BMB2006
VERİ YAPILARI

Doç. Dr. Murtaza CİCİOĞLU

Bursa Uludağ Üniversitesi

Bilgisayar Mühendisliği Bölümü

Bu ders notları Olcay Taner Yıldız ile Erkan TANYILDIZI hocaların notları kullanılarak hazırlanmıştır.

Hafta 5: Bağlı Liste - 2

Amaç:

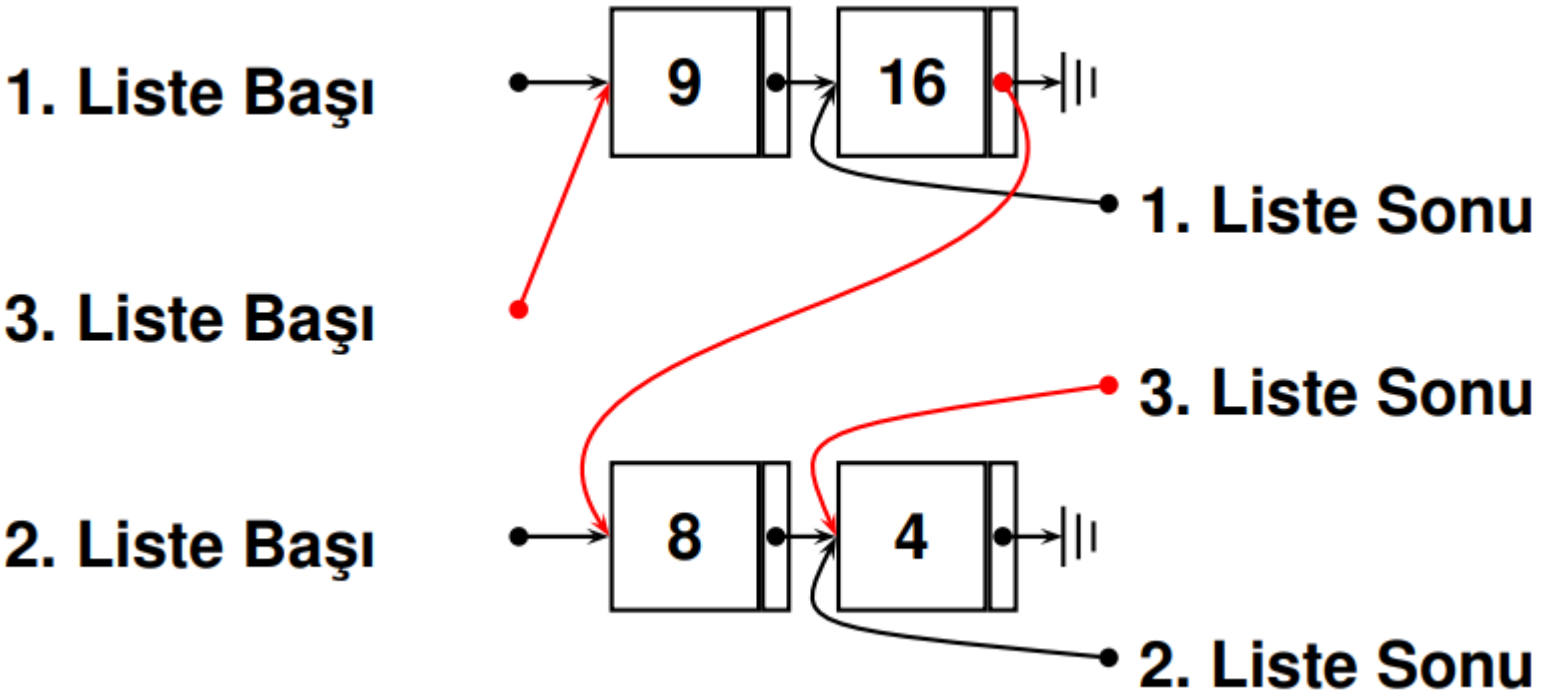
- Bağlı liste çalışma yapısı
- Bağlı liste çeşitleri

Yol haritası:

- Giriş
- Bağlı Liste Tanımı
- Temel Bağlı Liste İşlemleri
- İki Listeyi Birleştirme
- Çift Bağlı Liste
- Temel Çift Bağlı Liste
- Dairesel Bağlı Liste

İki Listeyi Birleştirme

- Yeni listenin elemanları eski iki listeden oluşur.

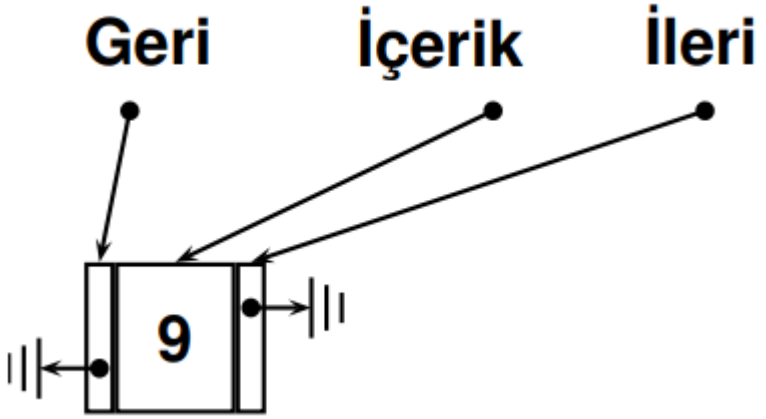



```
Liste  birlestir (Liste l1, Liste l2){
    Liste tmp;
    if (l1.bas == null)
        return l2;
    if (l2.bas == null)
        return l1;
    tmp = new Liste();
    tmp.bas = l1.bas;
    tmp.son = l2.son;
    l1.son.ileri  = l2.bas;
    return tmp;
}
```

Çift Bağlı Liste

- Listedeki elemanlar arasında iki yönlü bağ vardır. Elemanın bağlantı bilgisi bölümünde iki gösterici bulunur. Bu göstericinin biri kendisinden sonra gelen elemanı diğeri ise kendisinden önce gelen elamanın adres bilgisini tutar.
- Bu sayede listenin hem başından sonuna hem de listenin sonundan başına doğru hareket edilebilir. Bu yöntem daha esnek bir yapıya sahip olduğundan bazı problemlerin çözümünde daha işlevsel olabilmektedir.

