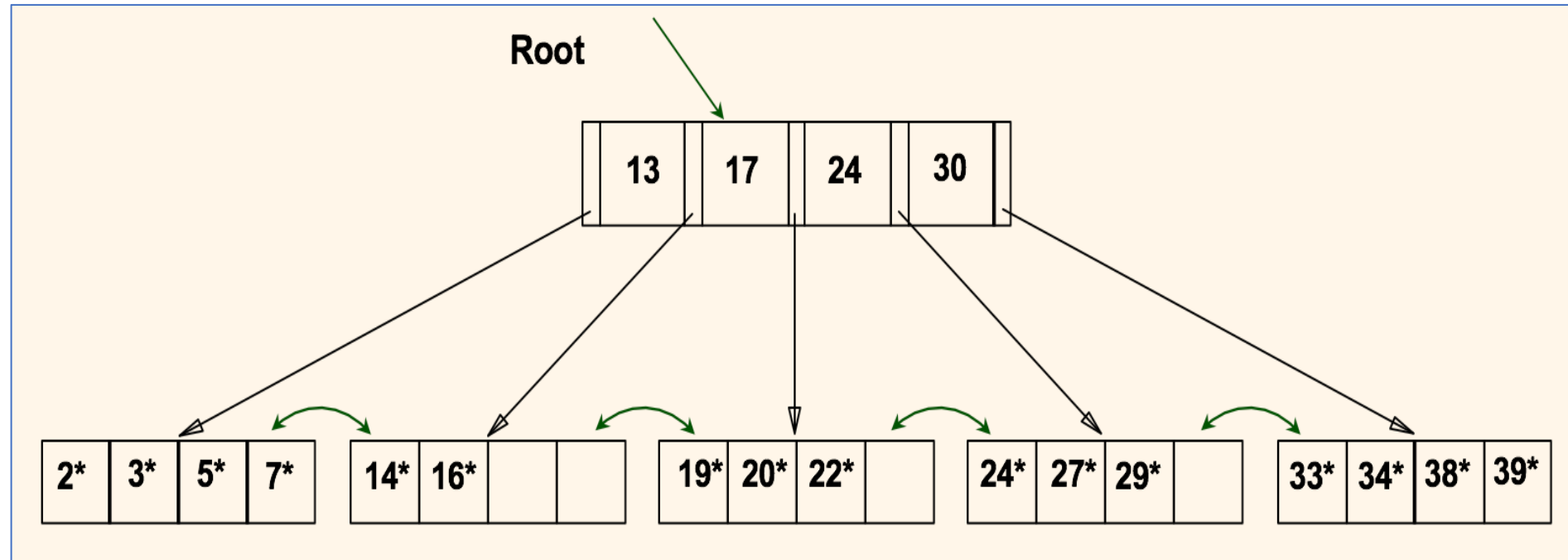
```
struct InteriorNode {  
    vector<Key> keys;  
    vector<PagePointer> pointers;  
}
```



Pointer sayısı anahtar
sayısından 1 fazla

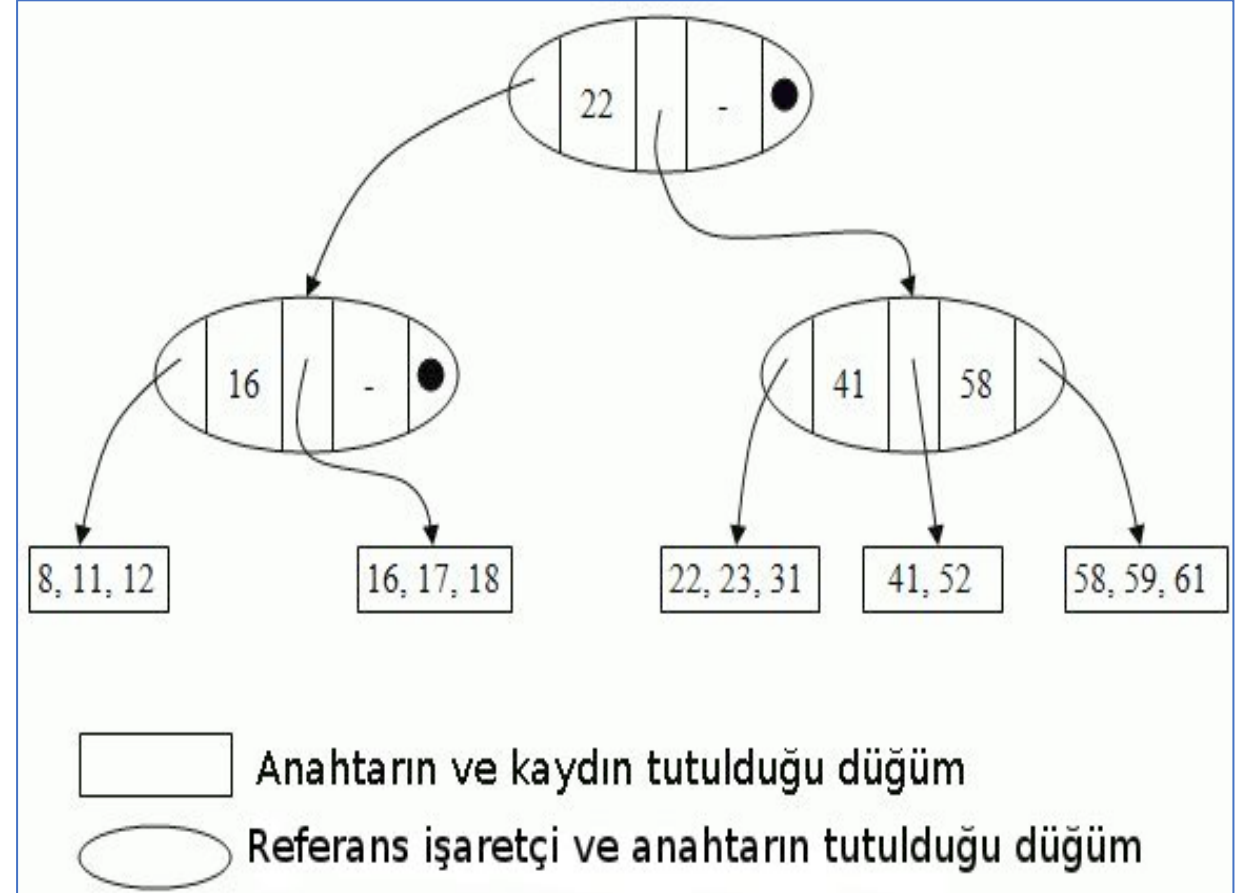
B+ Ağaç Yapısı

- B+ ağacının yapısının **pointer**'i node aramayı yönlendirmektedir.
- Leaf nodes (yaprak düğümleri) kaydedilen verileri içermektedir.
- B+ ağacının yapısı eklenen veya silinen anahtara göre dinamik olarak değişir.



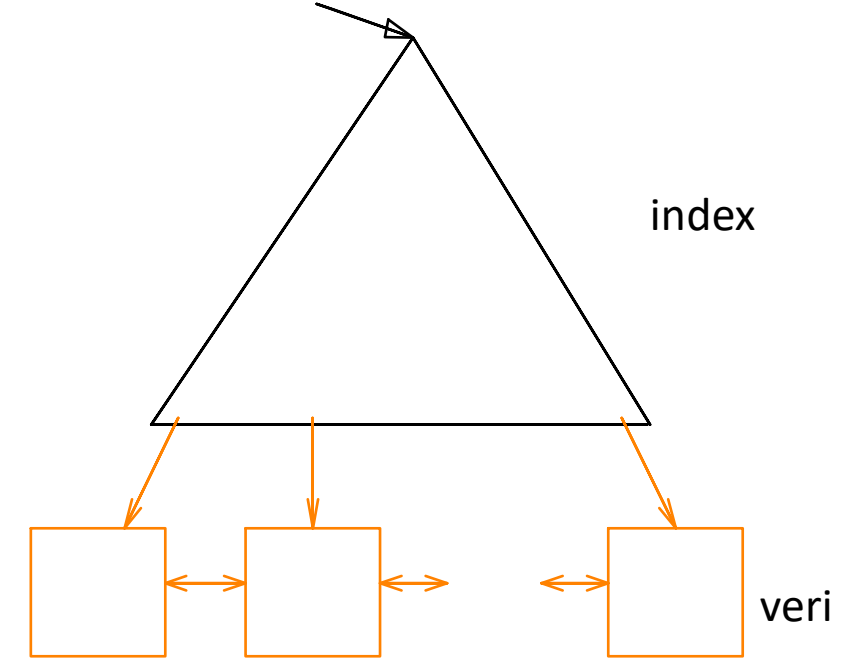
B+ Ağaç Yapısı

- ❑ B+ Tree dinamik olarak büyüyüp (grow) küçülür (shrink) ve ağacın yüksekliği de dinamik olarak değişir.
- ❑ Anahtarlar ve işaretçileri (pointer) yaprak olmayan düğümlerde (non leaf node) tutulur.



