

# PROBLEM ÇÖZME, PROGRAMLAMA ve ÖZGÜN ÜRÜN GELİŞTİRME

## PROGRAMLAMAYA GİRİŞ VE ALGORİTMA

- ☀ Yazılım Nedir
- ☀ Algoritma
- ☀ Akış Seması
- ☀ Örnekler



# 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...

Problemi  
iyi  
anlamak

Kısa ve  
anlaşılır  
biçimde  
çözmek

Ve sonucun  
doğruluğunu  
kontrol etmek

# 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.



**PROBLEM**

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.

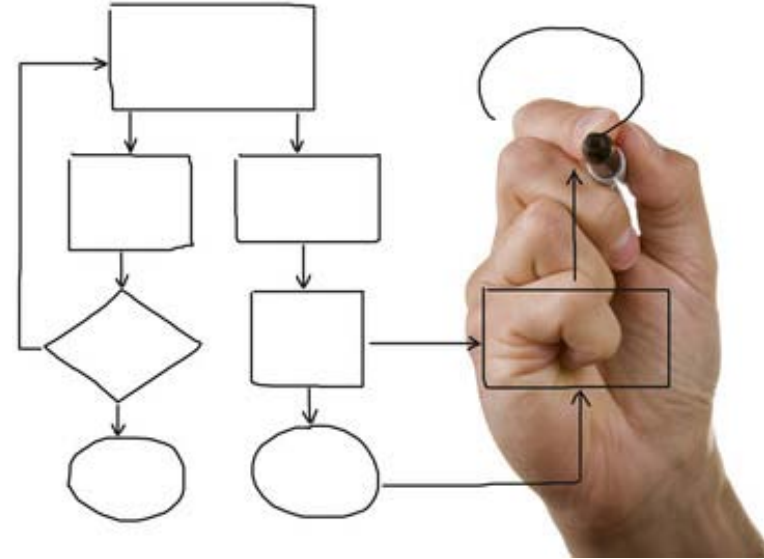
```
author Mike  
This class will use the ghost  
to insert a collection of particles which  
have no net charge. This is used to calcul  
chemical potential and activity coefficient  
class widom : public analysis {  
private:  
    average<double> expsum; //!< Average of  
protected:  
    int ghostin;  
    long long int cnt;  
    vector<particle> g; //!< count test  
public:  
    widom(int n=10); //!< List of gh  
    string info();  
    void add(particle);  
    void add(container &);  
    void insert(container &, energybase &  
    void check(checkValue &);  
    void check(checkValue &);  
    void ma() { return exp(muex());  
    } return -log(expsum.2
```

# 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.



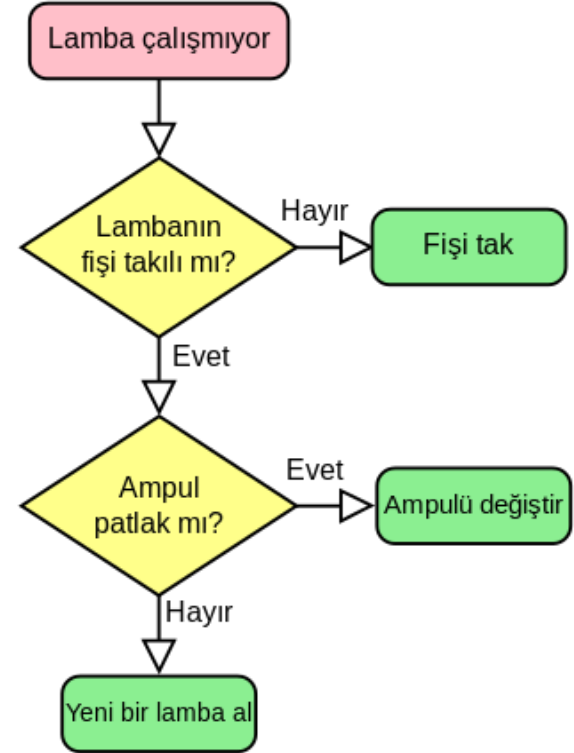


# Algoritma

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.



# 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 çıkarım.



Adım 1: Başla

Adım 2: Arabanın kapısını aç.

Adım 3: Sürücü koltuğuna geç.

