

# PROGRAMLAMA



# DERS NOTLARI

## **PROGRAMLAMA TEMELLERİ:**

Programlamaya yeni başlayanlar için öğrenmenin en iyi yolu değişik kaynaklardan buldukları küçük programları usanmadan yazmak, derlemek ve çalıştırmaktır. Bu küçük programlar, bilgisayarın yapabileceği bütün işleri size öğretecektir. Unutmayınız ki bütün büyük programlar bu küçük programların birleşmesiyle oluşmaktadır.

Dolayısıyla Programlama Temelleri dersinde yazılacak bu küçük programların, programlamayı öğrenmenin tek yol olduğu hiçbir zaman unutulmamalıdır.

Yazacağınız programlarda karşınıza çıkacak yanlışlar, sizi bıktırmamalıdır. Aksine, programlamayı yaptığınız bu yanlışlardan öğreneceğinizden emin olabilirsiniz. Bu nedenle yanlış yapmaktan korkmayınız. Üstelik yeni bir şey öğrenirken bilerek yanlışlar yapıp programların doğurdıkları sonuçları görmek eğlenceli ve öğretici de olabilir.

Programlama dünyasına sürekli olarak yeni dillerin eklendiği ya da mevcut programlama dillerinin yeni sürümlerle geliştirildiği hiçbir zaman unutulmamalıdır. Halen günümüzde kullanılan 2500 'den fazla programlama dilinin olduğu tahmin edilmektedir.

Dolayısıyla kişilerin programlama bilgilerini asla bir programlama diline bağlı tutmamaları gerekir. Çünkü kişinin algoritma ve problem çözme mantığı oluşmamışsa hiçbir programlama dilinde başarılı olamaz.

Ancak programlama temeli ve mantığı oluştuktan sonra çok kısa sürelerde yeni yeni programlama dilleri rahatlıkla öğrenilebilir.

### **Programlama Nedir?**

Program, bir problemi çözmek için bilgisayar dili kullanılarak yazılmış deyim, komut ve fonksiyonlar topluluğudur. Günlük işlerimizde veya resmi kurumlarda yoğun bir şekilde kullandığımız bilgisayarlar ile ortaya çıkan programlar günlük hayatta insanların işlerini oldukça kolaylaştırmıştır.

Örnek olarak, hastanelerde, marketlerde, eğitimde, oyunlarda, bilimsel alanlarda ve daha birçok alanda kullanılan otomasyonlar programlama dilleriyle yazılır.

## Programlama Dili Nedir?

Günlük hayatta konuştuğumuz her dilin kendine ait kelimeleri vardır. Programlama dillerinin de kendine ait benzer kelimeleri vardır. Programlama dillerindeki bu kelimeler, programlama dilinin **syntax** (anahtar) kelimeleridir. Ancak programlama dillerini kullanmak için sadece kelimeleri bilmek yeterli değildir. Eğer bu kelimeler anlamlı bir şekilde bir araya getirilemiyorsa, hiçbir anlam ifade etmeyecektir. Yani, programlama dilinin özelliklerinin nasıl ve ne için kullanıldığı o dilin mantığıdır.

Programlama dünyasına ilk defa adım atıyorsanız yada yeni bir programlama dili öğrenecekseniz mevcut programlama dilleri arasından seçim yapmanız gerekir. Programlama dilleri arasından seçim yaparken her programlama dilinin kendine has özellikleri ve işlevleri göz önünde tutulması gerekir.

Bilgisayar programlama konusunda şu ya da bu programlama dili daha iyidir demek yanıltır. Önemli olan yazılan programın işlevselliğidir. Sizler bir kişiye program yazarken, müşteri için istediği yazılımın hangi program diliyle yazıldığından çok hangi tür işlevleri yerine getirdiği daha önemlidir.

### 1. JAVA

- Java, platformdan bağımsız uygulama denince aklımıza gelen ilk dildir.
- Java daha çok mobil uygulamalarda tercih edilir.
- Java dilini bilmek kurumsal, orta ve büyük çaplı şirketlerde iş bulmada avantaj sağlar.

### 2. C / C++ / C#

- C dilini bilmek diğer programlama dillerini öğrenmekte kolaylık sağlar.
- C dilini bilmek orta ve ileri seviye teknolojik projeler iş bulmamızı sağlar.
- C dili daha çok elektronik & pic uygulamalarında kullanılır.

### 3. PYTHON

- Hem desktop, hem web, hem de server platformlarında oldukça güçlü bir dildir.
- Özellikle, C, Java gibi dillerle birleşik ve yardımcı olarak ta kullanılabilir.
- Linux masaüstü uygulamalarının çoğu python ile yazılmıştır.
- Türkiye'de python kullanan ve bilen oldukça azdır.

### 4. PHP

- Php, özgürlükler dünyasının en çok kullanılan programlama dilidir.
- Hem serbest hem de şirketlerde çok rahat iş bulabilirsiniz.

### 5. NET

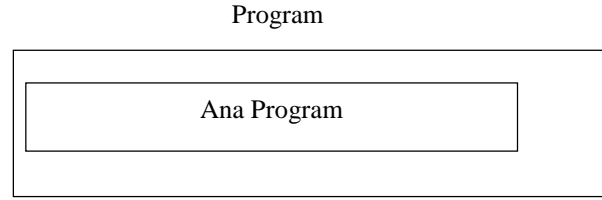
- .NET, windows masaüstü ve web uygulamalarında oldukça güçlüdür.
- Hem serbest hem de şirketlerde çok rahat iş bulabilirsiniz.
- .Net de hemen hemen her türlü problem için hazır bir çözüm bulmanız mümkündür.