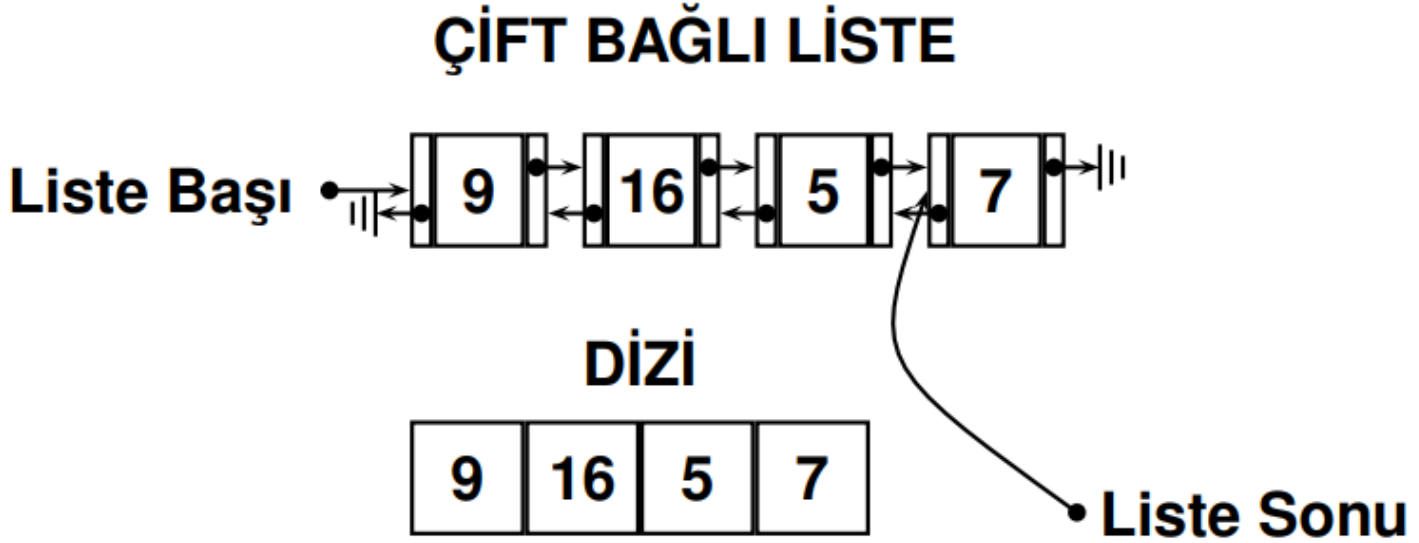
```
struct node{
        int icerik;
        cifteleman* ileri;
        cifteleman* geri; };
typedef struct node Node;
typedef Node* Nodeptr;
```



Çift Bağlı Liste

```
struct ciftliste
{
        Nodeptr bas;
         Nodeptr son;
};
typedef struct ciftliste Ciftliste;
typedef Ciftliste* Ciftlisteptr;
```

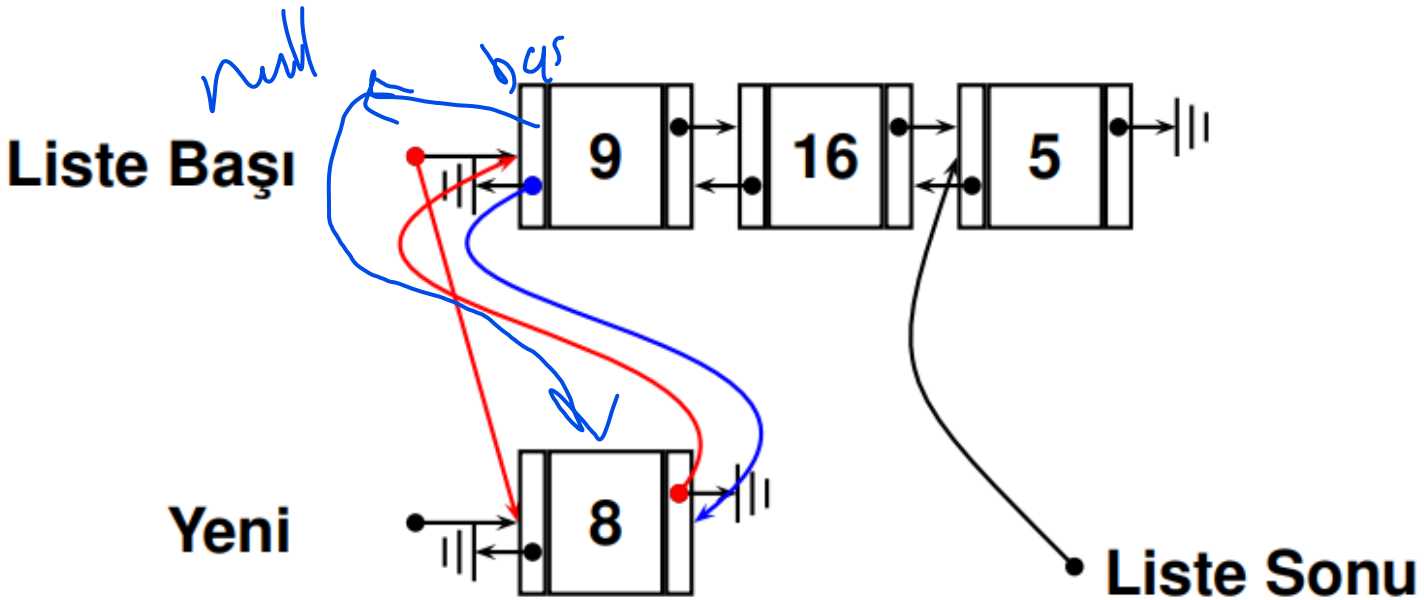
- Bir çift bağlı liste ve dizi yapısı

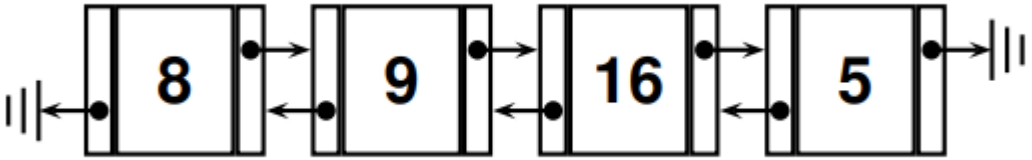


Temel Çift Baglı Liste İşlemleri

- Bir çift bağlı listenin başına eleman ekleme

```
void addHead(Ciftlisteptr liste,
Nodeptr n){
        if(liste->son == NULL)
                liste->son=n;
        else
                liste->bas->geri=n;
        n->ileri=liste->bas;
        liste->bas=n;
}
```