Adım 4: Emniyet kemerini tak.

Adım 5: Aynaları kontrol et.

Adım 6: Anahtarı tak.

Adım 7: Kontakı çevir.

Adım 8: El frenini indir.

Adım 9: Vitese geç.

Adım 10: Gaza bas.

Adım 11: 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.

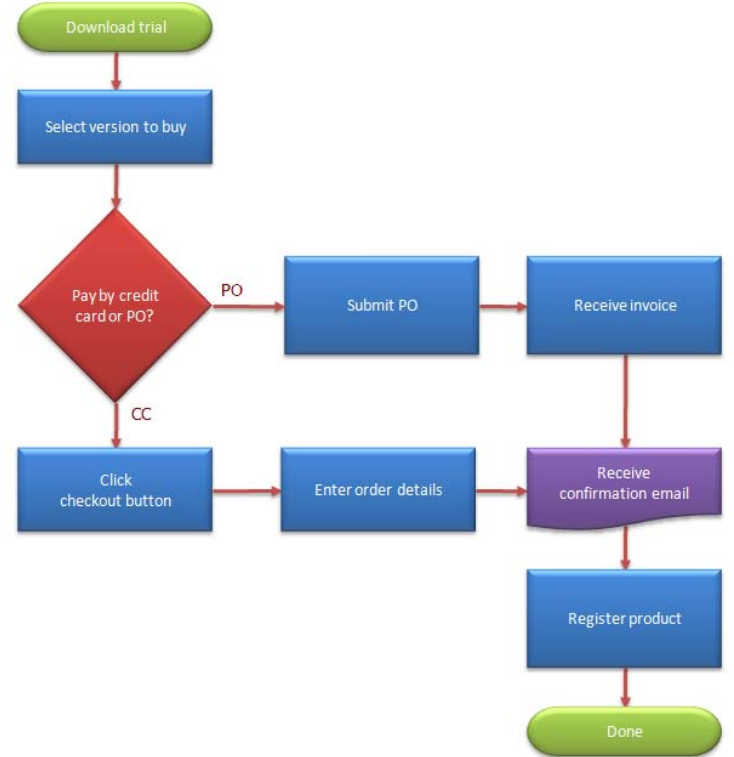


# Akış Şeması

Bilgisayar programlarının işlem basamaklarını geometrik şekillerle gösteren şemadır.

Algoritmanın daha rahat anlaşılabilmesi için şemalarla gösterilmesidir.

Şemada yer alan her şeklin bir kullanım amacı vardır.



# Elips

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



# Paralel Kenar

**Giriş** işlemleri için kullanılır. Dışarıdan bilgi/veri girişini belirtir

Örneğin; klavyeden bir sayı girilmesi istenmesi gibi.

Bir sayı  
giriniz.

İkinci sayıyı  
giriniz



# 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.

İlk sayı = A

# Eşkenar Dörtgen

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

Örneğin; girilen sayı 5'ten büyük mü?

Kalan  
süre  
0'dan  
büyük  
mü?

Oyunda  
başka  
elma var  
mı?