## Programlama Türleri:

### 1. Yapısal Olmayan Programlama:

Programlamaya yeni başlayanlar kod yazmaya genellikle küçük ve basit kodlar yazarak başlarlar. Bu kodlar sadece bir **ana program(main)** bloğundan oluşur. Bu blok içerisindeki komut ve deyimler programın tümünde geçerli olan **global** veriler kullanır.

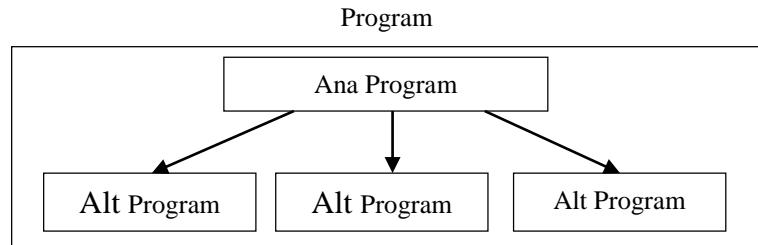
Bu programlama tekniğiyle program yazmak çok büyük programlar için oldukça zordur. Çünkü bu yapıdaki aynı komut ve deyimler yazılan kodlar içerisinde defalarca tekrarlanması gerekebilir. Dolayısıyla bu yapıdaki programların okunabilirliği ve anlaşılabilirliği kod miktarı arttıkça zorlaşmaktadır. Ayrıca kod yazarken hata yapma olasılığı artmakla beraber hata ayıklama (debugging) işlemi de zorlaşmaktadır. Doğal doğal olarak yazılan programın güvenilirliği de oldukça düşecektir. Bu yapıya Basic, Fortran, Cobol gibi diller örnektir.



Yapısal Olmayan Programlama

### 2. Yapısal Programlama:

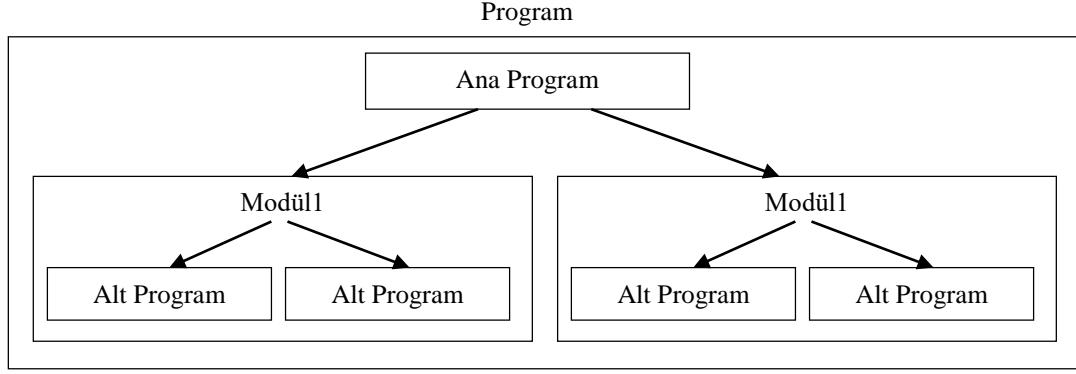
Yapısal Programlama yönteminde, program akışı esnasında, ana program içerisinde altprogramlar tek tek çağırılmaktadır. Bir altprogram çalıştırdıktan sonra, program akışı tekrar ana programa geri dönecek ve programın işleyişi kaldığı yerden devam edecektir. Yapısal programlama tekniğinde kod içerisinde aynı altprogram defalarca çağırılabilir. Böylece gereksiz kod tekrarı ortadan kalktığı gibi, program akışının kontrolünde de çok büyük kolaylıklar sağlanmış olur. Yapısal programlama tekniğinde, altprogram kavramı procedure ve function vasıtasıyla sağlanmaktadır. Bu yapıya Pascal, C gibi diller örnektir.



Yapısal Programlama

### 3. Modüler Programlama:

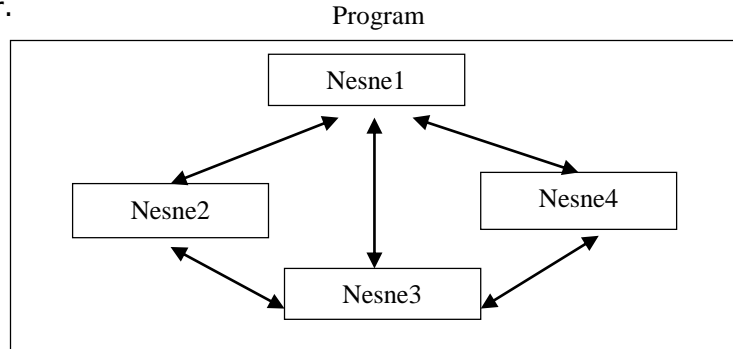
Modüler programlama tekniğinde belli altprogramlar ayrı ayrı modüller içinde gruplandırılır. Her modül içinde ana program içinde tanımlı global değişkenler geçerlidir. Aynı zamanda her modül kendi verisine sahiptir. Bu yapıya Pascal, C gibi diller örnektir.