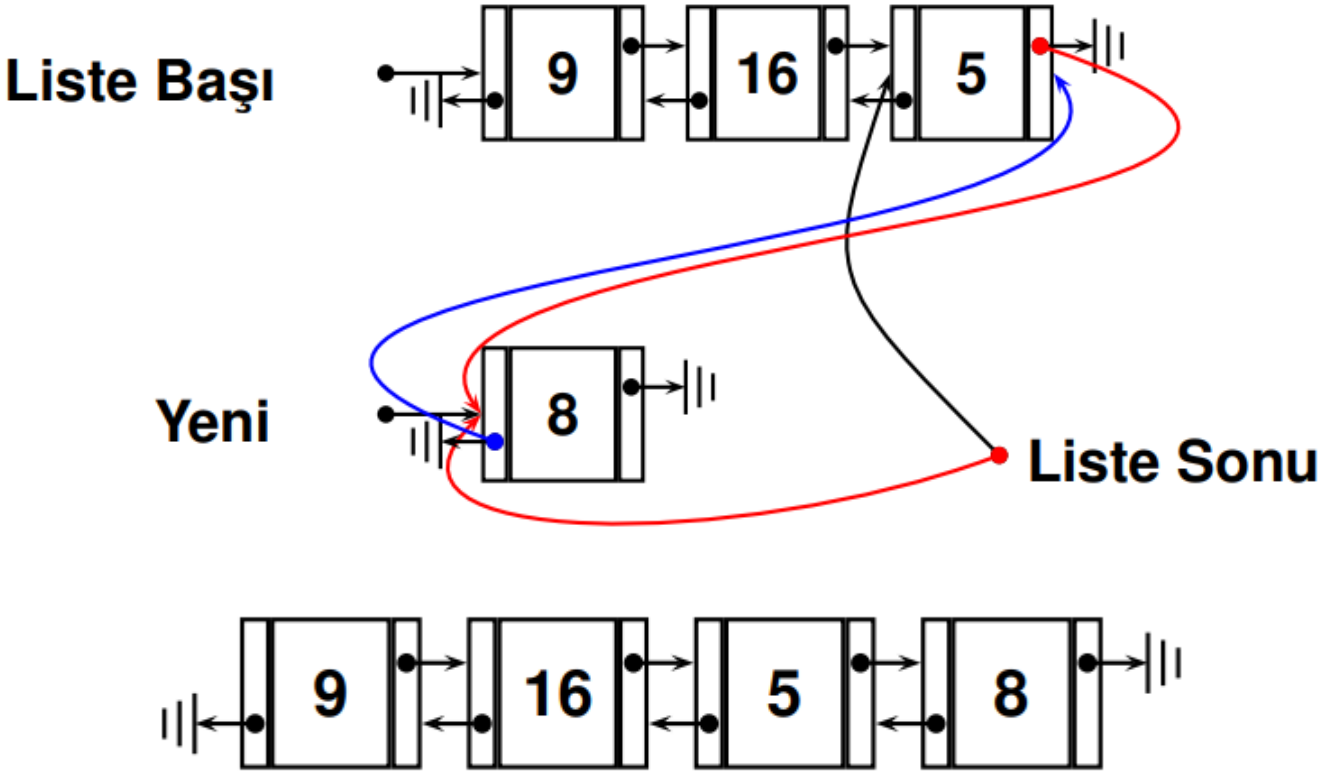




Temel Çift Baglı Liste İşlemleri

- Bir çift bağlı listenin sonuna eleman ekleme

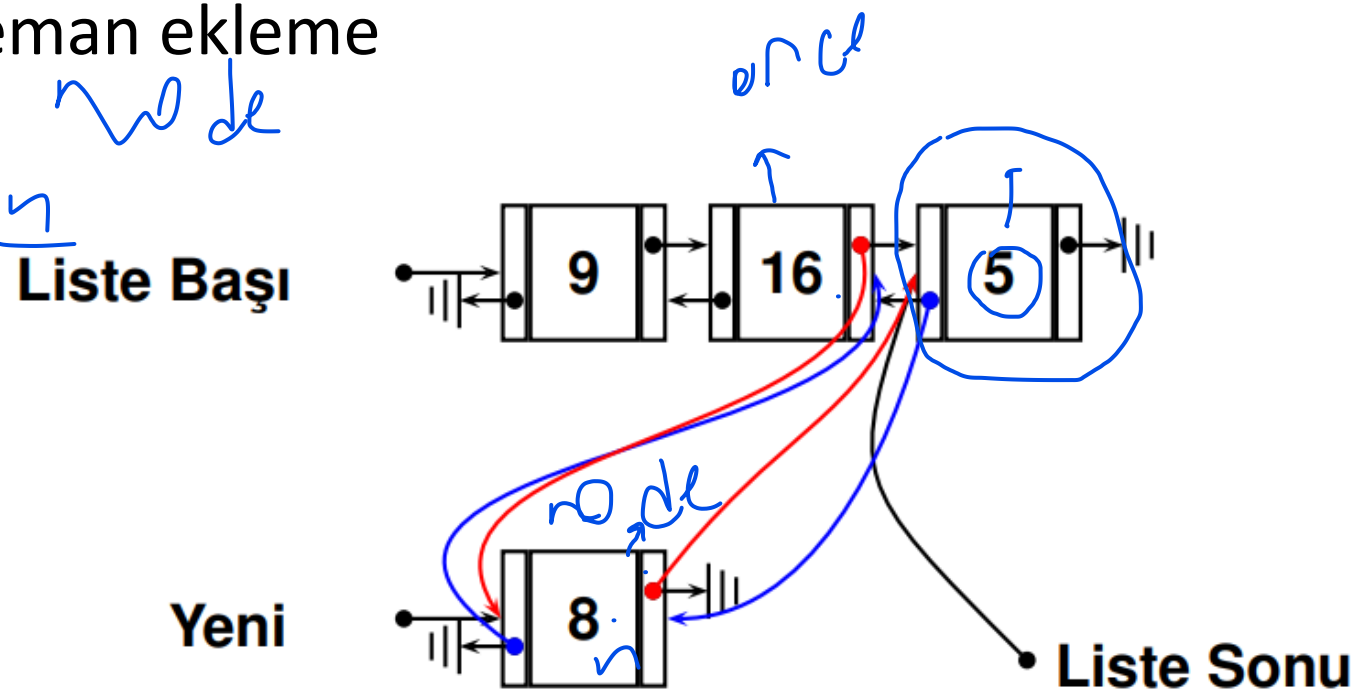
```
void addTail(Ciftlisteptr liste,
Nodeptr n){
        if(liste->bas==NULL)
                liste->bas=n;
        else
                liste->son->ileri=n;
        n->geri=liste->son;
        liste->son=n;
}
```