# Dalgalı Dörtgen

**Ekran/ Yazıcı çıktısı için kullanılır.**

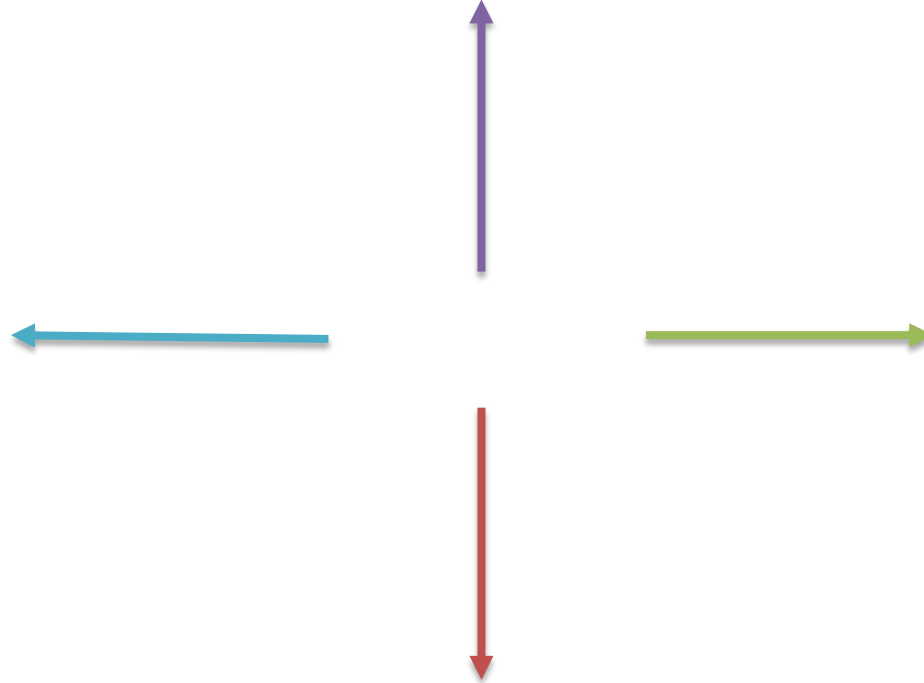
Örneğin; Sonucu Göster

Sonucu göster

Şemsiyeni evde bırak.

# 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: İlk sayı = A

Adım 4: İkinci sayıyı gir.

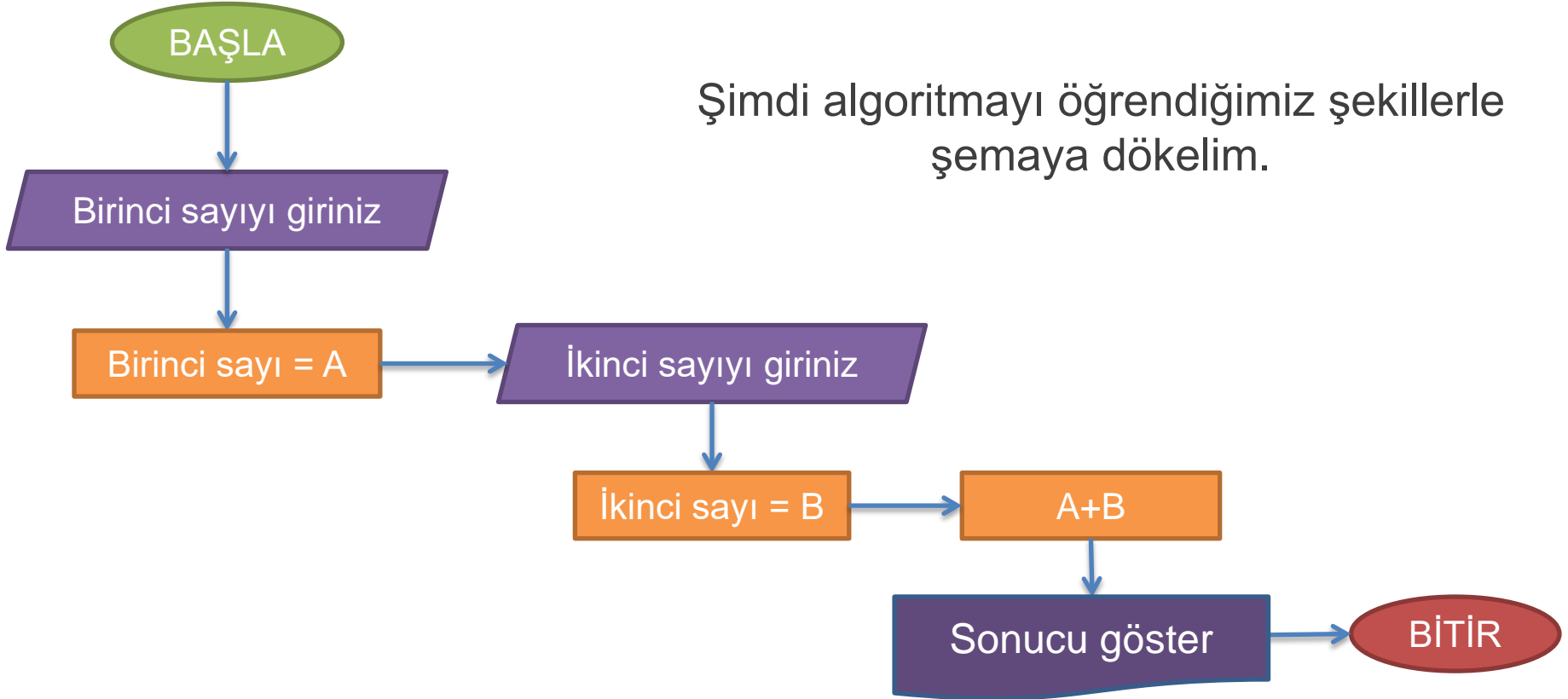
Adım 5: İkinci sayı = B

Adım 6: İki sayıyı topla (A+B)