Modüler Programlama

### 4. Nesne Tabanlı Programlama:

Nesneye yönelik programlama, programcının kendi sınıfını ve nesnesini oluşturup bunun üzerinde işlemler yapmasına olanak sağlar. Bu programlama sisteminin zor olması nedeniyle, çok sayıdaki nesneler önceden programcıya hazır bir şekilde sunulur. Bu yöntemde nesneler birbirlerine mesaj göndererek etkileşim içinde bulunurlar. Nesne yönelimli programlama tekniğinde açık bir biçimde altprogramları çağırmak yerine, direkt olarak ilgili nesneye bir mesaj gönderilir. Nesne kendine gelen mesajı alır ve öncelikle nesnenin bir kopyasını oluşturur. Bu kopya gerekli işlemleri yaptıktan sonra kendini yok eder. Tüm bu işlemlerden nesnenin kendisi sorumludur. Bu yapıya C++ Builder, C#, Java gibi diller örnektir.



Nesne Tabanlı Programlama

### 5. Olay Temelli Programlama:

Olay-temelli programlama (event-driven programming), klasik programlamanın aksine kullanıcıların işlemlerine göre programın yanıt vermesi temeline kurulu bir programlama sistemidir. Kullanıcının fareye tıklaması, klavyeden yazı yazması, programı çalıştırması, programı sonlandırması ya da neden olduğu diğer işlemler birer olay olarak algılanır ve programın işleyişi ona göre yönlendirir. Bu yapıya Visual Basic, Delphi gibi diller örnektir.

## Algoritma Nedir?

Algoritma, bir sorunu çözebilmek için gerekli olan sıralı mantıksal adımların bütünüdür. Günlük yaşamda kullandığımız konuşma dili (sözlü algoritma) kullanılabileceği gibi, grafiksel şekiller (akış şemaları) yardımıyla da algoritmalar oluşturulabilir. Algoritma sadece bilgisayar programlamada değil, hayatın her aşamasında kullanılabilir. Öğrencinin ders çalışma düzeni algoritması, Planlı gezi organizasyon algoritması, Fabrikanın üretim algoritması gibi durumlar örnek gösterilebilir.

## Algoritma Yazmanın Faydaları

1. Program yazımını kolaylaştırır.
2. Program yazımında geçen süreyi kısaltır.
3. Program kontrolünü kolaylaştırır.
4. Hatalı kodlama oranını azaltır.

## Algoritma Yazarken Sağlanacak Kurallar

**1.Netlik:** Algoritmada bulunan anlatım satırları kesin olmalıdır. Kesin olmayan anlatımlar algoritmada bulunmamalıdır. Başka bir deyişle her işlem (komut) açık olmalı ve farklı anlamlar içermemelidir.

**2. Etkinlik:** Algoritmada, her komut, bir kişinin kalem ve kâğıt ile yürütebileceği kadar basit olmalıdır. Algoritmada tekrar anlatımlar olmamalıdır. Bir algoritma bünyesinde ne kadar az tekrar varsa algoritmanın etkinliği o kadar artar. Kaçınılmaz tekrarlarda ise bir algoritmayı etkin hale getirebilmek için; tekrar anlatımların alt algoritma yapılması gerekmektedir.

**3. Sonluluk:** Her türlü olasılık için algoritma sonlu adımda bitmelidir. Her algoritmanın bir bitiş ya da geriye dönüş noktası olmalıdır. Ana algoritmada bitiş noktası END, alt algoritmalarda ise geriye dönüş noktası RETURN komutları ile sağlanır. İşletim sistemleri istisnai olarak sonsuza dek çalışabilir.

**4. Giriş/Çıkış Bilgileri:** Bir algoritmada mutlaka Giriş ve Çıkış bilgisi olmalıdır. Giriş bilgisi, algoritmaya dışarıdan bilgi aktarımını, Çıkış bilgisi ise, algoritma içinde oluşan sonuçların algoritma dışına çıkartılabilmesi işlemidir. Genelde Giriş ve Çıkış işlemleri için Read ve Write (veya Print) kullanılır. Bir bilginin okunabilmesi için değişken kullanılır.

Sonuç olarak programlamanın temeli olan algoritma, problem çözümleri için birbirinden farklı çok sayıda çözüm sunabilmesi dikkat çekici bir noktadır. Bu da gösteriyor ki herhangi bir problemin çözümü için birbirinden farklı yüzlerce program yazılabilir.

## Akış Diyagramı Nedir?

Akış diyagramı problemin çözümüyle ilgili kişiye ortak bir dil veya referans noktası sağlamak amacını taşır. Akış diyagramı, algorithmadaki her bir adımın sembolle gösterilmesidir. Yani algorithmadaki adımlar akış diyagramı sembollerinin içerisinde yazılır.

### Akış Diyagramı Sembolleri



Bir algoritmanın başladığı veya bittiği konumu gösterir.



Giriş-Çıkış komutunun kullanılacağı yeri belirler. Kutu içerisine hangi değişken veya değişkenlere okuma mı, yazma mı yapılacağını belirtmek gerekir.



Klavyeden bilgisayara bilgi girilecek konumu belirtir. Girilecek bilginin hangi değişkene okunacağını kutu içerisine yazılır.