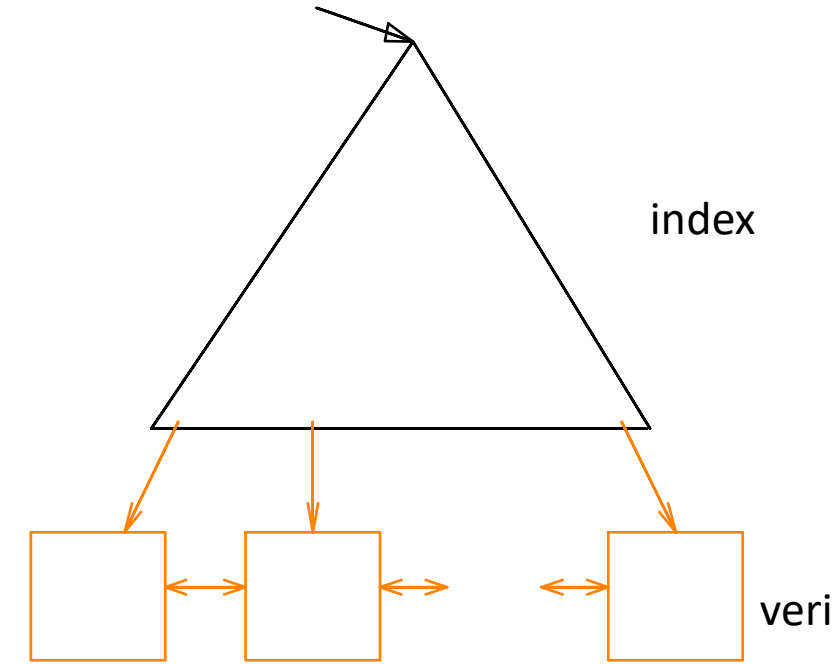
B+ Ağaç Yapısının Temel Karakteristikleri

- Bir tane root (ana düğüm) bulunmak zorundadır.
- Her bir düğüm için (root hariç) ortalama %50 doluluk oranı istenmektedir
 - $d \leq$ düğümde bulunabilecek anahtar sayısı $\leq 2d$
 - d parametresi (order of tree): bir düğümde bulunabilecek minimum kayıt sayısını belirler
 - d bir sayfaya kaydedilen kayıtların kapladığı alana göre belirlenir
- B+ ağacının yüksekliği: $h = \log_k N$
 - k = düğümün kapasitesi, N = kayıt sayısı



B+ Ağacı : Pratikteki değerler

- Tipik order 100 ve doluluk oranı % 67
 - Düzgümlerde bulunan ortalama alt düğüm sayısı (fanout) 133
- Kapasite
 - Yükseklik 4: $133^4 = 312,900,700$ kayıt
 - Yükseklik 3: $133^3 = 2,352,637$ kayıt
- Ara bellekte tutulan değerler :
 - Seviye 1 = 1 page = 8 Kbytes
 - Seviye 2 = 133 pages = 1 Mbyte
 - Seviye 3 = 17,689 pages = 133 MBytes



B+ Ağaç Yapısı: Arama (search) işlemi

1. Aramaya **root (kök düğüm)** ile başlanır
2. Arama yapılan düğüm (node) bir yaprak değilse;
 - Anahtar değerine göre düğümde arama yapılır
 - Aranılan **anahtar değerden küçük olan en yüksek değerli anahtardan sonraki** işaretleyicinin gösterdiği düğüme gidilir.
 - Arama yapılan düğüm güncellenir
 - 2.adıma geri dönülür.
3. Arama yapılan node (düğüm) bir **yaprak ise** aranılan deger bu düğümdedir veya yoktur.

B+ Ağaç Yapısı: Arama (search) işlemi

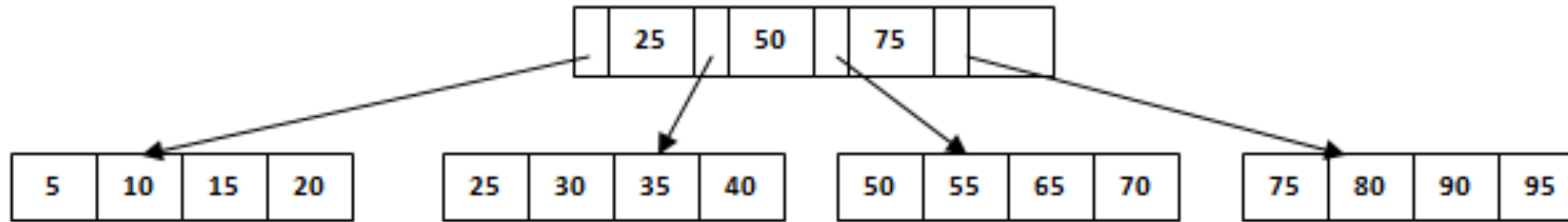
Find record with search-key value V

1. $C = \text{root}$
2. While C is not a leaf node {
 1. Let i be least value s.t. $V \leq K_i$.
 2. If no such exists, set $C = \text{last non-null pointer in } C$
 3. Else { if ($V = K_i$) Set $C = P_{i+1}$ else set $C = P_i$ }}
3. Let i be least value s.t. $K_i = V$
4. If there is such a value i , follow pointer P_i to the desired record.
5. Else no record with search-key value k exists.

B+ Ağaç Yapısı: Arama (search)

65 değerini arayalım

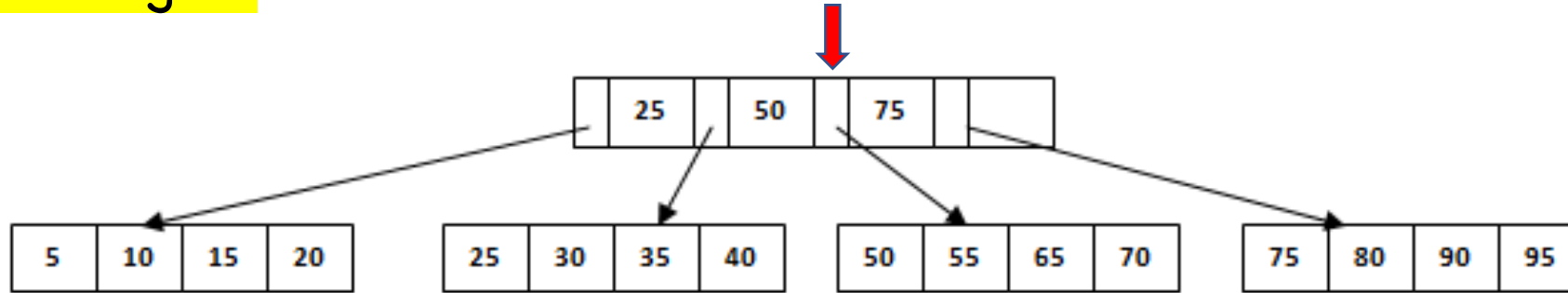
<https://www.tutorialcup.com/dbms/b-tree.htm>



B+ Ağaç Yapısı: Arama (search)

65 değerini arayalım

<https://www.tutorialcup.com/dbms/b-tree.htm>

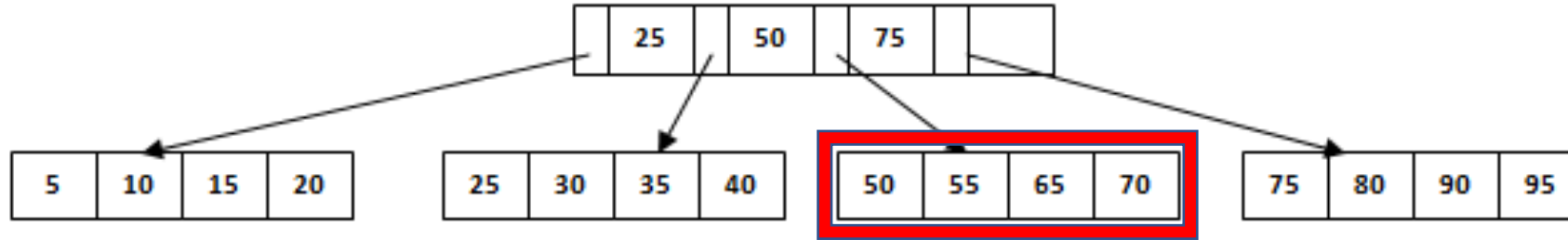


1. 65 değerinden küçük olan en büyük anahtardan sonraki pointera gidilir

B+ Ağaç Yapısı: Arama (search)

65 değerini arayalım

<https://www.tutorialcup.com/dbms/b-tree.htm>

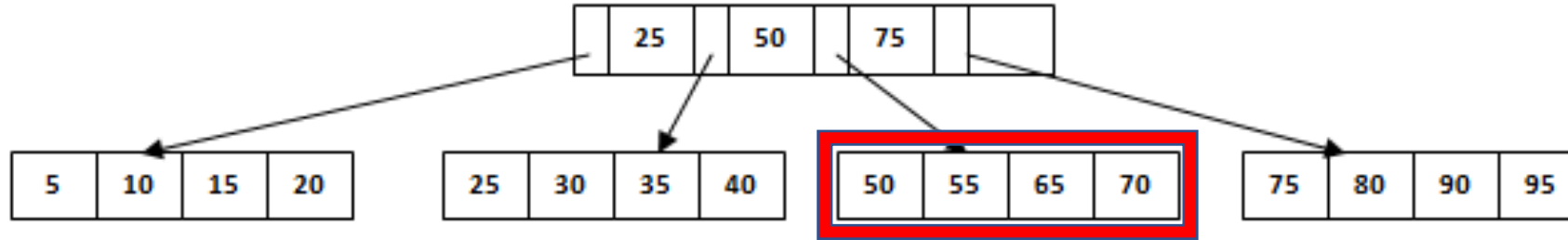


1. 65 değerinden küçük olan en büyük anahtardan sonraki pointera gidilir
2. Bulunan düğüm yaprak (leaf) olduğu için bu node içerisinde bulunan kayıtlar seri olarak (sequential) olarak taranır.

B+ Ağaç Yapısı: Arama (search)

65 değerini arayalım

<https://www.tutorialcup.com/dbms/b-tree.htm>



1. 65 değerinden küçük olan en büyük anahtardan sonraki pointera gidilir
2. Bulunan düğüm yaprak (leaf) olduğu için bu node içerisinde bulunan kayıtlar seri olarak (sequential) olarak taranır.