Adım 7: Sonucu ekranda göster.

Adım 8: Bitir.

# Akış Şeması Örneği



# Akış Şeması Örneđi - 2

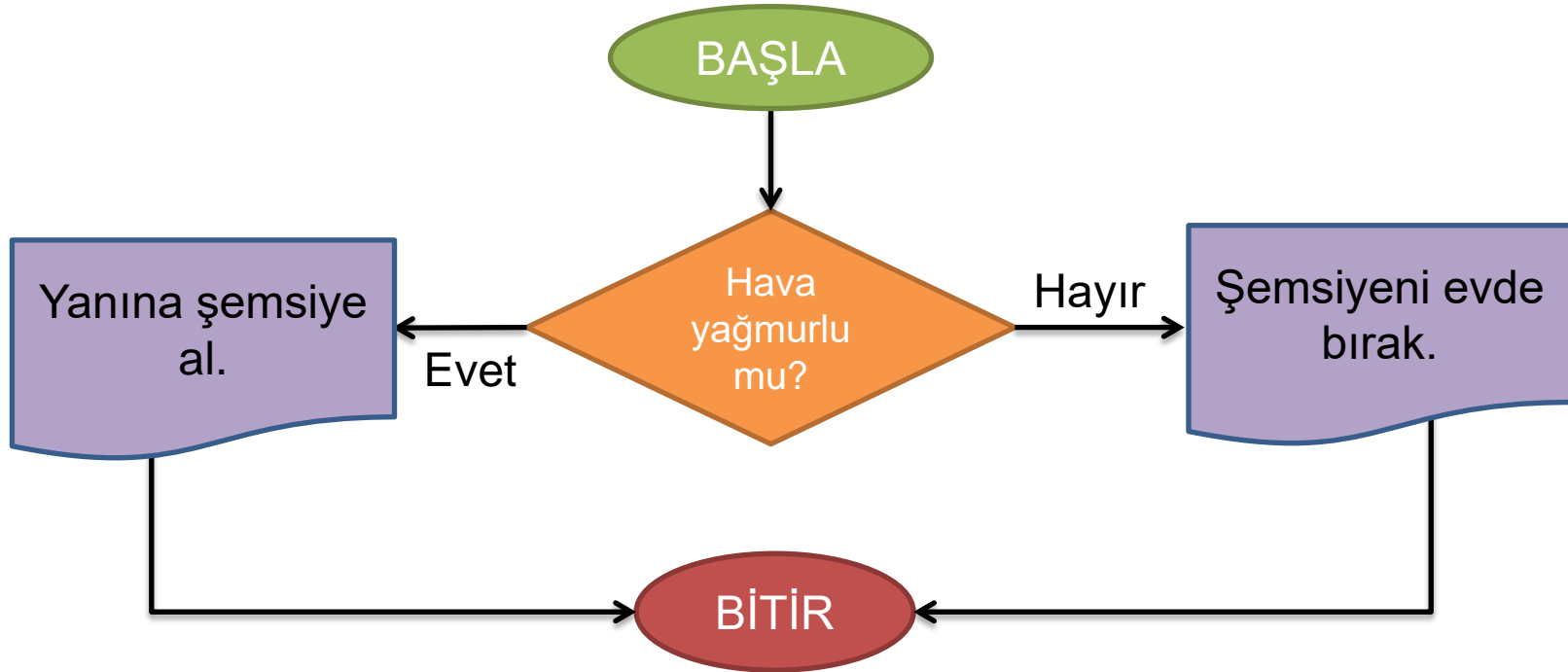
Şimdi hava yağmurlu ise bizi şemsiye almamız konusunda uyaran programın akış şemasını çizeceğiz. Önce **algoritmasını** yazalım.