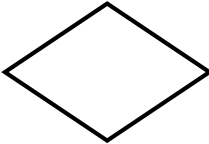
Ekranı temsil eden çıkış sembolüdür. Yazılacak ifade veya değişken bu sembolde belirtilir.



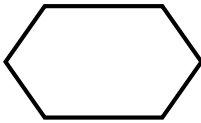
Yazıcıyı temsil eden çıkış sembolüdür.



Aritmetik ve alfabetik işlemlerin gerçekleştirilmesini sağlayan işlem sembolüdür.



Bir algorithmada bir kararın verilmesini ve bu karara göre iki seçenekten birinin uygulanmasını sağlayan şekildir. Bu sembol içerisine kontrol edilecek mantıksal koşul yazılır. Mantıksal koşulun doğru olması durumunda "Evet" yazılan kısma Yanlış olması durumunda "Hayır" yazılan kısma sapılır. Tek girişli ve çift çıkışlı bir şekildir.



Bir işlemin belli bir sayıda veya belli bir koşul doğru olduğu sürece tekrar edilmesini sağlayan döngü komutunu temsil eder. Bu sembol içerisine ya koşul ya da döngünün başlangıç, adım ve sonlanma değerleri belirtilir. Döngü olarak belirlenen blokta da tekrar edilmek istenen adımlar yer alır.



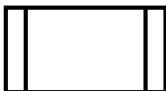
Akış sembolü olup, akış diyagram sembollerini birbirine bağlar.



Bağlama sembolü olup akış şemasının aynı sayfadaki bölümleri sayı/harfle birleştirerek akış şemasındaki kopuklukları bağlar. Ayrıca döngülerin blok sonunu da temsil eder.



Bağlama sembolü olup, akış şemasının farklı sayfalardaki bölümleri sayı/harfle birleştirerek akış şemasındaki kopuklukları bağlar.



Alt algoritma bloğunu gösterir. Kutu içerisine alt algoritma bloğunun adı yazılır.

**Örnek:** Klavyeden girilen iki sayının ortalamasını hesaplayan algoritma ve akış şeması.

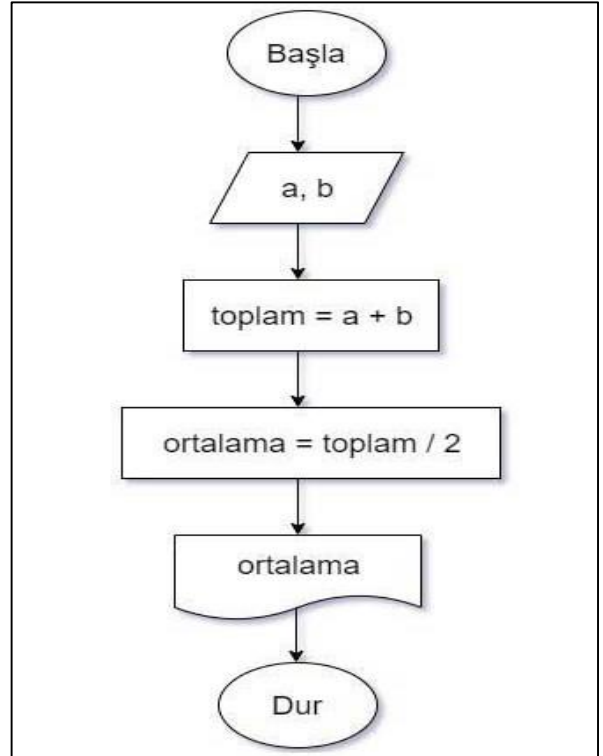
**Problemi Tanımlama:** Bu program klavyeden girilen iki adet sayının ortalamasını bulur.

**Analiz Yapma:**

- a. **Girdiler:** İki adet sayı
- b. **İşlem:**  $\text{Toplam} = A + B$   
 $\text{Ortalama} = \text{toplama} / 2$
- c. **Çıktılar:** Ortalama

**Tasarımı Oluşturma:**

- A1. Başla
- A2. "Birinci sayıyı gir" ; A
- A3. "İkinci sayıyı gir" ; B
- A4. İki sayı topla
- A5. Ortalamayı hesapla
- A5. Ortalama yazdır
- A6. Dur



**Örnek:** Tabanı ve yüksekliği girilen üçgenin alanını hesaplayan algoritma ve akış şeması

**Problemi Tanımlama:** Bu program üçgenin alanını hesaplar.

**Analiz Yapma:**

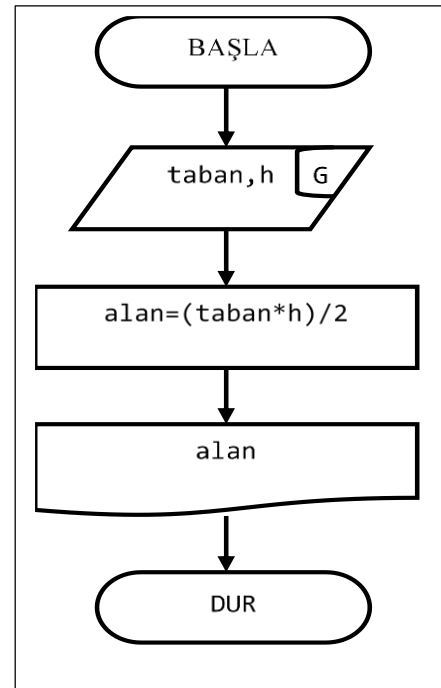
**Girdiler:** Taban ve Yükseklik

**İşlem:**  $\text{alan} = (\text{taban} * h) / 2$

**Çıktılar:** Üçgenin Alanı

**Tasarım Oluşturma:**

- A1. Başla
- A2. OKU taban
- A3. OKU h
- A4.  $\text{alan} = (\text{taban} * h) / 2$
- A5. YAZ alan
- A6. Dur





**Örnek:** Birbirinden farklı olarak verilen iki adet sayıdan, büyük olanı bulup gösteren algoritma ve akış diyagramını tasarlayınız.