Arama (search) işlemi gerçekleştirilirken
insert, delete ve update
İşlemlerine izin verilmemektedir

B+ Ağaç Yapısı: Ekleme (insert) işlemi

1- Eklenmesi gereken kayıt için (düğüm) uygun düğüm aranır

2-Eğer düğüm maximum kaydedebileceği değerden az kayıt bulunduruyorsa kayıt sırası dikkate alınarak ekleme yapılır (işlem tamamlanır)

3- Eklenecek değer için düğümde yer yoksa

- Düğüm ikiye ayrılır, yeni bir yaprak (leaf) oluşturulur
- elemanların yarısı sırası değişmeden oluşturulan yeni yaprağa eklenir.
- *Yeni oluşturulan düğümün anahtarı bir üst seviyedeki düğüme eklenir*
- *Üst seviyedeki düğüm dolu ise ikiye ayrılır ve ortadaki anahtar üst seviyeye alınır.*
- Bölünme gerektirmeyen düğüm oluşuncaya kadar devam eder.

4- Eğer root (kök) düğümünün bölünmesi gerekiyorsa tek bir değeri ve iki pointeri bulunan yeni bir root oluşturulur.

B+ Ağaç Yapısı: Ekleme (insert) işlemi

Root (kök) düğüm istisnadır.
Kökte olabilecek anahtar sayısı

$$1 \leq \text{anahtar sayısı} \leq 2d$$

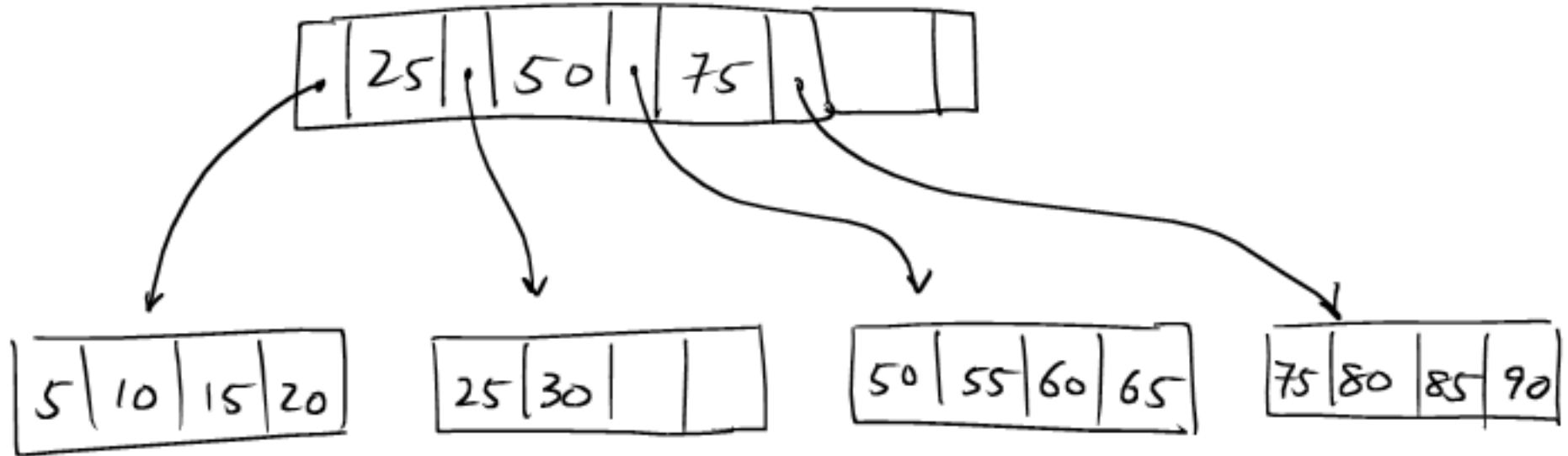
B+ ağacının kök olmayan düğümlerin de
bulunan anahtar sayısı

$$d \leq \text{anahtar sayısı} \leq 2d$$

B+ Ağaç Yapısı: Ekleme (insert) işlemi

28 değerini ekleyelim

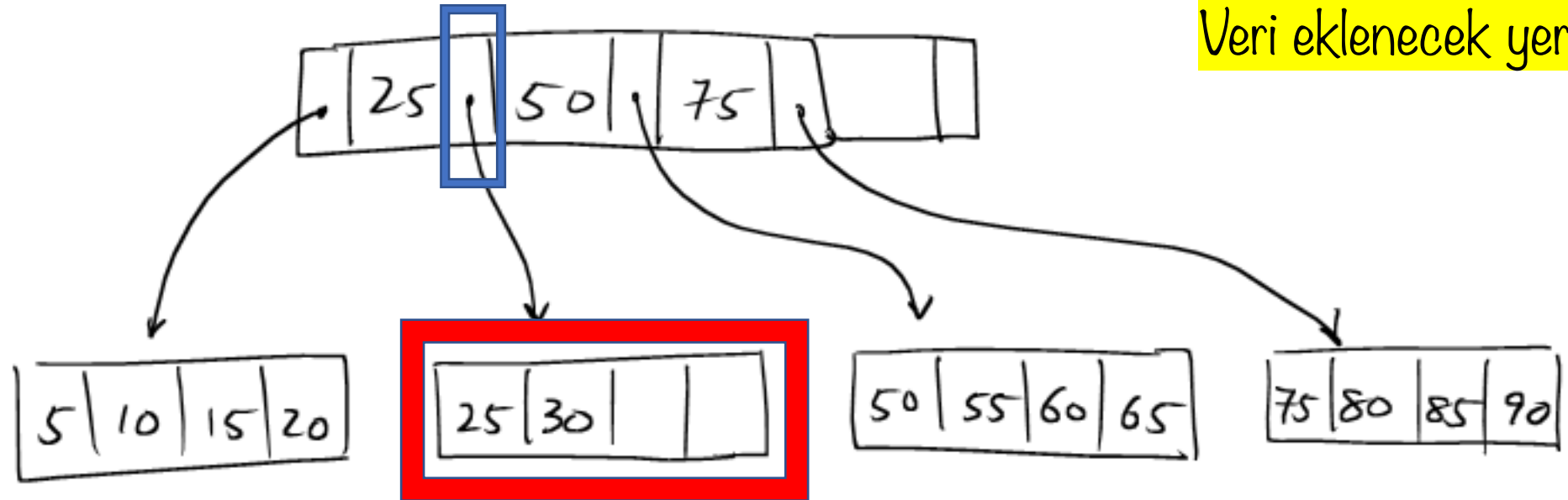
Ekleyeceğiniz değerden küçük olan en büyük değerden sonraki pointeri bul



B+ Ağaç Yapısı: Ekleme (insert) işlemi

28 değerini ekleyelim

Search (root, 28)



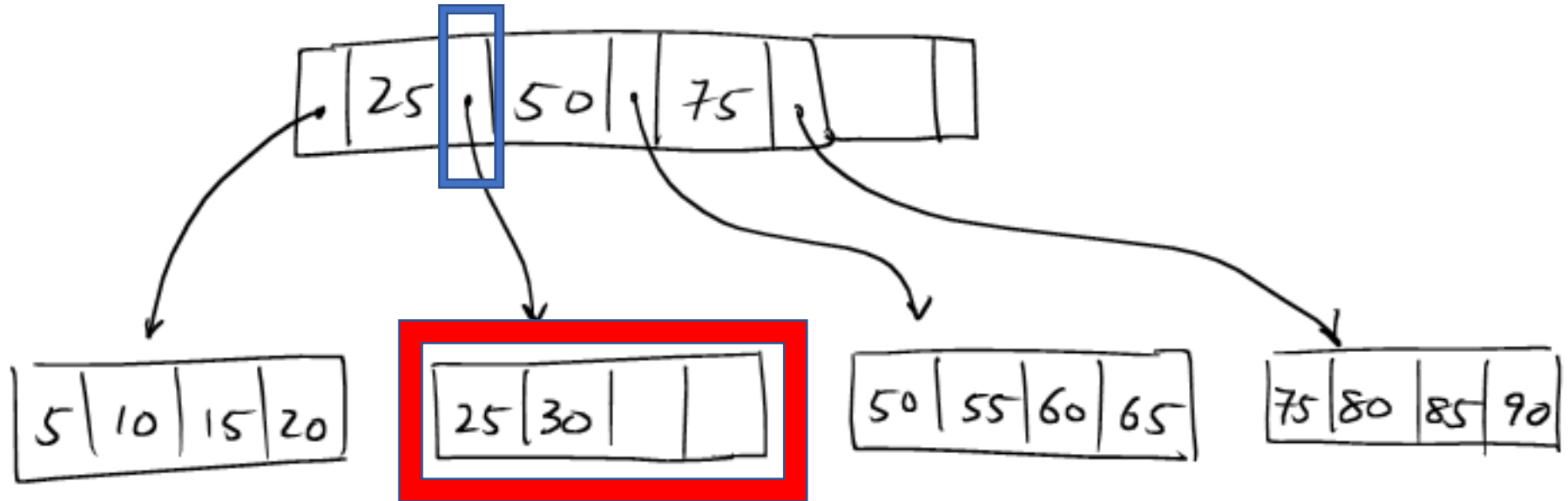
Pointer'in gösterdiği
yaprak düğüm mü?

Veri eklenecek yer var mı?