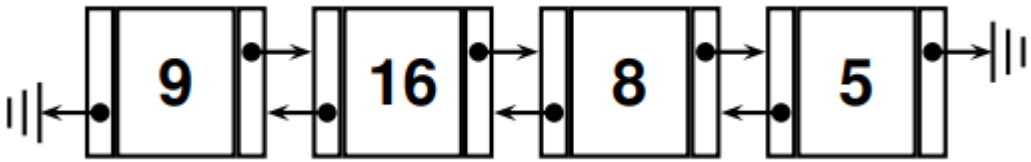


Temel Çift Baglı Liste İşlemleri

- Bir çift bağlı listenin ortasına eleman ekleme

```
void addMiddle(Ciftelemanptr
once, Ciftelemanptr n){
        n>ileri=once->ileri;
        n>geri=once;
        once->ileri->geri=n;
        once->ileri=n;
}
```



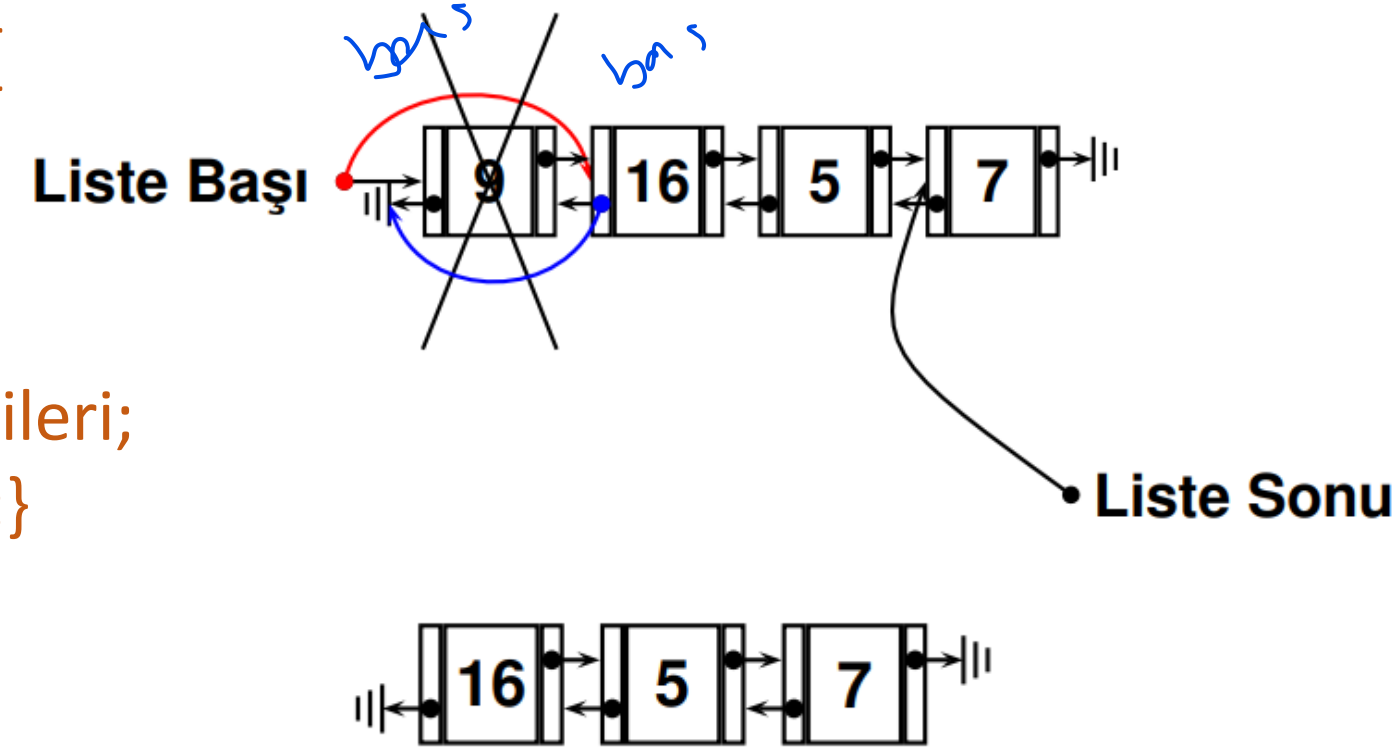


Listenin başına ekleme: O(1)
Listenin sonuna ekleme: O(1)
Listenin ortasına ekleme: O(1)

Temel Çift Baglı Liste İşlemleri

- Bir çift bağlı listenin ilk elemanını silme

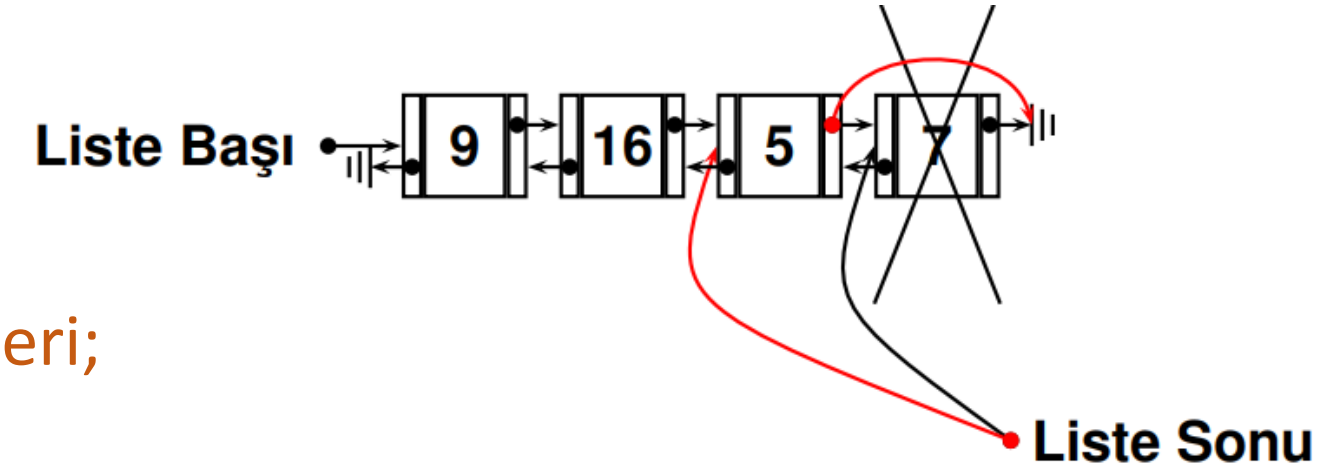
```
void removeHead(Ciftlisteptr liste){
        if(liste->bas==NULL)
                liste->son=NULL;
        else{
                liste->bas=liste->bas->ileri;
                liste->bas->geri=NULL;}
}
```

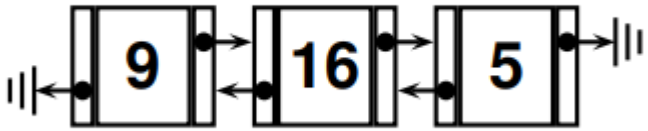


Temel Çift Baglı Liste İşlemleri

- Bir çift bağlı listenin son elemanını silme

```
void removeTail(Ciftlisteptr liste){
        if(liste->son==NULL)
                liste->bas=NULL;
        else{
                liste->son=liste->son->geri;
                liste->son->ileri=NULL;}
}
```