A1. Başla

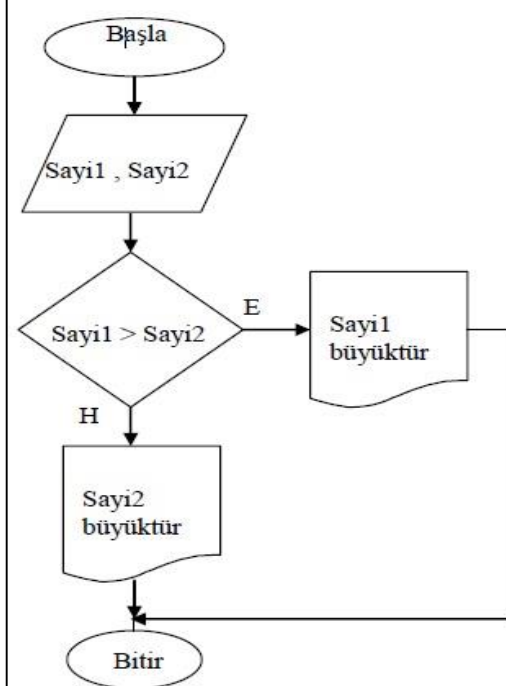
A2. OKU sayı1

A3. OKU sayı2

A4. Eğer sayı1 > sayı2 ise YAZ sayı1

değilse YAZ sayı2

A5. Bitir



**Örnek:** Klavyeden girilen 2 sayıyı karşılaştırıp sonucu ekrana yazdıran programın algoritmasını yazarak akış diyagramını çiziniz?

A1. Başla

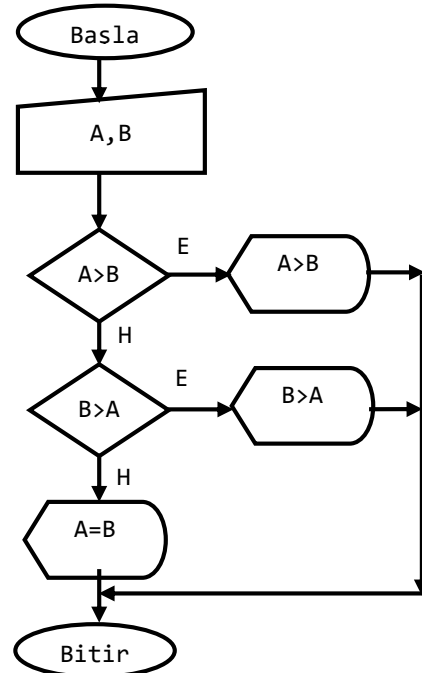
A2. OKU A , B

A3. Eğer A > B ise Yaz " A > B "

Değil Eğer B > A ise YAZ " B > A "

Değilse YAZ " A = B "

A4. Bitir



**Örnek:** Kenar uzunlukları girilen dikdörtgenin alanını hesaplayıp ekranda gösteren programın algoritmasını yazınız. (Not: Kenar uzunlukları negatif veya sıfır olarak girildiği durumda veri girişi tekrarlanacaktır.)

A1. Başla

A2. OKU kkenar

A3. Eğer kkenar  $\leq 0$  ise git A2

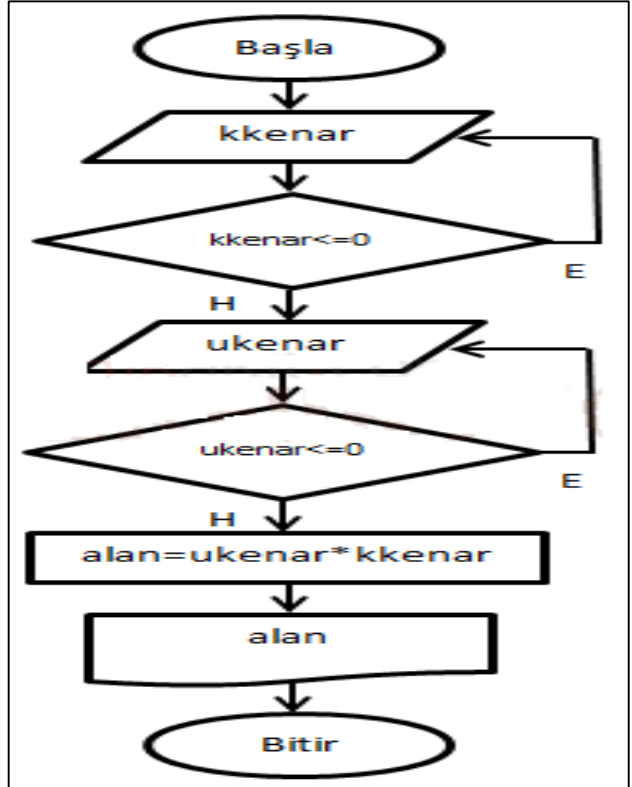
A4. OKU ukenar

A5. Eğer ukenar  $\leq 0$  ise git A4

A6. alan = ukenar \* kkenar

A7. YAZ alan

A8. Bitir



**Örnek:** Klavyeden girilen bir sayının işaretini kontrol ederek sonucu ekrana yazdıran programın algoritmasını yazarak akış diyagramını çiziniz?

