

# KODLAMAYA GİRİŞ VE ALGORİTMA

---

❖ Yazılım Nedir

❖ Algoritma

❖ Akış Seması

❖ Örnekler

# Yazılım Nedir?

Elektronik cihazlarda çeşitli görevleri gerçekleştirmek amacıyla hazırlanmış programlara yazılım adı verilir.



# Yazılımlar...

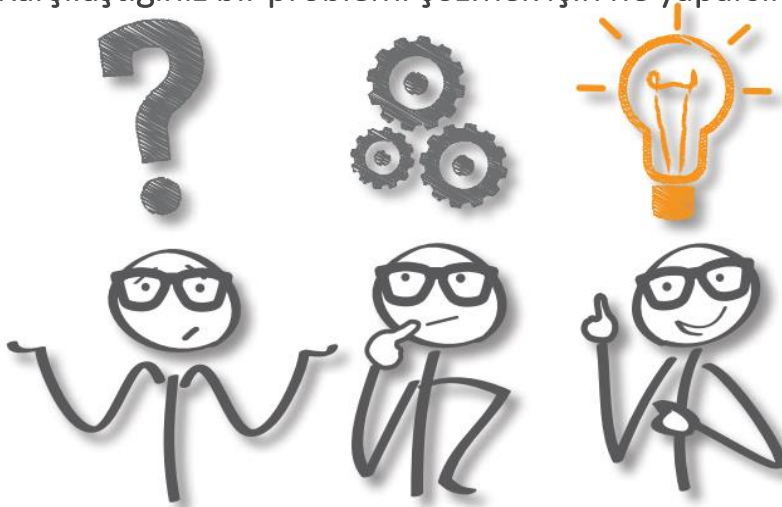
Her yazılım bir problemi çözmek amacıyla geliştirilmiştir.



# Problem Nedir?

Problem, çözümlmesi gereken sorun ya da aşılması gereken engel anlamına gelir.  
Günlük hayatta sık sık problemlerle karşılaşırız.

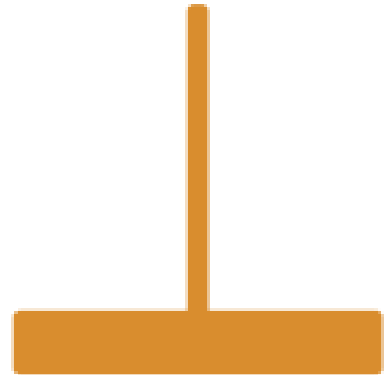
Karşılaştığınız bir problemi çözmek için ne yaparsınız?



# Bir Problemin Çözümü İçin...



# Hanoi Kuleleri



# Problem Çözme

Günlük yaşamda karşılaştığımız problemleri bilerek veya farkında olmadan adım adım çözmeye çalışırız.

Örneğin yazı yazarken kaleminizin ucu kırıldığında şu adımları takip ederek bu sorunu çözersiniz.



1. Kalem tıraşı çıkar.
2. Kalemi al.
3. Çöp kovasının yanına git.
4. Kalemin ucunu aç.
5. Sırana geri dön.
6. Yazmaya devam et.

# Peki Ya Bilgisayarlar?

Bilgisayarlar da problemleri tıpkı bizler gibi çözmeye çalışır. Kullanıcı tarafından kendisine verilen komutları **adım adım** uygulayarak problemin çözümüne ulaşır.

Kullandığımız yazılımların tamamı «**kod**» adı verilen bilgisayarın anlayacağı dilde yazılmış özel komutlardan oluşur. Bu kodlar bilgisayar yazılımcıları tarafından yazılır.

```
(* Programming Languages *)  
  
fun append (xs,ys) =  
  if null xs  
  then ys  
  else hd xs :: append (tl xs, ys)  
  
fun map (f,xs) =  
  case xs of  
    [] => []  
  | x::xs' => f x :: map (f,xs')  
  
val a = map ((fn x => x+1) o hd,  
             [[9, 10, 15],  
              [12, 27, 13]])
```



# Kodlamadan Önce...

Kodlamaya başlamadan önce oluşturacağımız yazılımın adım adım ne yapacağını tasarlamamız gerekir.

İşte açık ve net ifadelerle problemin adım adım çözümünü gösteren bu taslağa «**algoritma**» adı verilir.