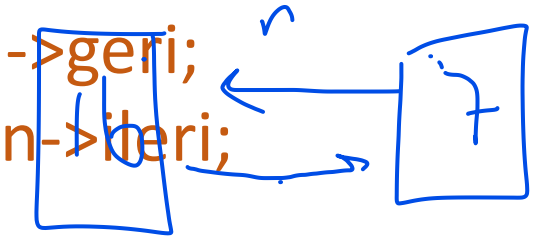




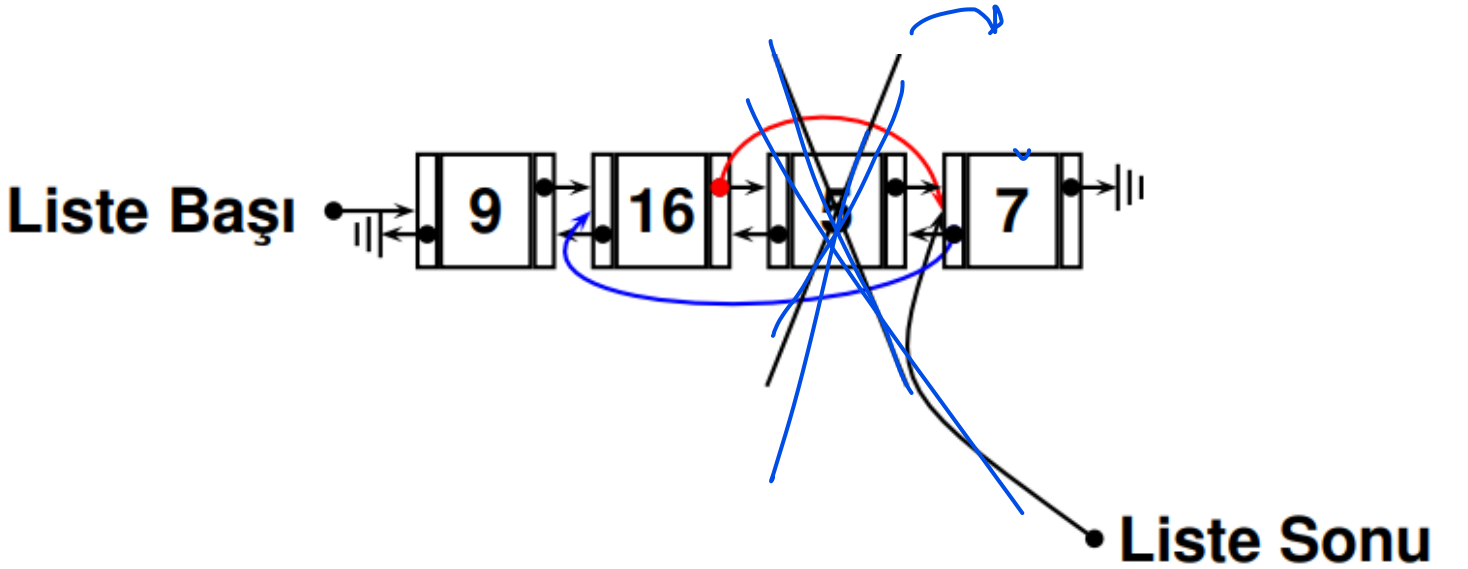
Temel Çift Baglı Liste İşlemleri

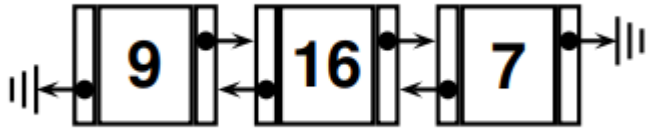
- Bir çift bağlı listenin ortasından silme

```
void removeMiddle(Ciftelemanptr n){
        n->ileri->geri=n->geri;
        n->geri->ileri = n->ileri;
}
```



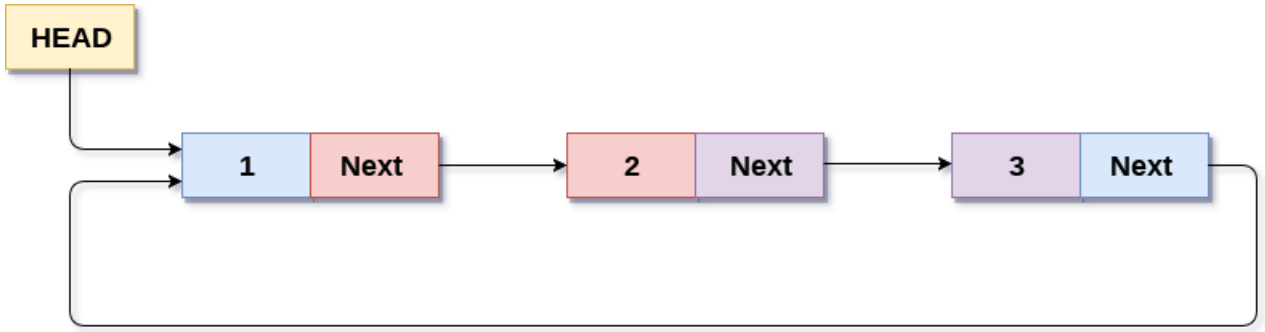
Listenin ilk elemanını silme: O(1)
Listenin ilk elemanını silme: O(1)
Listenin ortasından silme : O(1)