B+ Ağaç Yapısı: Ekleme (insert) işlemi

28 değerini ekleyelim

Search (root, 28)



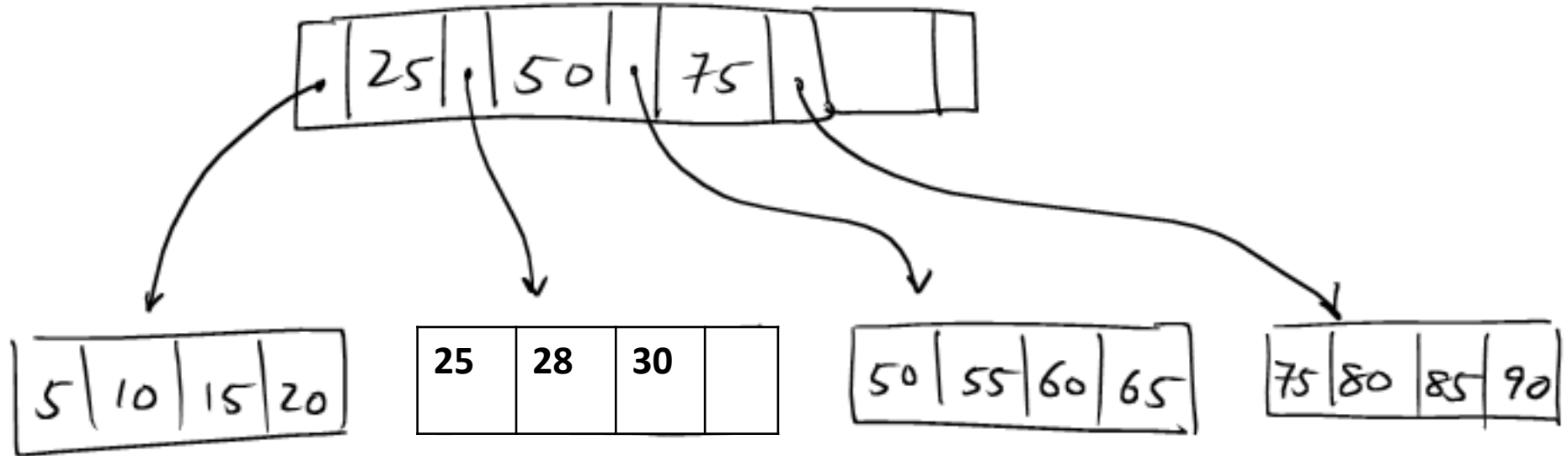
Yaprak düğüm ve veri eklenecek alan var !

B+ Ağaç Yapısı: Ekleme (insert) işlemi

28 değerini ekleyelim

Search (root, 28)

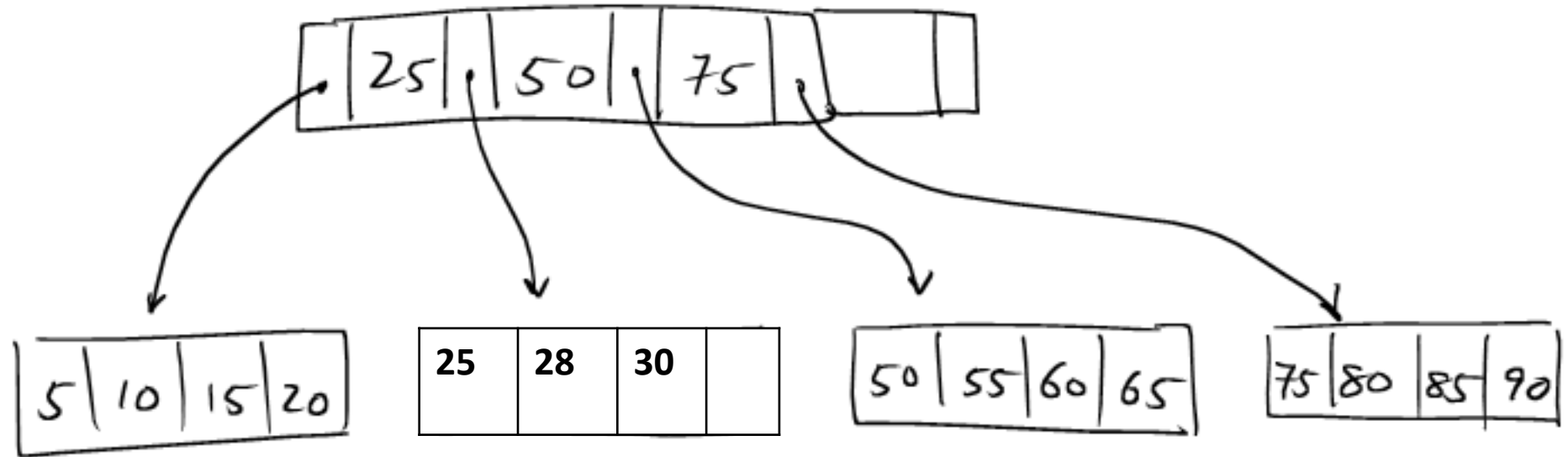
Ekleyeceğiniz değerden küçük olan en büyük değerden sonraki pointeri bul



Yaprak düğümde yer olduğundan sayı eklendi ve yaprakda bulunan değerler kendi içinde sıralandı

B+ Ağaç Yapısı: Ekleme (insert) işlemi

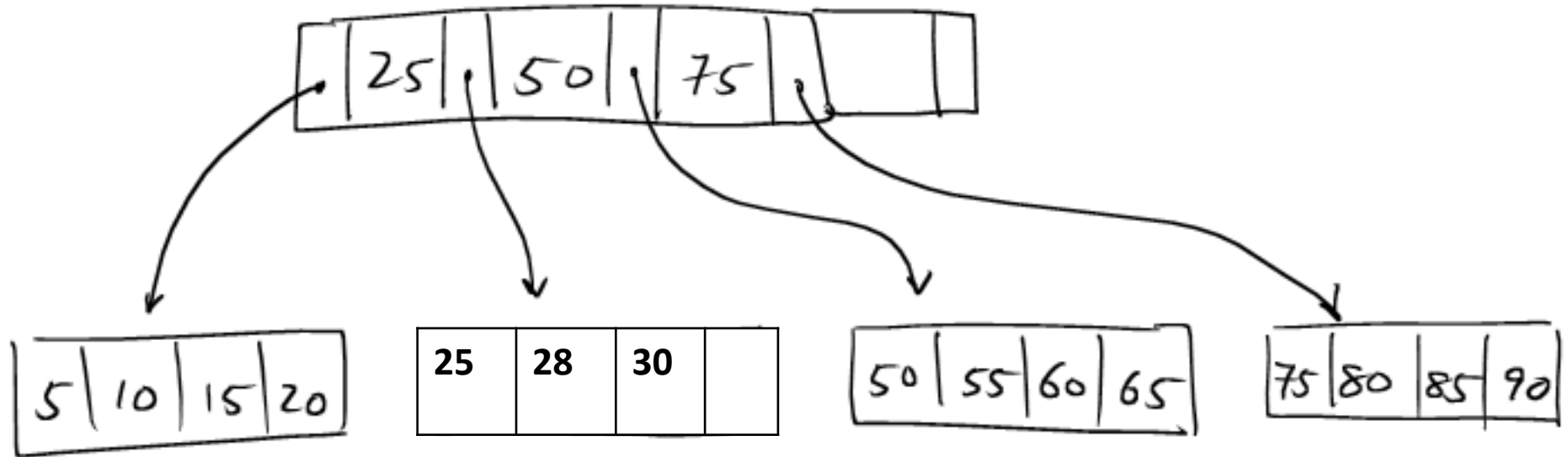
70 değerini ekleyelim



B+ Ağaç Yapısı: Ekleme (insert) işlemi

70 değerini ekleyelim

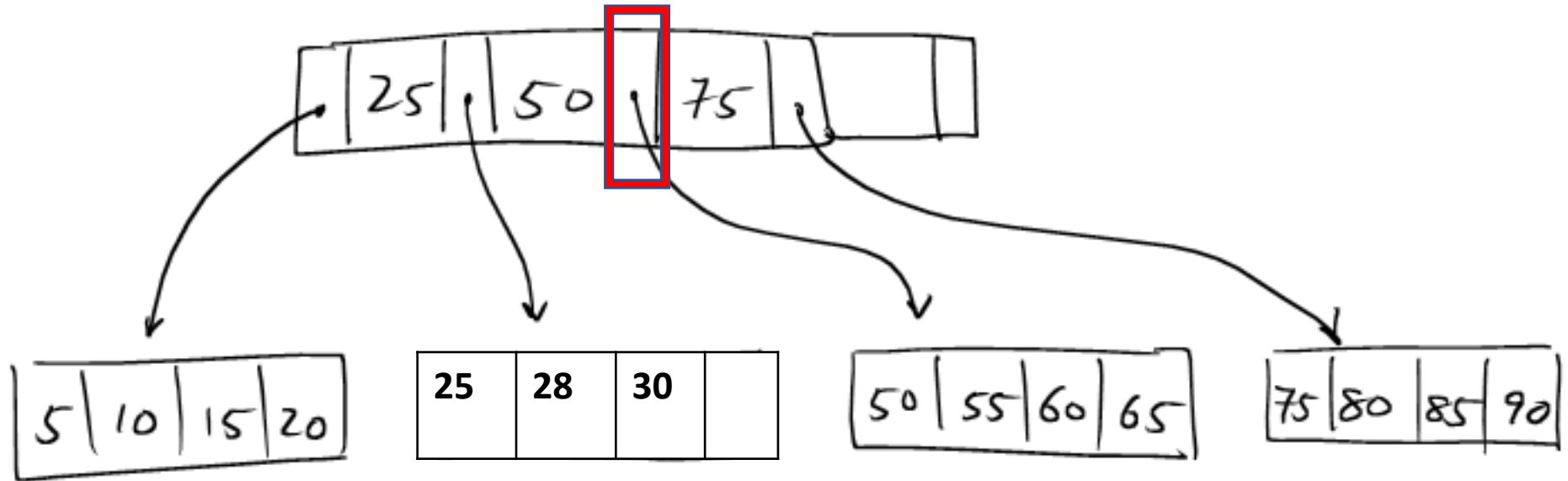
Ekleyeceğiniz değerden küçük
olan en büyük değerden
sonraki pointeri bul