Programlamanın ilk adımı algoritma oluşturmaktır.



# Algoritma

Bir problemin çözümünde izlenecek yol anlamına gelir ve problemin çözümünün adımlar halinde yazılmasıyla oluşturulur.

Algoritma basamaklarının bir başlangıcı ve sonu bulunur. Her adımda yapılacak işlem açıkça belirtilir.

Adım 1: Başla

Adım 2: Hava yağmurlu mu?

Adım 3: Evet ise Adım 5'e git.

Adım 4: Hayır ise Adım 6'ya git.

Adım 5: Yanına şemsiye al.

Adım 6: Evden çık.

Adım 7: Bitir.

# Örnek Algoritma

Şimdi basit bir problemin çözümünü gösteren bir algoritma hazırlayalım.

Ayran yapıp bardağa dolduralım.



Adım 1: Başla

Adım 2: Yoğurdu kaba koy.

Adım 3: Su ekle.

Adım 4: Çırp.

Adım 5: Tuz koy.

Adım 6: Bardağa doldur.

Adım 7: Bitir.



# Örnek Algoritma - 2

Arabayı çalıştırıp yola çıkalım

Adım 1: Başla

Adım 2: Sürücü koltuğuna geç.

Adım 3: Emniyet kemerini tak.

Adım 4: Aynaları kontrol et.

Adım 5: Anahtarı tak.

Adım 6: Kontakı çevir.

Adım 7: El frenini indir.

Adım 8: Vitese geç.

Adım 9: Gaza bas.

Adım 10: Bitir.



# Neden Algoritma Kullanıyoruz?

Sizce kodlamaya başlamadan önce niçin algoritma hazırlıyoruz?

Gerekli tüm bilgi ve birikime sahipsiniz ve sizden bir bina yapmanız isteniyor.  
Yapacağınız ilk iş ne olurdu?



# Neden Algoritma Kullanıyoruz?

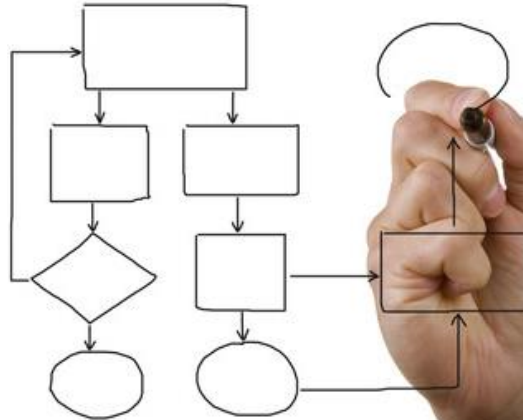
Oluşturacağımız yazılımın kusursuz olması için öncelikle her adımını gösteren planını, yani algoritmasını hazırlamalıyız.



# Akış Şeması

Bilgisayar programlarının işlem basamaklarını geometrik şekillerle gösteren şemadır. Bu şekiller vasıtasıyla algoritmanın daha rahat anlaşılabilmesi sağlanır.

Şemada yer alan her şeklin bir kullanım amacı vardır.



# Elips

**Başla** ve **Bitir** adımları için kullanılır. Akış şemasının başlangıç ve bitiş noktasında yer alır.



BAŞLA



BİTİR



# Paralel Kenar

Giriş işlemleri için kullanılır.

Örneğin; klavyeden bir sayı girilmesi istenmesi gibi.



# Dalgalı Dikdörtgen

Çıkış işlemleri için kullanılır.

Örneğin; ekrana işlem sonucunun yazdırılması gibi.

Girdiğiniz sayı çift

# Dikdörtgen

**Hesaplama** ya da **Değişkene Değer Atama** işlemleri için kullanılır.  
Örneğin; iki sayıyı topla veya girilen ilk sayıyı A olarak kabul et.