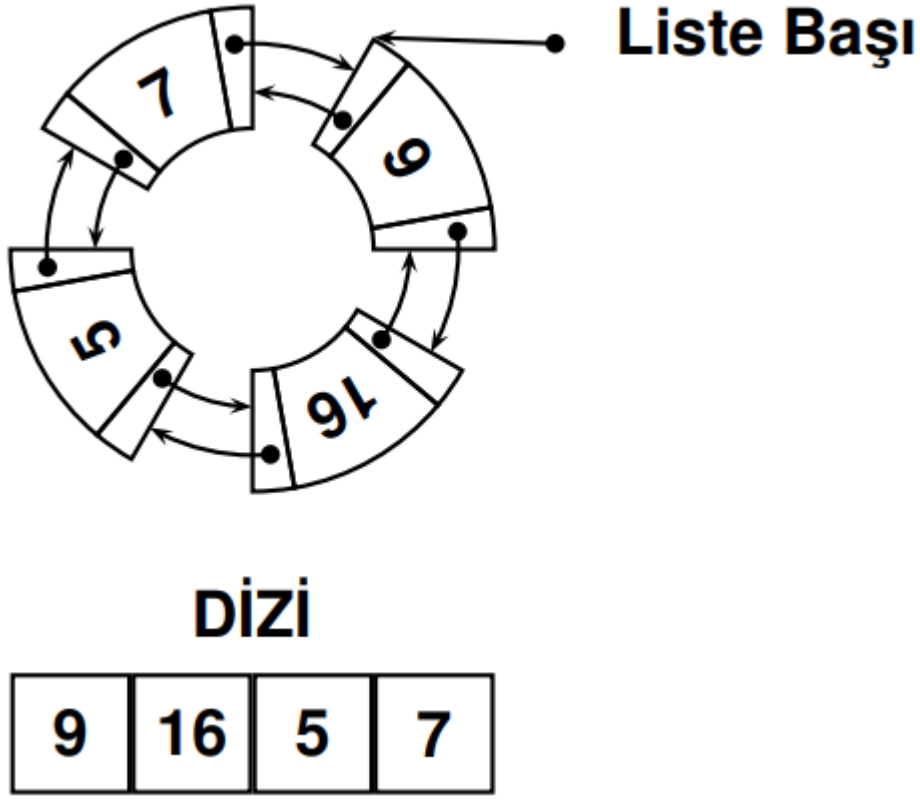




Dairesel Bağlı Liste

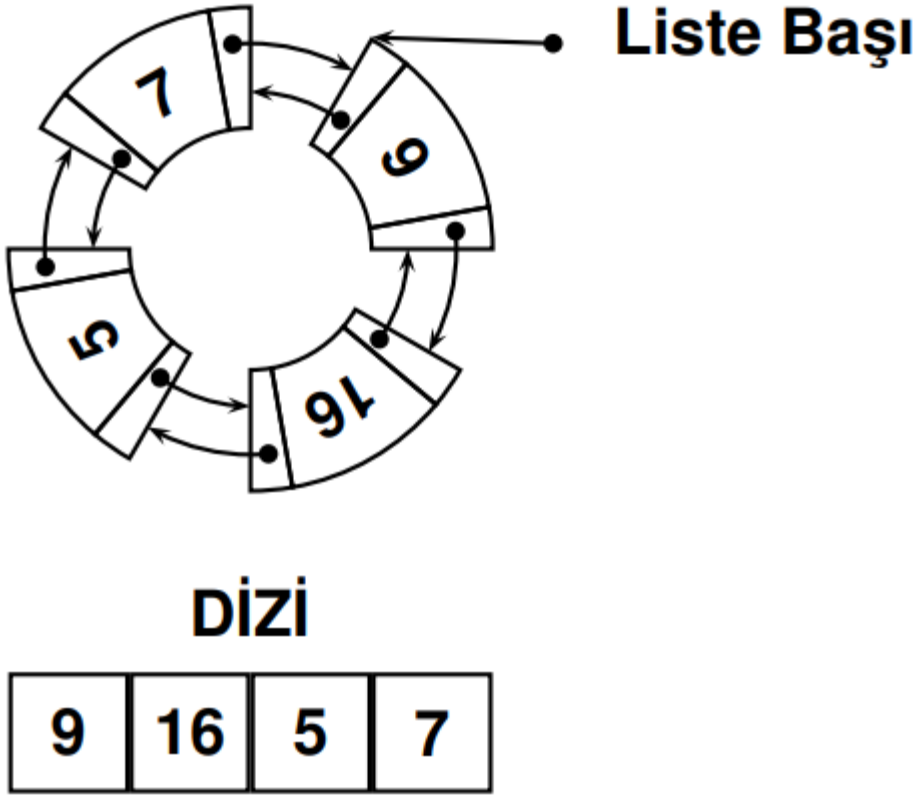
- Bir dairesel bağlı liste ve dizi yapısı





Dairesel Bağlı Liste

- Bir dairesel bağlı liste ve dizi yapısı

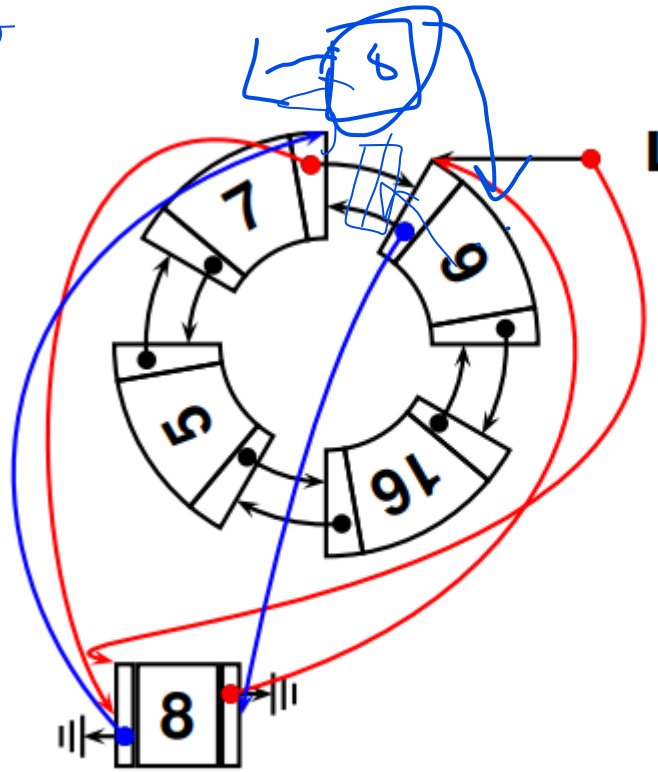


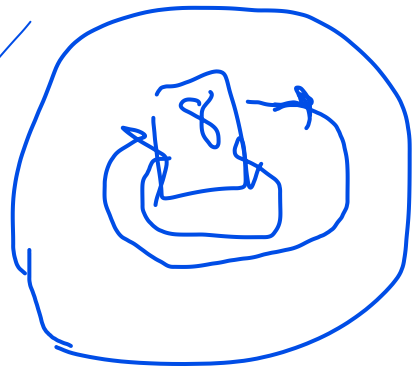

```
struct daireliste{
        Nodeptr bas; };
typedef struct daireliste Daireliste;
typedef Daireliste* Dairelisteptr;

Dairelisteptr new(){
        Dairelisteptr dliste;
        dliste = (Dairelisteptr)malloc(sizeof(Daireliste));
        dliste->bas=NULL;
        return dliste;
}
```