B+ Ağaç Yapısı: Ekleme (insert) işlemi

70 değerini ekleyelim

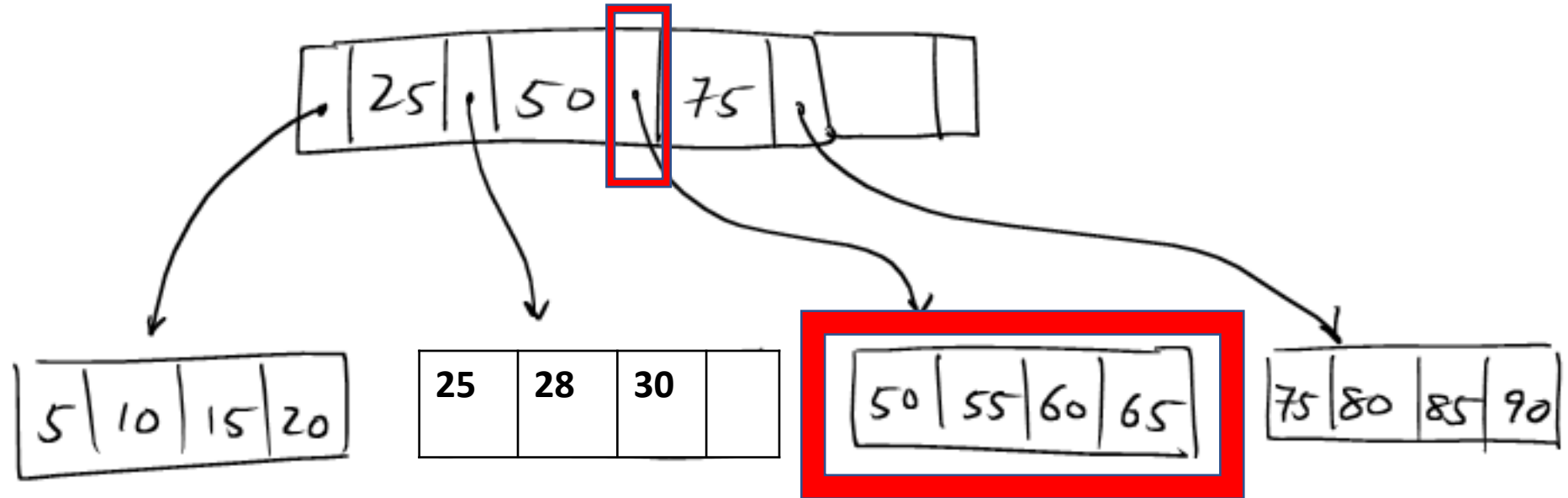
Ekleyeceğiniz değerden küçük
olan en büyük değerden
sonraki pointeri bul



B+ Ağaç Yapısı: Ekleme (insert) işlemi

70 değerini ekleyelim

Ekleyeceğiniz değerden küçük
olan en büyük değerden
sonraki pointeri bul



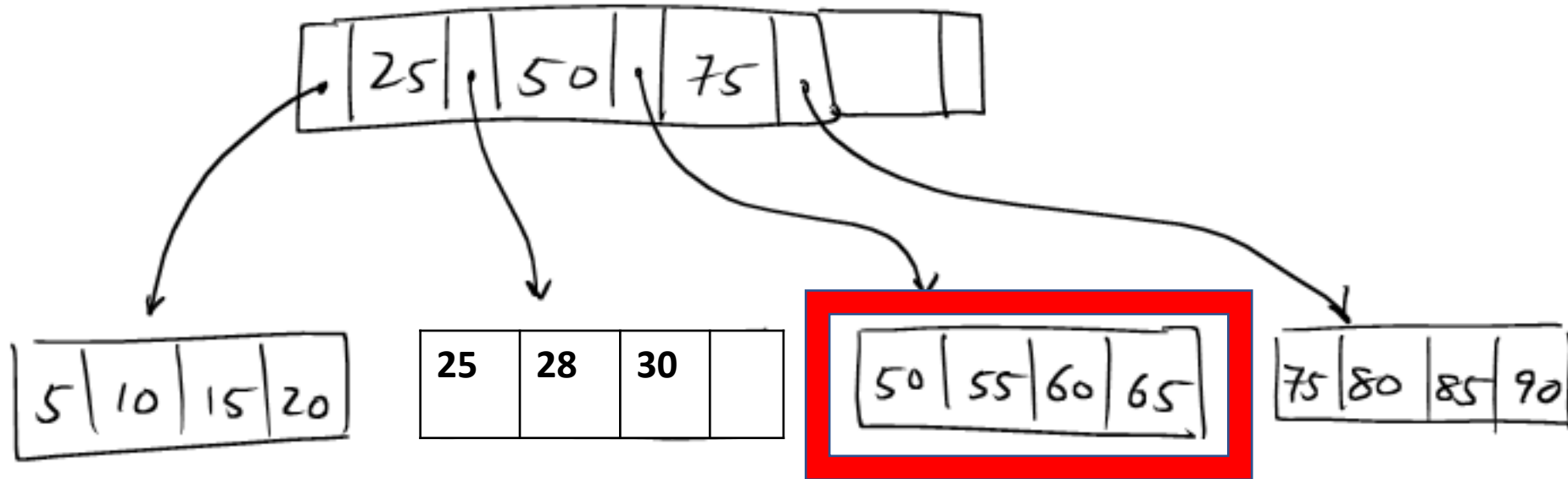
Değerin eklenmesi
gerekten düğüm bulunur.

Yaprak düğümde değer eklenmesi için
eklenmesi gereken alan yok !!!!

B+ Ağaç Yapısı: Ekleme (insert) işlemi

70 değerini ekleyelim

Ekleyeceğiniz değerden küçük
olan en büyük değerden
sonraki pointeri bul



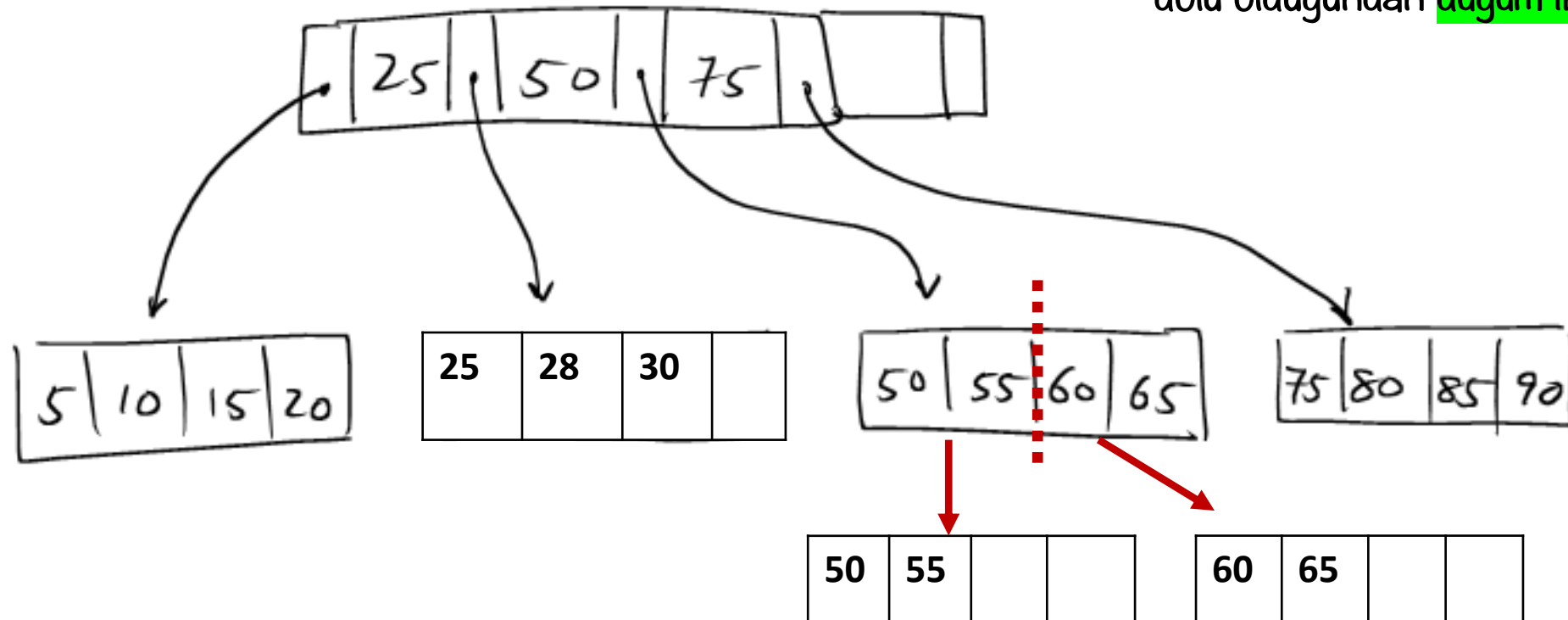
Değerin eklenmesi
gereken düğüm bulunur.

Yaprak düğümde değer eklenmesi için
eklenmesi gereken alan yok !!!!

