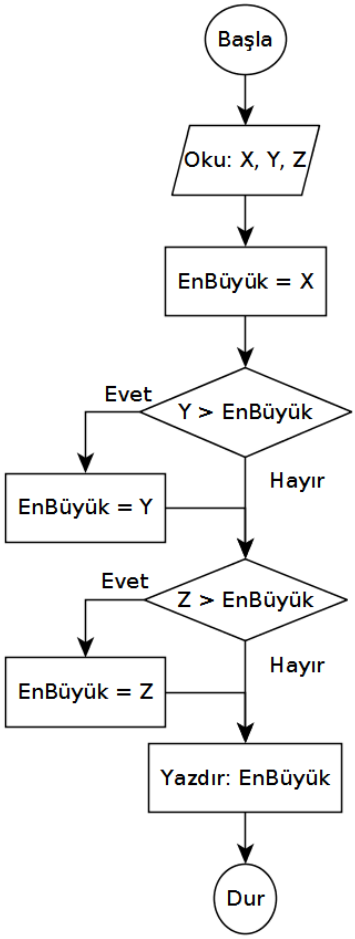
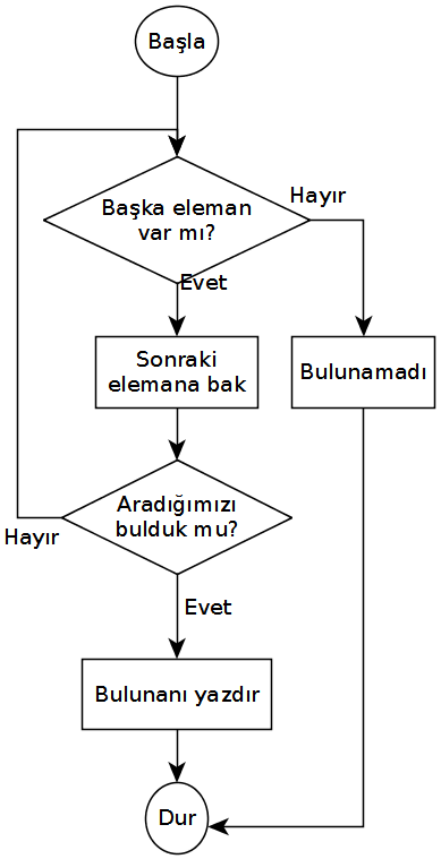
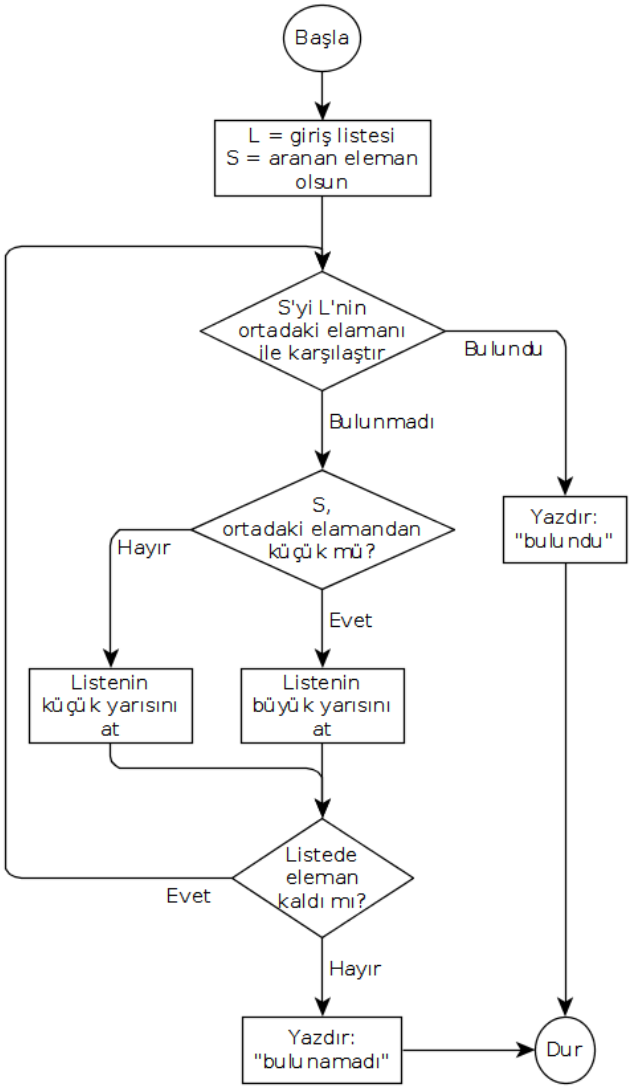
Tanım soruları:

- Algoritma
- Sözde kod
- Özyineleme algoritma
- Böl ve fethet algoritma
- Açgözlü (greedy) algoritma
- Sonlu durum makinesi
- Veri yapısı
- Primitif (basit) ve bileşik veri yapılarını kısaca açıklayınız. İki türe de birer örnek veriniz.

---

3 elemanlı kabarcık Sıralama	Sıralı (doğrusal, naif) arama	İkiye Bölme Arama
		

Örnek akış diyagramı soruları:

1. Kabarcık sıralama akış diyagramına bakarak cevaplayın. {X,Y,Z} sayıları için sisteme sırasıyla {-3,5,0} verilmiş olsun. “EnBüyük=Z” işlemi tamamlandığında, EnBüyük değişkeninin değeri ne olur?
2. Kabarcık sıralama akış diyagramına bakarak cevaplayın. {X,Y,Z} sayıları için sisteme sırasıyla {14,-21,44} verilmiş olsun. Akış diyagramı tamamlandığında ekranda ne yazacaktır?

3. Sıralı arama akış diyagramına göre cevaplayın. Aranan sayı “2” olsun. Sayıyı aradığımız dizi {5,-1,2,0,8} şeklinde olsun. Bu durumda; “Aradığımızı bulduk mu?” etiketli kutunun “Hayır” yönündeki bağlantı kaç kere işleyecektir?
4. Sıralı arama akış diyagramına göre cevaplayın. Sayıyı aradığımız dizi, 0 elemanlı olsun. Akış diyagramı üzerindeki sembollerden (aşamalar) kaç tanesi işletilecektir?
5. Vb.

Örnek algoritma işletme sorusu: Aşağıda; kabarcık sıralama için sözde kod bloğu verilmiştir. Bu kodlarda; `değiştir` fonksiyonu kendisine verilen iki değişkenin değerlerini yer değiştirmektedir. Bu kod bloğu, kendisine verilen `liste` isimli bir dizinin tüm elemanlarını sıralamaktadır. Buna göre programın `liste={0,3,1,6}` şeklinde başlatıldığını varsayarak her bir aşamada `liste` değişkeninin durumunu yazarak programı kağıt üzerinde işletiniz.

```
for: liste’nin tüm elemanları için dön. Döngü sayacı “i” olsun.
   eğer: liste[i] > liste[i+1]
      değiştir (liste[i], liste[i+1])
   eğer_sonu
for_sonu
```

Örnek çözüm:

- for döngüsü 3 kere (eleman sayısının 1 eksiği kadar) dönecektir. Karşılaştırılacak olan elemanlar “1 ile 2”, “2 ile 3”, “3 ile 4” şeklinde olacaktır.
- Eğer fonksiyonunun ilk işletilmesinde liste[1] ve liste[2] şeklinde ilk iki elemanı karşılaştırıyor. liste[1] > liste[2] (0>3) koşulu sağlanmadığından bir değiştirme işlemi yapılmıyor.
- Eğer fonksiyonunun ikinci işletilmesinde 2. ve 3. elemanlar karşılaştırılıyor. 3>1 olduğundan ikisi yer değiştiriyor ve liste={0,1,3,6} şeklini alıyor.
- Eğer fonksiyonunun üçüncü işletilmesinde güncel listenin 3. ve 4. elemanları karşılaştırılıyor. 3>6 koşulu sağlanmadığından yer değiştirme yapılmıyor.