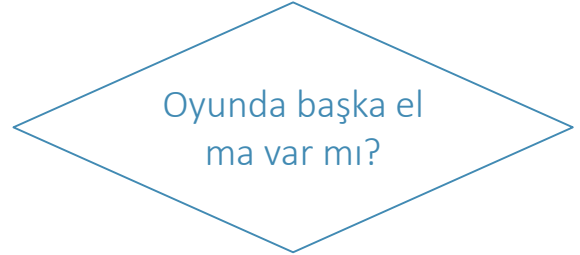
A ile B'yi topla

A = İlk Sayı

# Eşkenar Dörtgen

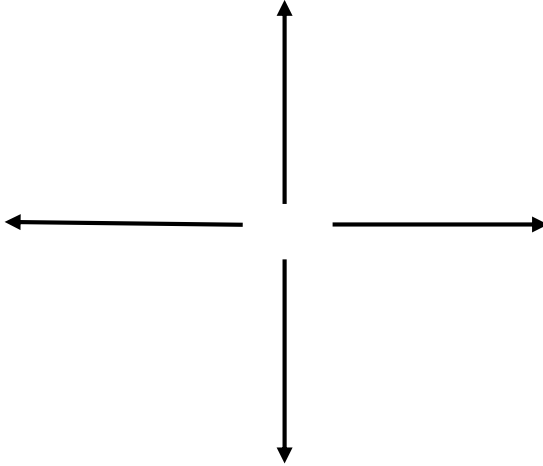
**Karşılaştırma** ya da **Karar Verme** işlemleri için kullanılır.

Örneğin; girilen sayı 5'ten büyük mü?



# Yön Okları

Akış şemasının ilerleme yönünü gösterir.



# Akış Şeması Örneği

Klavyeden girilen iki sayıyı toplayıp ekrana yazdıran programın akış şemasını çizeceğiz. Önce **algoritmasını** yazalım.

Adım 1: Başla

Adım 2: İlk sayıyı gir.

Adım 3:  $A =$  İlk sayı

Adım 4: İkinci sayıyı gir.

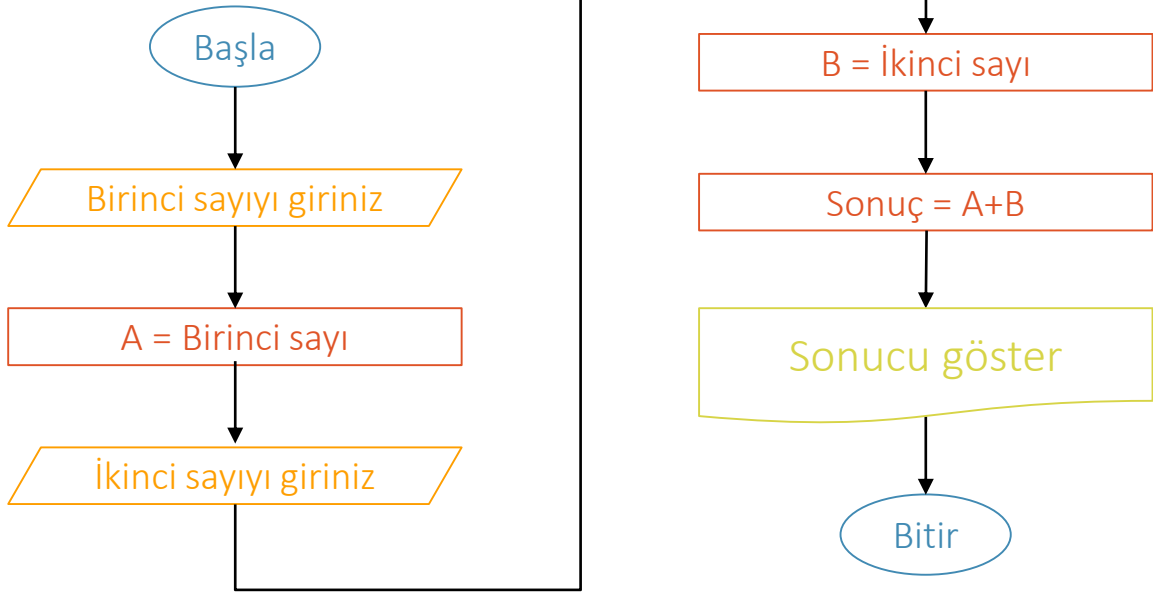
Adım 5:  $B =$  İkinci sayı

Adım 6: İki sayıyı topla ( $A+B$ )

Adım 7: Sonucu ekranda göster.

Adım 8: Bitir.

# Akış Şeması Örneği



# Akış Şeması Örneđi - 2

Şimdi hava yağmurlu ise bizi şemsiye almamız konusunda uyarın programın akış şemasını çizeceđiz. Önce **algoritmasını** yazalım.