B+ Ağaç Yapısı: Ekleme (insert) işlemi

70 değerini ekleyelim

Değerin eklenmesi gereken düğüm dolu olduğundan **düğüm ikiye bölünür**

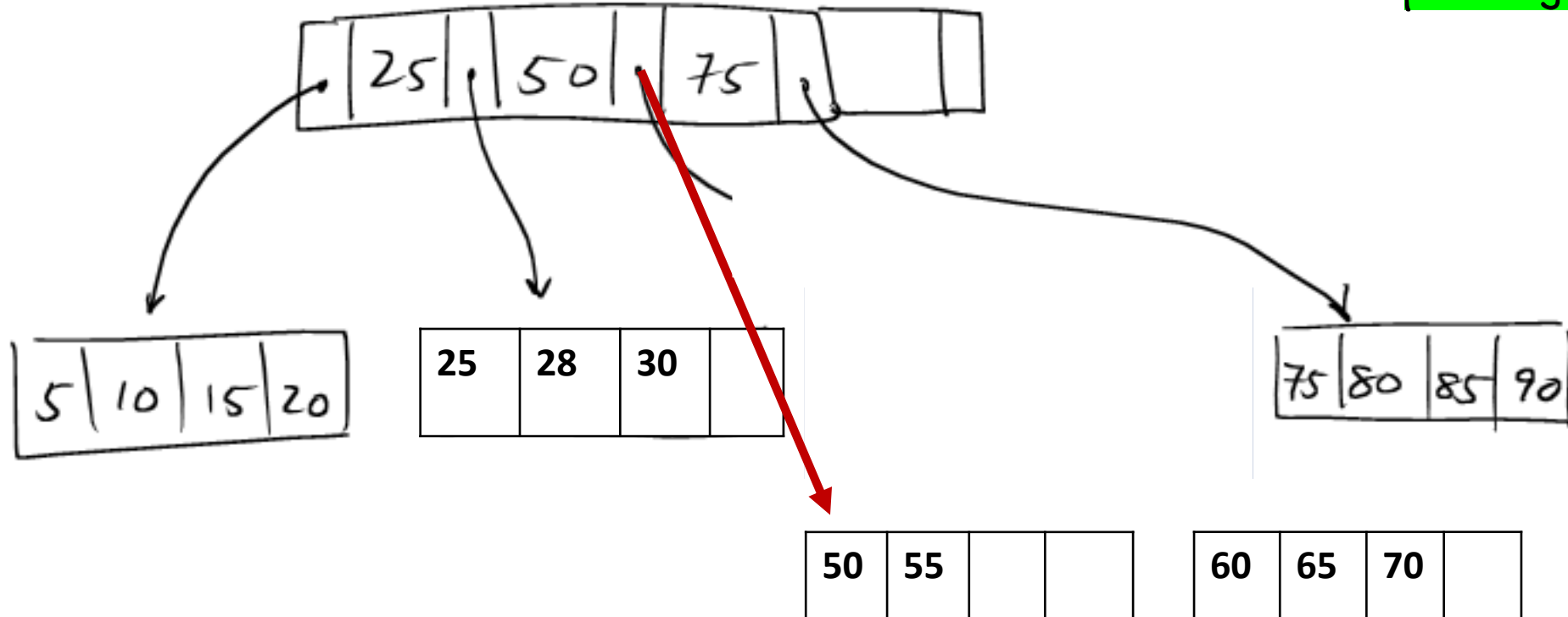


B+ Ağaç Yapısı: Ekleme (insert) işlemi

70 değerini ekleyelim

Oluşturulan yeni düğüme yeni değer eklenir.

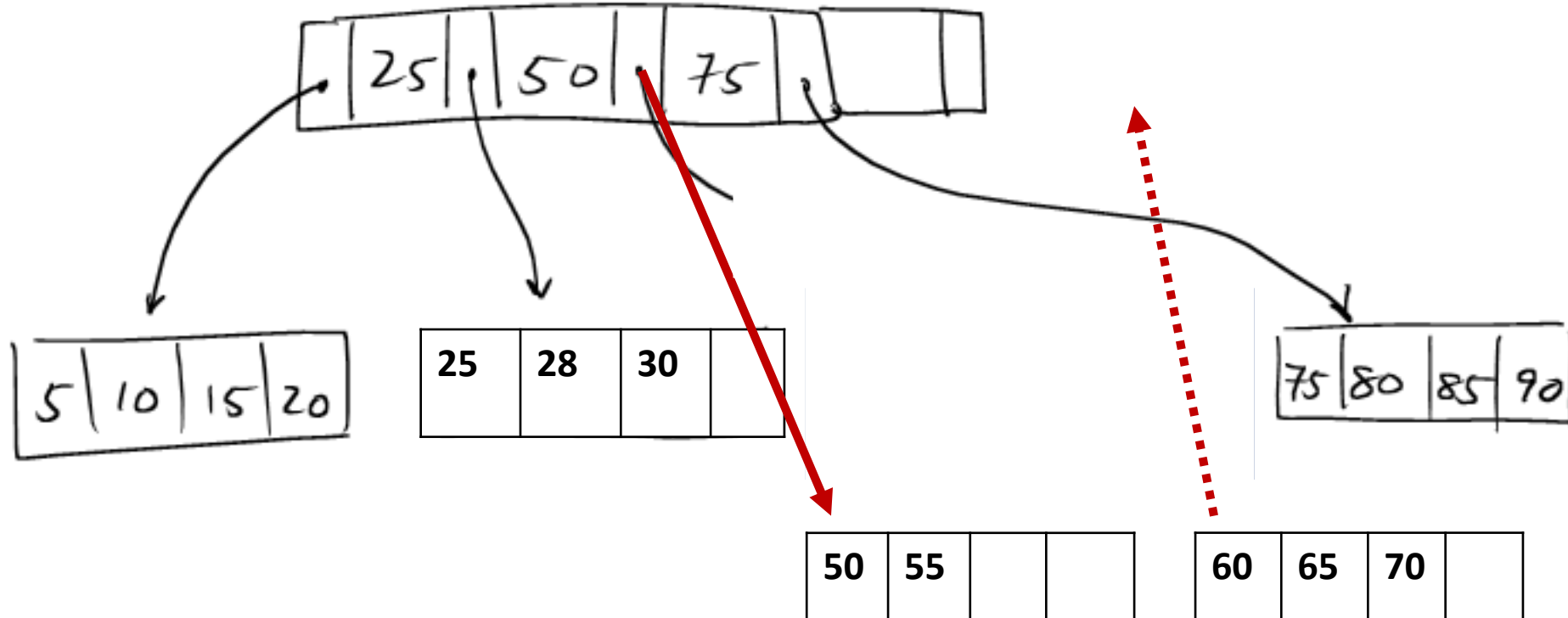
Fakat oluşturulan yeni düğümün pointeri yok !



B+ Ağaç Yapısı: Ekleme (insert) işlemi

70 değerini ekleyelim

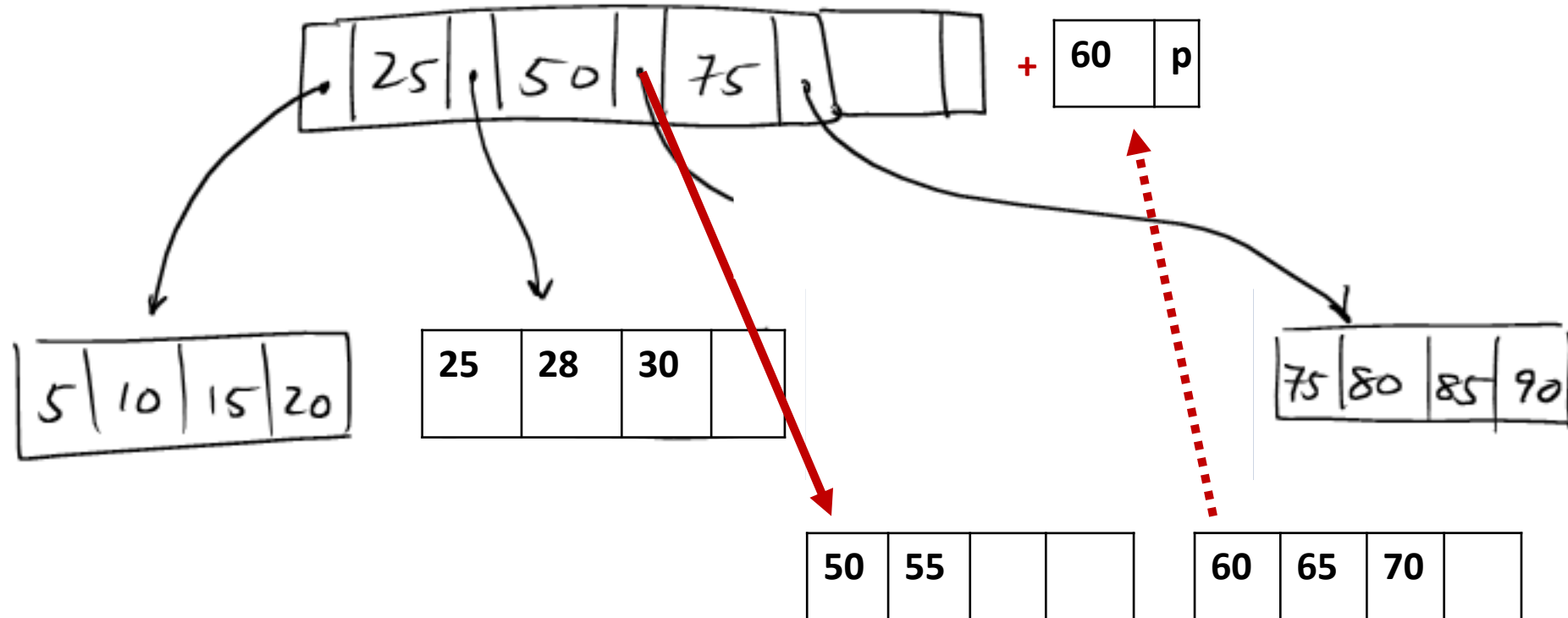
Oluşturulan yeni düğüm için
bir üst seviyede
anahtar ve pointer gerekecek