A1. Başla

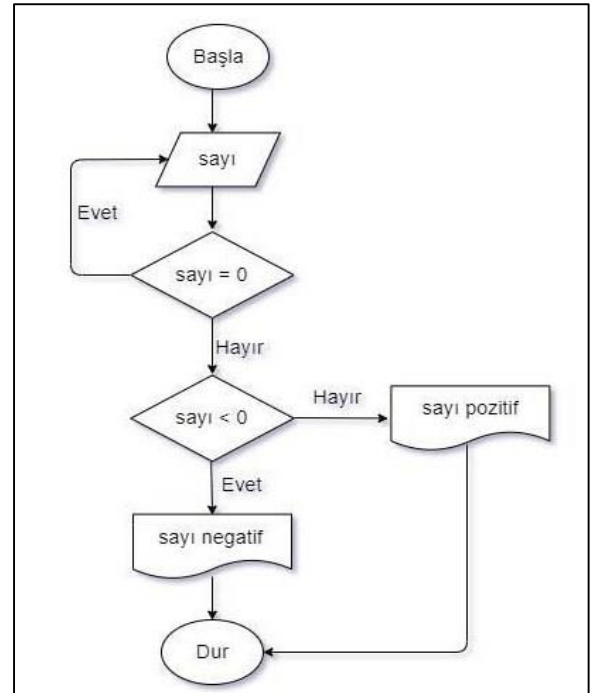
A2. Sayı oku

A3. Eğer sayı = 0 ise 2. adıma git

A4. Eğer sayı < 0 ise YAZ " Sayı Pozitif "

Değilse YAZ " Sayı Negatif "

A5. Dur



**Örnek:** Klavyeden girilen iki sayıdan birincisi büyükse çıkarma, ikincisi büyükse toplama, eşitse çarpma işlemi yapan problemin algoritmasını yazarak akış diyagramını çiziniz?

A1. Başla

A2. X , Y sayılarını gir

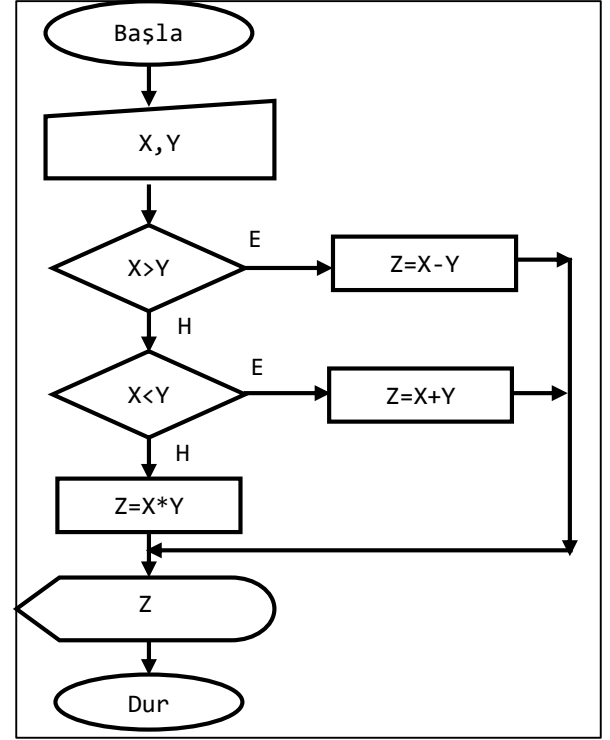
A3. Eğer  $X > Y$  ise  $Z = X - Y$  ; 6. adıma git

A4. Eğer  $X < Y$  ise  $Z = X + Y$  ; 6. adıma git

A5.  $Z = X * Y$

A6. " Sonucu yaz " , Z

A7. Dur



**Örnek:** Girilen vize ve final notlarına göre öğrencinin dersten geçip geçmediğini bulan algoritma ve akış diyagramını tasarlayınız.

A1. Başla

A2. YAZ "Vize notunu gir"

A3. OKU vize

A4. YAZ "Final notunu gir"