- 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 7'ye git.
- Adım 6: Şemsiyeyi evde bırak. Adım 7'ye git.
- Adım 7: Bitir.



# Akış Şeması Örneği - 2



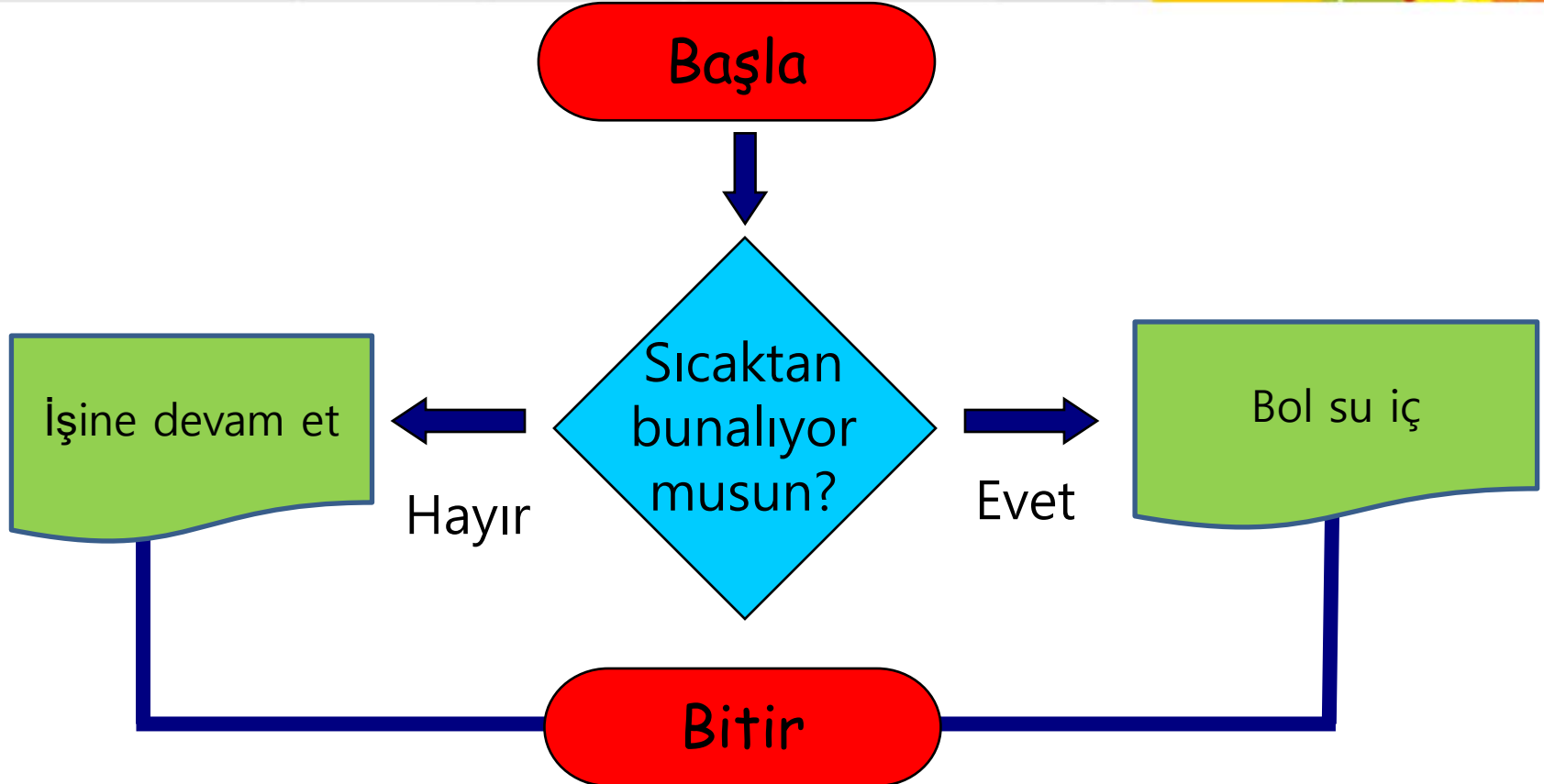


# Örnek Algoritma

Sıcaktan bunalıyor musun?