Adım 1: Başla

Adım 2: Hava yağmurlu mu?

Adım 3: Evet ise Adım 5'e git.

Adım 4: Hayır ise Adım 6'ya git.

Adım 5: Yanına şemsiye al.

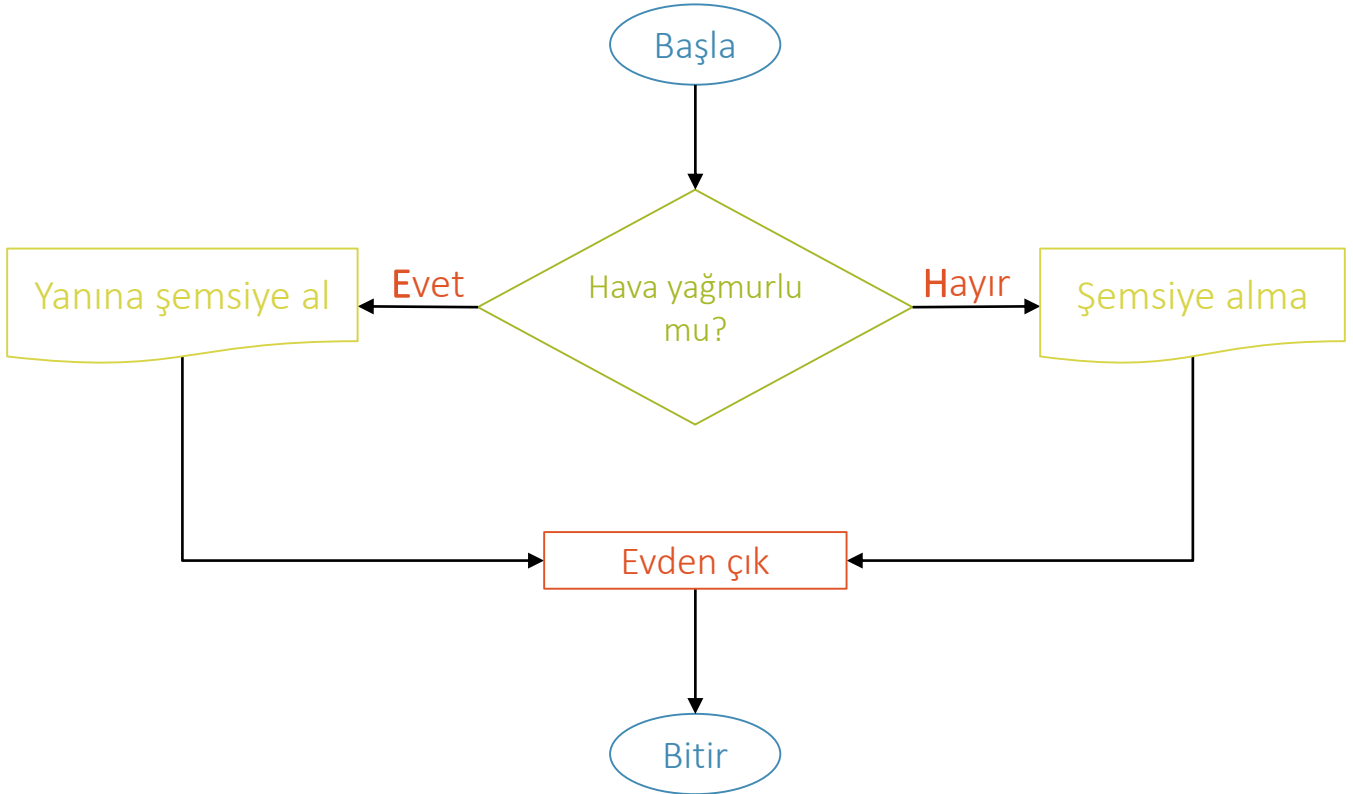
Adım 6: Evden çık.

Adım 7: Bitir.





# Akış Şeması Örneği - 2



# Uygulama

Bir öğrencinin klavyeden girilen iki notunun ortalamasını hesaplayan ve çıkan sonuca göre notun iyi veya kötü olduğunu ekrana yazdıran programın algoritmasını ve akış şemasını hazırlayınız.

(Ortalama 70'ten büyük ise **İYİ**, küçük ise **KÖTÜ** kabul edilecek.)



# Uygulama