B+ Ağaç Yapısı: Ekleme (insert) işlemi

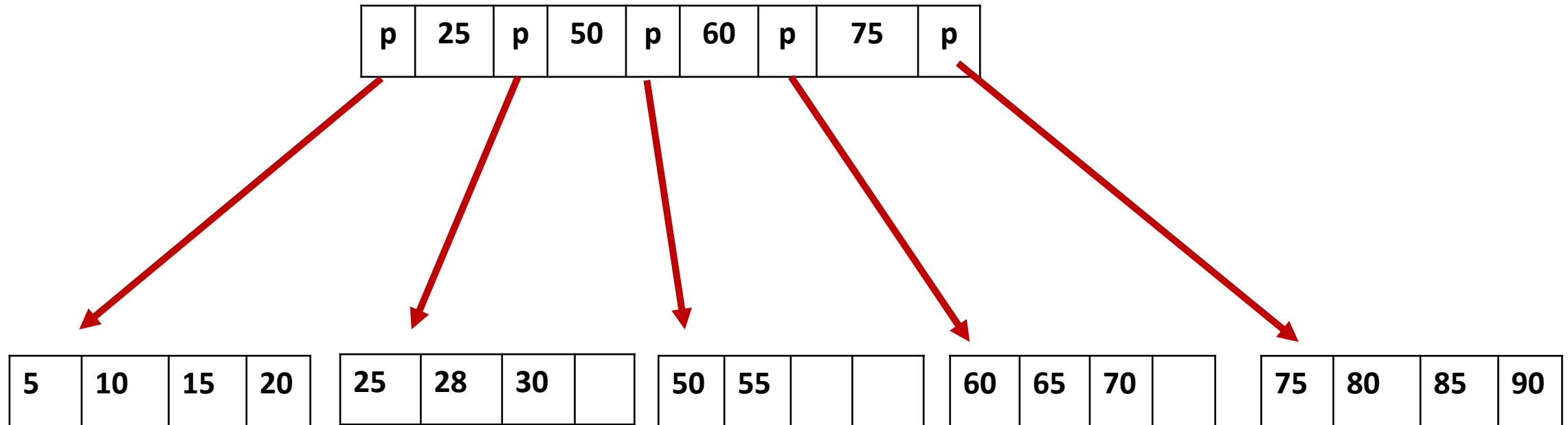
70 değerini ekleyelim

yeni düğümün anahtar ve pointeri
bir üst seviyeye eklenecek



B+ Ağaç Yapısı: Ekleme (insert) işlemi

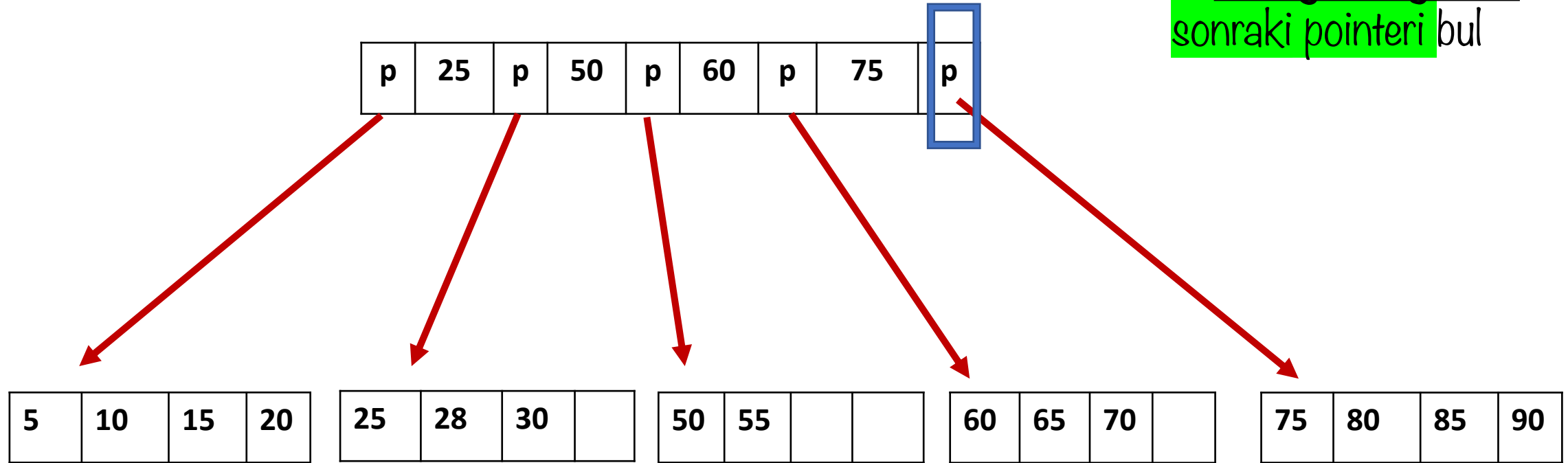
70 değerini ekledikten sonra elde edilen yeni ağaç