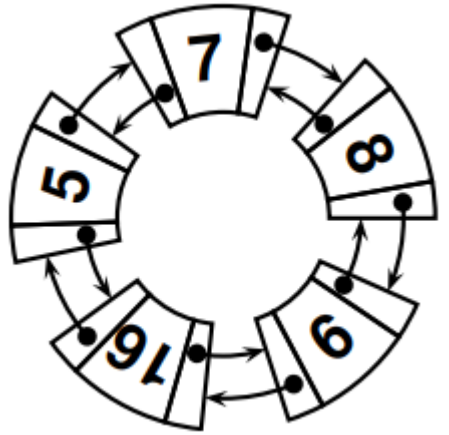
Dairesel Bağlı Liste

- Bir dairesel bağlı listenin başına 8 eklenmesi

```
void add(Dairelisteptr liste, Ciftelemanptr yeni){
        if(liste->bas==NULL){
                yeni->ileri=yeni;
                yeni->geri=yeni;}
        else{
                yeni->ileri=liste->bas;
                yeni->geri=liste->bas->geri;
                liste->bas->geri->ileri=yeni;
                liste->bas->geri=yeni;}
        liste->bas=yeni;
}
```

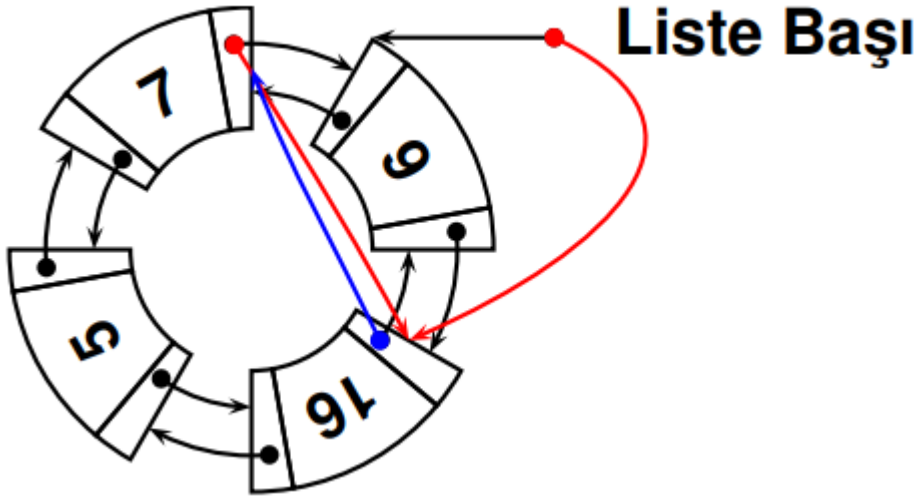




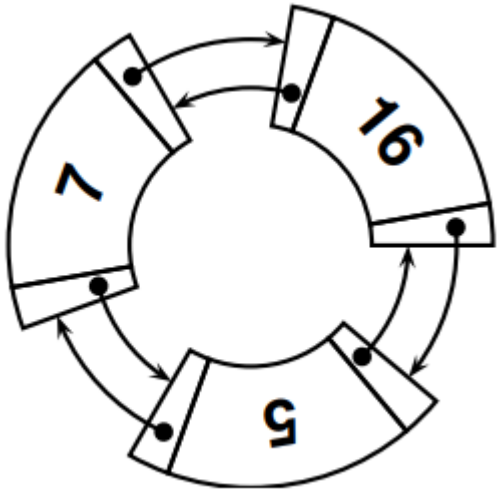


Dairesel Bağlı Liste

- Bir dairesel bağlı listenin ilk elemanını silme

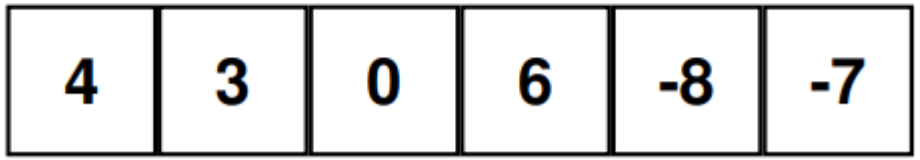



```
void remove(Dairelisteptr liste){
        if(liste->bas->ileri==liste->bas)
                liste->bas=NULL;
        else{
                liste->bas->geri->ileri=liste->bas->ileri;
                liste->bas->ileri->geri=liste->bas->geri;
                liste->bas=liste->bas->ileri;
        }
}
```

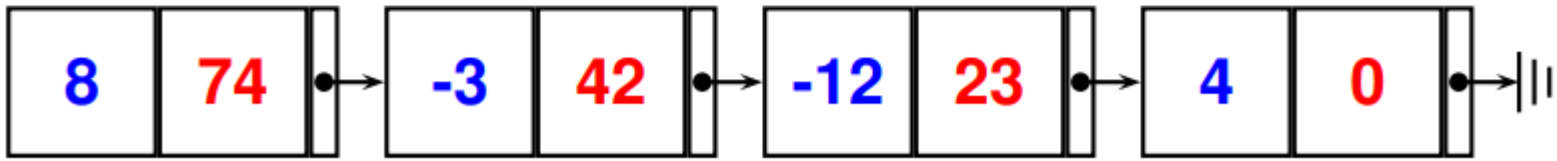


Uygulama: Polinom Aritmetiği

$4x^5 + 3x^4 + 6x^2 - 8x - 7$ ***polinomunun sabit dizi ile gösterimi***

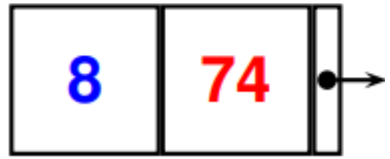


$8x^{74} - 3x^{42} - 12x^{23} + 4$ ***polinomunun bağlı liste ile gösterimi***



Uygulama: Polinom Aritmetiği

```
struct node{
        int katsayi;
        int us;
        struct node *ileri;};
typedef struct node Node;
typedef Node* Nodeptr;
Nodeptr yeniEleman(int katsayi, int us){
        Nodeptr node;
        node=(Nodeptr)malloc(sizeof(Node));
        node->katsayi=katsayi;
        node->us=us;
        node->ileri=NULL;
        return node;}
```



Uygulama: Polinom Aritmetiği

```
Listeptr topla(Listeptr l1, Listeptr l2){
        Nodeptr x, y, k, yeni;
        Listeptr sonuc=yeniListe();
        int katsayi , us;
        x=l1->bas;
        y=l2->bas;
        while(x!=NULL && y!=NULL){
                if(x->us ==y->us){
                        katsayi=x->katsayi+y->katsayi;
                        us=x->us;
                        x=x->ileri;
                        y=y->ileri;
                }
```

Uygulama: Polinom Aritmetiği

```
else{
                if(x->us > y->us){
                        katsayi=x->katsayi;
                        us=x->us;
                        x=x->ileri;
                }
                else{
                        katsayi=y->katsayi;
                        us=y->us;
                        y=y->ileri;
                }
        }
```