A5. OKU final

A6. ortalama = vize \* 0.40 + final \* 0.60

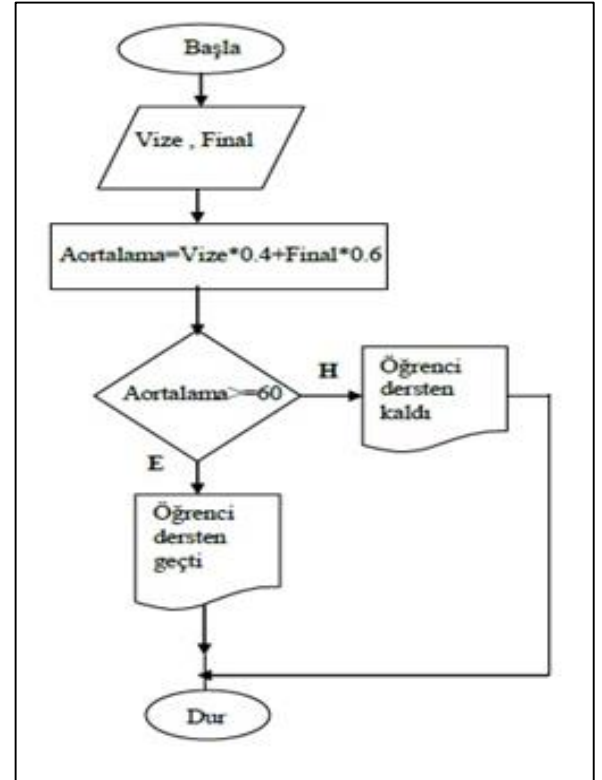
A7. Eğer ortalama  $\geq 60$  ise

YAZ " Öğrenci Dersten Geçti "

değilse

YAZ " Öğrenci Dersten Kaldı "

A8. Dur



**Örnek:** Ekrana 10 defa adınızı yazdırınız.

A1. Başla

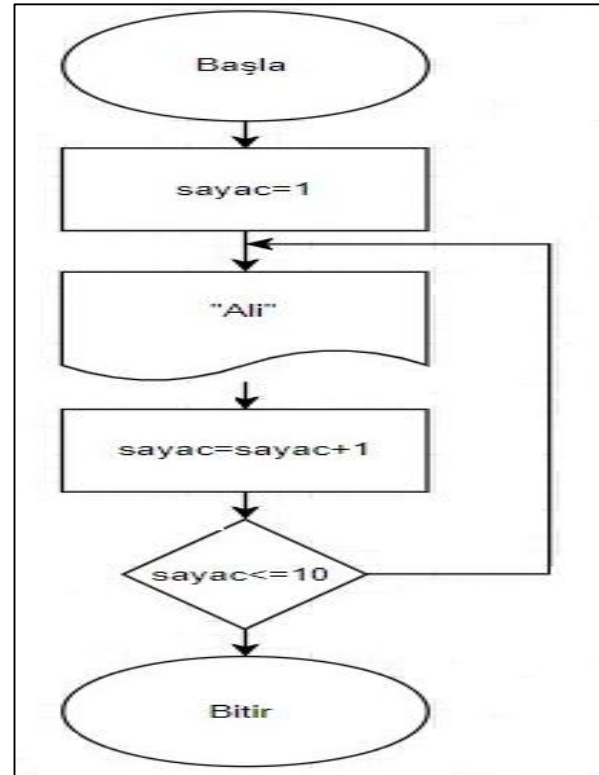
A2. Sayac = 1

A3. YAZ "Ali"