B+ Ağaç Yapısı: Ekleme (insert) işlemi

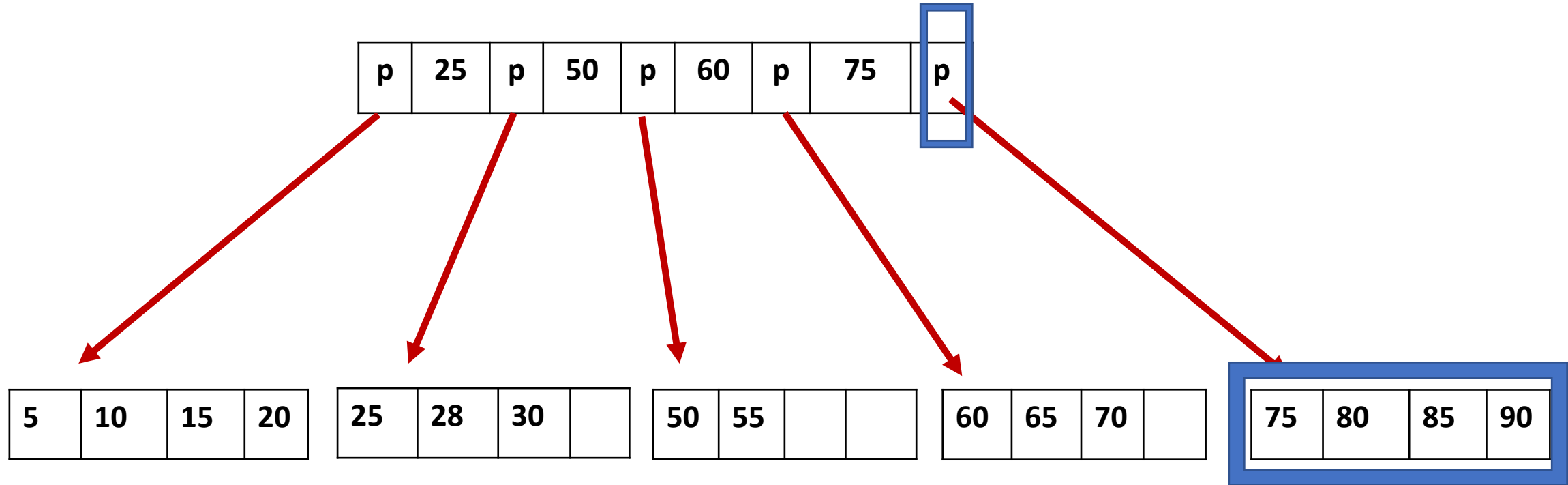
Yeni ağaca 95 değerini ekleyelim

Ekleyeceğiniz değerden küçük olan en büyük değerden sonraki pointeri bul



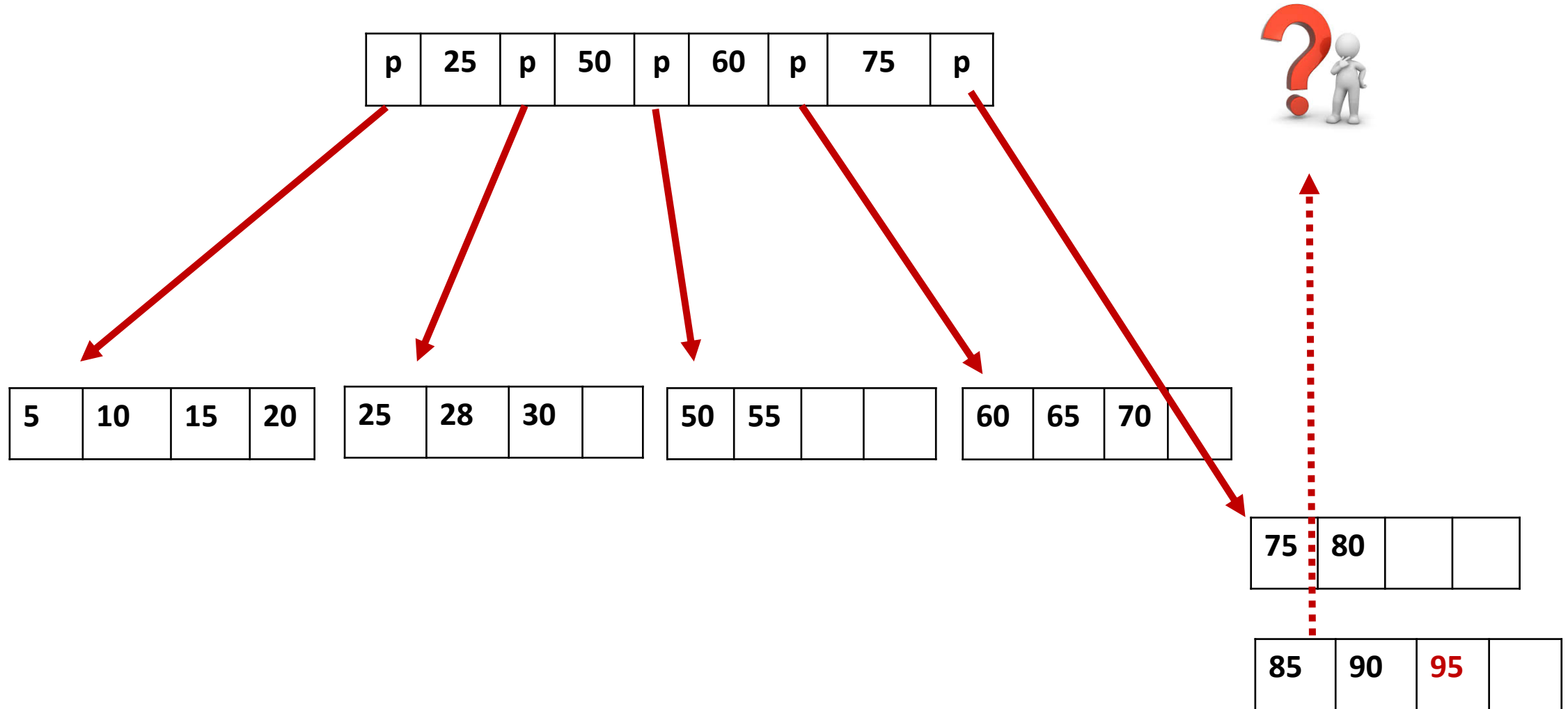
B+ Ağaç Yapısı: Ekleme (insert) işlemi

Yeni ağaca 95 değerini ekleyelim



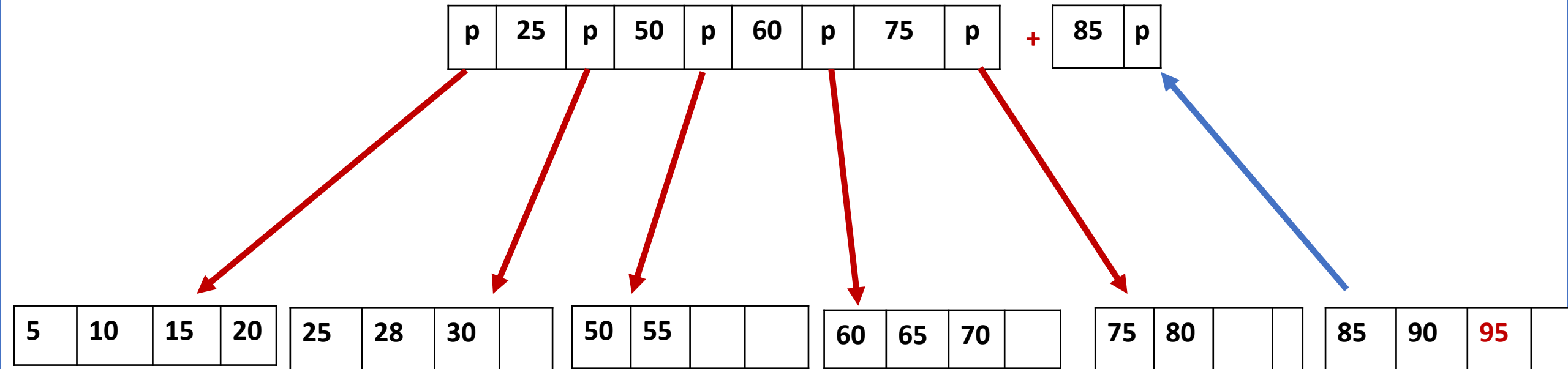
B+ Ağaç Yapısı: Ekleme (insert) işlemi

Yeni ağaca 95 değerini ekleyelim



B+ Ağaç Yapısı: Ekleme (insert) işlemi

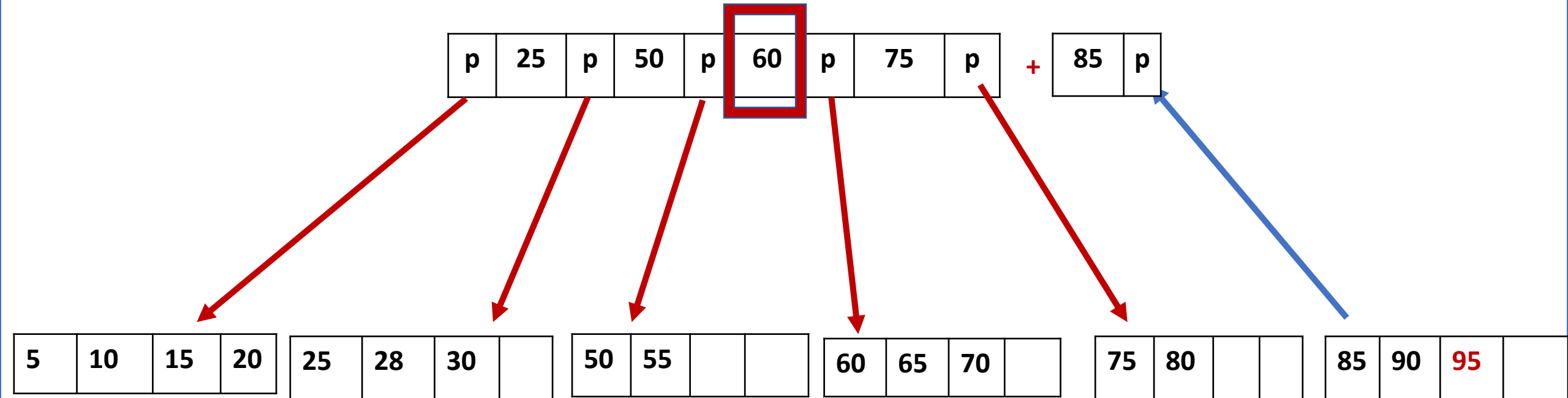
Yeni ağaca 95 değerini ekleyelim



85 değerinin ve pointerinin eklenmesi için alan olmadığından kök düğüm bölünecek

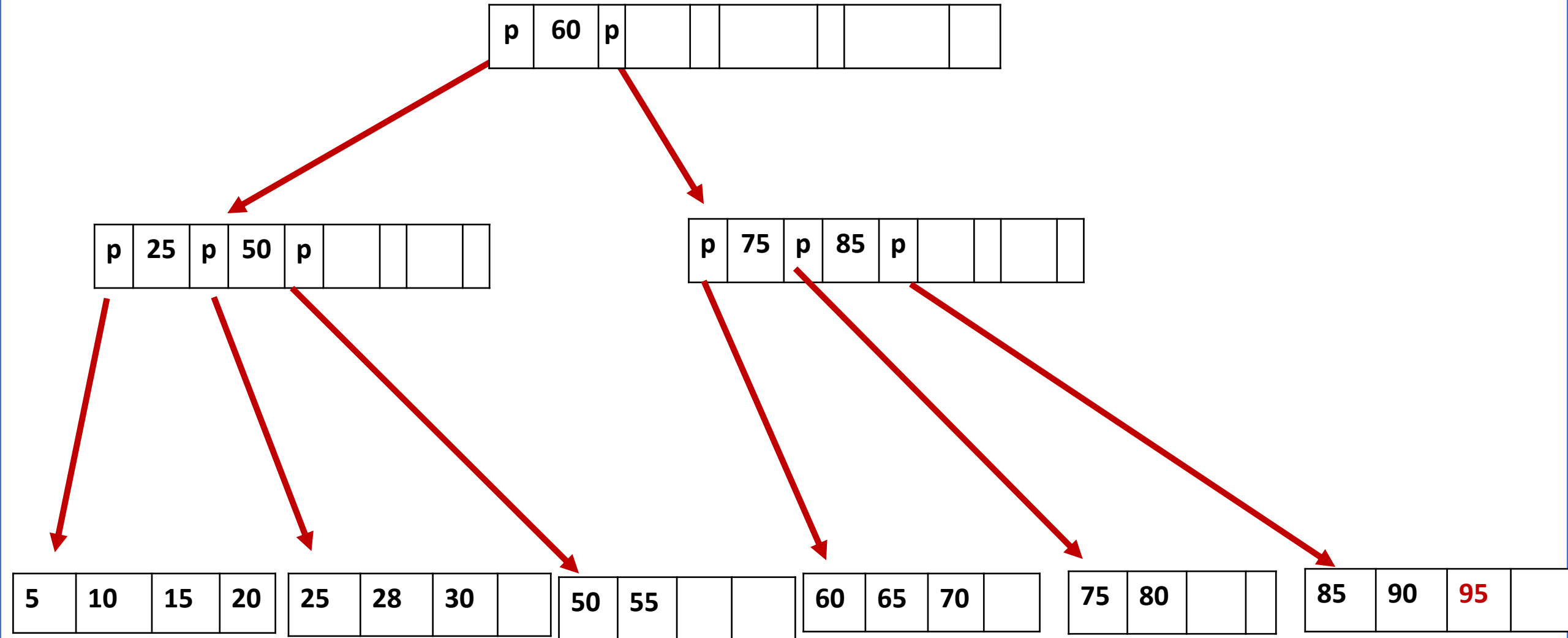
B+ Ağaç Yapısı: Ekleme (insert) işlemi

Yeni ağaca 95 değerini ekleyelim



ortadaki değer
ile yeni root (kök) dizin
oluşturulacak

B+ Ağaç Yapısı: Ekleme (insert) işlemi



B+ Ağaç Yapısı: Silme (delete) işlemi

1. Silinecek kayıttın yaprak düğümü anahtar değeri yardımıyla bulunur.
2. Silme işleminin gerçekleştirileceği düğümde ağacın derecesine eşit veya büyük sayıda kayıt varsa silme işlemi gerçekleştirilir.
3. $d \leq \text{düğümdeki kayıt sayısı} \leq 2d$ şartını sağlamıyorsa düğümler aynı parent(ebeveyn) a sahip düğümler birleştirilir (tekrar dağıtma işlemi)
4. Birleştirme işlemi gerçekleştirilirken bir üst seviyede (ebeveyn) sağdaki düğümü gösteren pointer ve anahtarı silinir
5. Bu işlem root (kök) a kadar devam eder.

Dinlediğiniz İçin
Teşekkürler....