- ADIM 1: Başla.
- ADIM 2: Sıcaktan bunalıyor musun?
- ADIM 3: Evet ise ADIM 5'e git.
- ADIM 4: Hayır ise ADIM 6'ya git.
- ADIM 5: Bol su iç.
- ADIM 6: İşine devam et.
- ADIM 7: Bitir.

# Örnek Akış Şeması



# Örnek Algoritma

Problem: Verilen bir sayı tek ise; 2 ile, çift ise; 3 ile çarp

ADIM 1: Başla.

ADIM 2: Sayıyı al.

ADIM 3: Sayıyı bir değişken aktar.(Sayı=**a**)

ADIM 4: **a** tek mi çift mi karar ver.

ADIM 5: **a** tekse ADIM 7'ye git.

ADIM 6: **a** çiftse ADIM 8'e git.

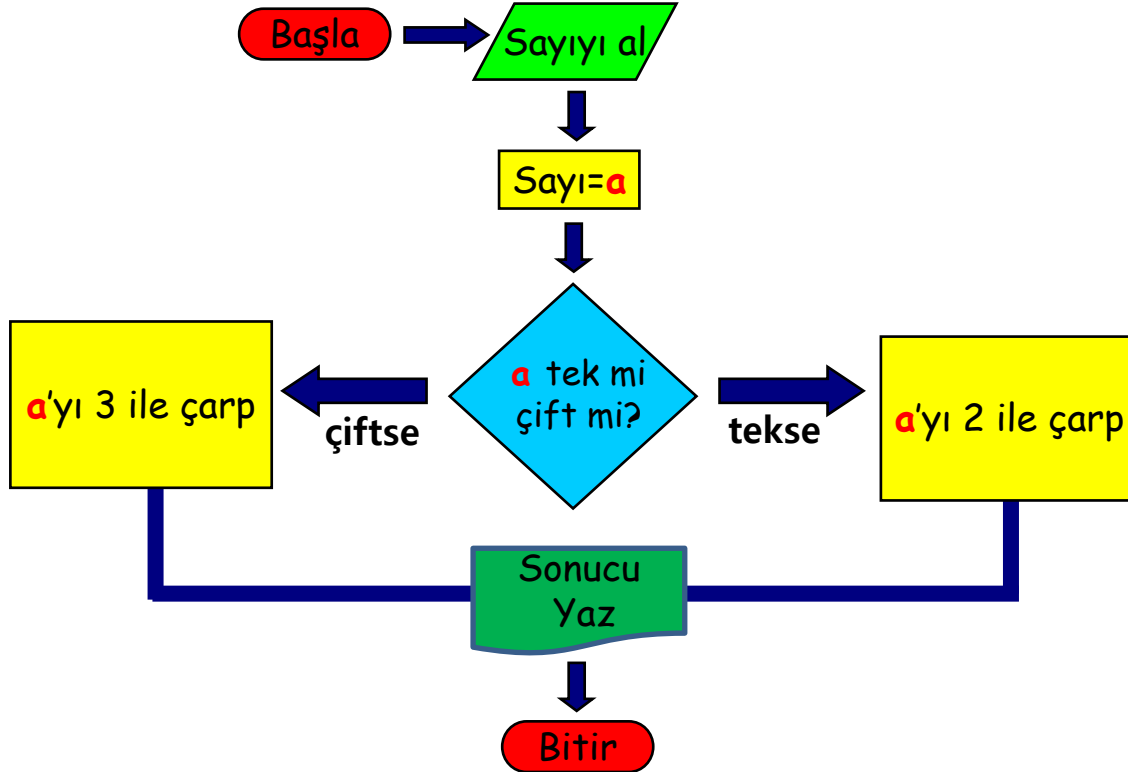
ADIM 7: **a**'yı 2 ile çarp ve ADIM 9'a git.

ADIM 8: **a**'yı 3 ile çarp ve ADIM 9'a git.

ADIM 9: Çarpımı yazdır.

ADIM 10: Bitir.

# Örnek Akış Şeması



# 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 50'den büyük ise **İYİ**, küçük ise **KÖTÜ** kabul edilecek.)





SON