Uygulama: Polinom Aritmetiği

```
if(katsayi!=0){
                yeni=yeniEleman(katsayi, us);
                listeSonunaEkle(sonuc, yeni);
}}
if(x==NULL)          k=y;
else                 k=x;
while(k!=NULL){
        yeni=yeniEleman(k->katsayi, k->us);
        listeSonunaEkle(sonuc, yeni);
        k=k->ileri;
}
return sonuc;}
```

Bağlı Listenin Dizi Üzerinde Uygulanması

```
Define BOYUT 100
struct node{
        int veri, sonraki;
};
struct node Node[BOYUT];
typedef Node* Nodeptr;
```

	işaretçi	veri	sonraki
	0	Tülay	-1
	1	Vedat	6
LB3=2	2	Pelin	15
	3	Burak	9
	4	Kadir	-1
	5	Seyhan	13
	6	Utku	16
LB1=7	7	Ayşe	3
	8		
	9	Ceyda	11
	10		
	11	Dilek	-1
	12		
	13	Şükrü	4
LB2=14	14	Zeynep	17
	15	Ramazan	5
	16	Ümmihan	0
	17	Yasin	1
	18		
	19		

Bağlı Listenin Dizi Üzerinde Uygulanması

LB1 listesinin başına Hasan isimli veri ekleyelim.

	işaretçi	veri	sonraki
	0	Tülay	-1
	1	Vedat	6
LB3=2	2	Pelin	15
	3	Burak	9
	4	Kadir	-1
	5	Seyhan	13
	6	Utku	16
LB1=7	7	Ayşe	3
	8		
	9	Ceyda	11
	10		
	11	Dilek	-1
	12		
	13	Şükrü	4
LB2=14	14	Zeynep	17
	15	Ramazan	5
	16	Ümmihan	0
	17	Yasin	1
	18		
	19		

Bağlı Listenin Dizi Üzerinde Uygulanması

Boş düğüm olarak 8'i seçtik.

	işaretçi	veri	sonraki
	0	Tülay	-1
	1	Vedat	6
LB3=2	2	Pelin	15
	3	Burak	9
	4	Kadir	-1
	5	Seyhan	13
	6	Utku	16
	7	Ayşe	3
LB1=8	8	Hasan	7
	9	Ceyda	11
	10		
	11	Dilek	-1
	12		
	13	Şükrü	4
LB2=14	14	Zeynep	17
	15	Ramazan	5
	16	Ümmihan	0
	17	Yasin	1
	18		
	19		

Bağlı Listenin Dizi Üzerinde Uygulanması

LB2'den Utku verisini silelim.

	işaretçi	veri	sonraki
	0	Tülay	-1
	1	Vedat	6
LB3=2	2	Pelin	15
	3	Burak	9
	4	Kadir	-1
	5	Seyhan	13
	6	Utku	16
	7	Ayşe	3
LB1=8	8	Hasan	7
	9	Ceyda	11
	10		
	11	Dilek	-1
	12		
	13	Şükrü	4
LB2=14	14	Zeynep	17
	15	Ramazan	5
	16	Ümmihan	0
	17	Yasin	1
	18		
	19		

Bağlı Listenin Dizi Üzerinde Uygulanması

LB2'den Utku verisini silelim.

	işaretçi	veri	sonraki
	0	Tülay	-1
	1	Vedat	16
LB3=2	2	Pelin	15
	3	Burak	9
	4	Kadir	-1
	5	Seyhan	13
	6	Utku	16
	7	Ayşe	3
LB1=8	8	Hasan	7
	9	Ceyda	11
	10		
	11	Dilek	-1
	12		
	13	Şükrü	4
LB2=14	14	Zeynep	17
	15	Ramazan	5
	16	Ümmihan	0
	17	Yasin	1
	18		
	19		

Bağlı Listenin Dizi Üzerinde Uygulanması

- 10 yataklı bir hastane binasında boş ve dolu yatakların izlenmesi için bağlı listeden yararlanılacaktır.
- Hastaneye yeni yatış yapanlar ve taburcu olanlar için gerekli güncellemeler bu bağlı liste üzerinden yapılacaktır. Bunun için yatış yapanların izlenmesi (dolu yataklar) için bir bağlı liste, boş yatakların izlenmesi için ayrı bir bağlı liste aynı dizi üzerinde düzenlenebilir.
- Mevcut durumda yataklardan 8 tanesi dolu 2 tanesi boş olsun. Bu örnekte işaretçiler yatak numaralarına işaret edeceklerdir.
- Dolu yataklar listesinin liste başı için bir işaretçi, boş yataklar liste başı için ayrı bir işaretçi kullanılacaktır.

Bağlı Listenin Dizi Üzerinde Uygulanması

- Gül taburcu olursa dizi nasıl değişir?

	Yatak no	Hasta Adı	İşaretçi
	1	Dilek	8
	2	Leyla	9
	3	----------	-1
	4	Vildan	-1
	5	Gül	10
LB1=6	6	Banu	1
LB2=7	7	----------	3
	8	Elif	5
	9	Selin	4
	10	Jale	2

	Yatak no	Hasta Adı	İşaretçi
	1	Dilek	8
	2	Leyla	9
	3	----------	-1
	4	Vildan	-1
LB2=5	5	----------	7
LB1=6	6	Banu	1
	7	----------	3
	8	Elif	10
	9	Selin	4
	10	Jale	2

Bağlı Listenin Dizi Üzerinde Uygulanması

- Reyhan isimli bir hasta hastaneye yatış yapacak. Listemiz alfabetik sırada olmalıdır!

	Yatak no	Hasta Adı	İşaretçi
	1	Dilek	8
	2	Leyla	9
	3	----------	-1
	4	Vildan	-1
LB2=5	5	----------	7
LB1=6	6	Banu	1
	7	----------	3
	8	Elif	10
	9	Selin	4
	10	Jale	2

	Yatak no	Hasta Adı	İşaretçi
	1	Dilek	8
	2	Leyla	5
	3	----------	-1
	4	Vildan	-1
	5	Reyhan	9
LB1=6	6	Banu	1
LB2=7	7	----------	3
	8	Elif	10
	9	Selin	4
	10	Jale	2

ÖDEV

- Kullanıcıdan alınan değerlere göre dizi üzerinde tutulacak iki yönlü bir bağlı liste için gerekli algoritmaları oluşturunuz.
- Rastgele sayısal değerlere sahip iki yönlü bir bağlı liste içinde kullanıcı tarafından alınan değeri ikili arama (Binary Search) algoritmasına göre bulan algoritmayı yazınız.
- Josephus probleminin tek yönlü dairesel bağlı liste ile C, C# dilinde gerçekleştiriniz.