A4. Sayac = Sayac + 1

A5. Eğer Sayac  $\leq$  10 ise git Adım 3

A6. Bitir



**Örnek:** 0'dan 100 'e kadar olan çift sayıların toplamını veren programın algoritma.

A1. Başla

A2. Topl = 0 ; S = 0

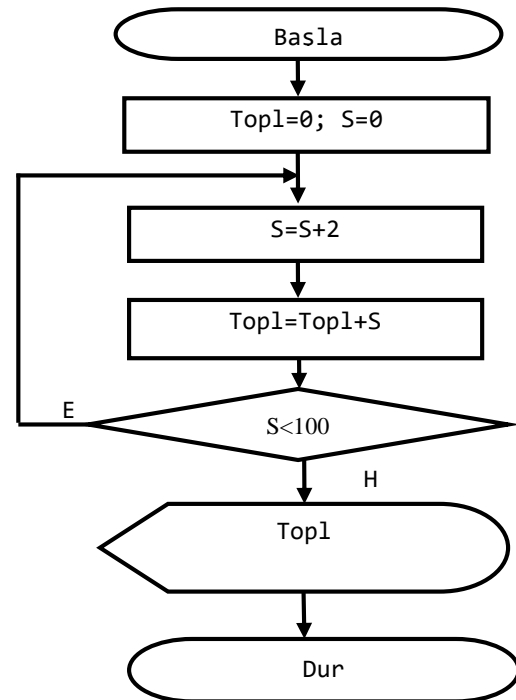
A3. S = S + 2

A4. Topl = Topl + S

A5. Eğer S  $\leq$  100 ise 3. adıma git

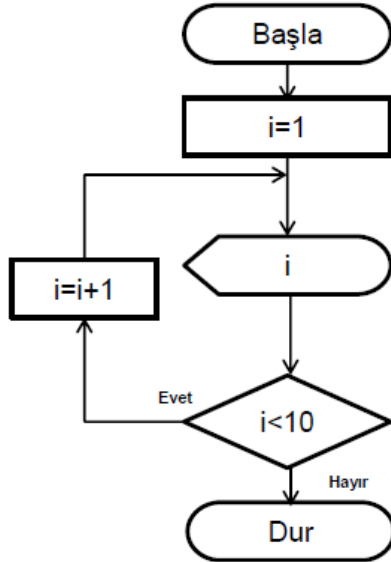
A6. Toplamı yaz

A7. Dur

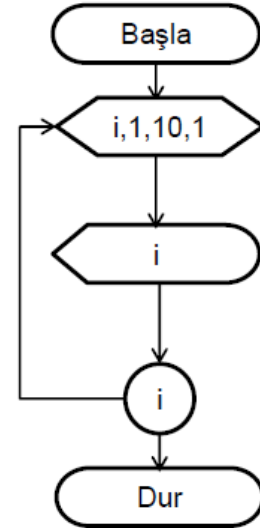


**Örnek:** 1' den 10 'a kadar olan sayıları ekran listeleyen akış diyagramı.

### Kontrol İfadesiyle

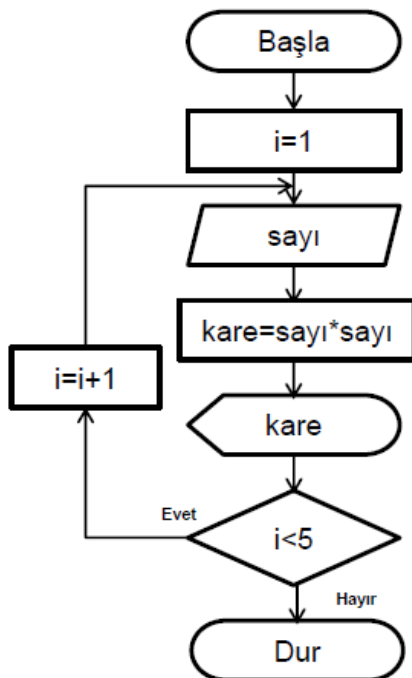


### Döngü İfadesiyle

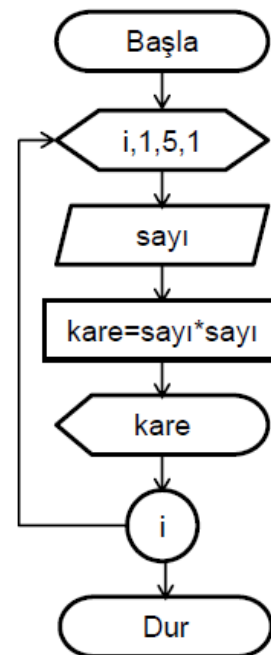


**Örnek:** Klavyeden girilen 5 adet sayının karesini hesaplayan akış diyagramı.

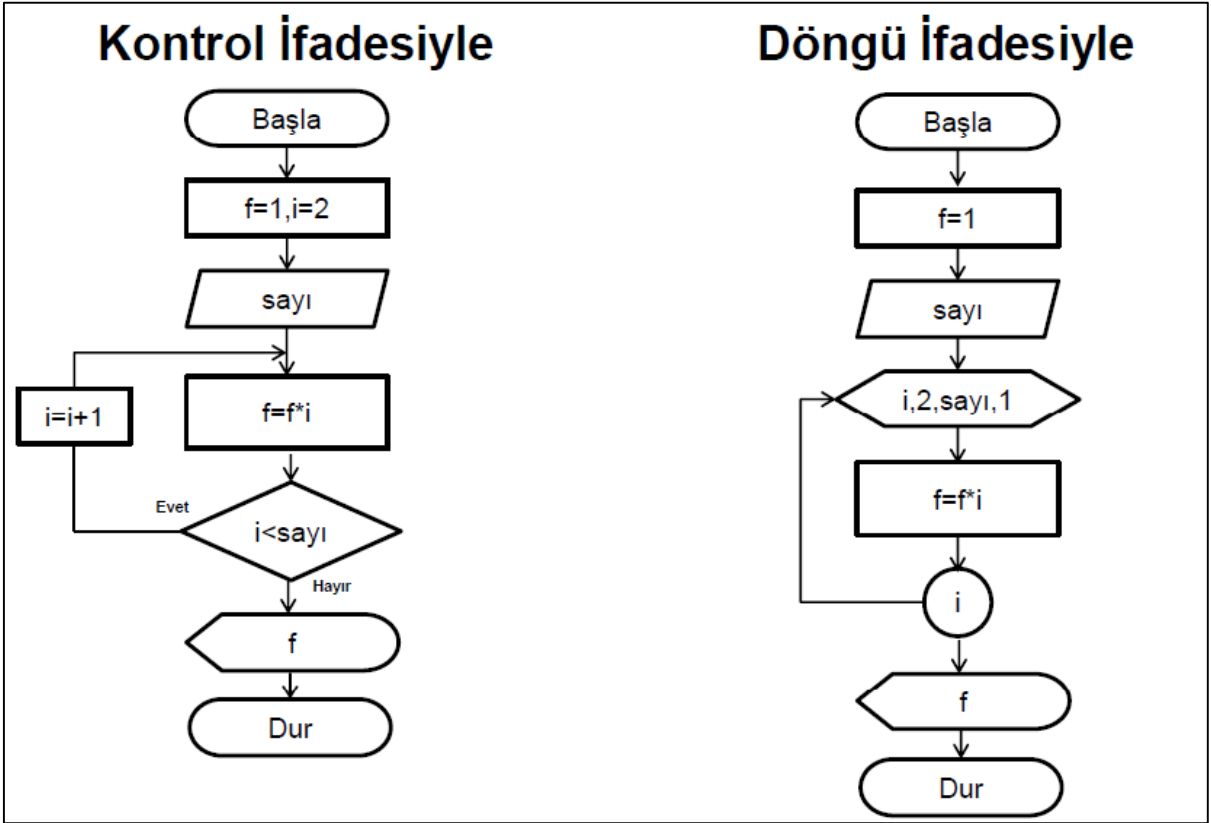
### Kontrol İfadesiyle



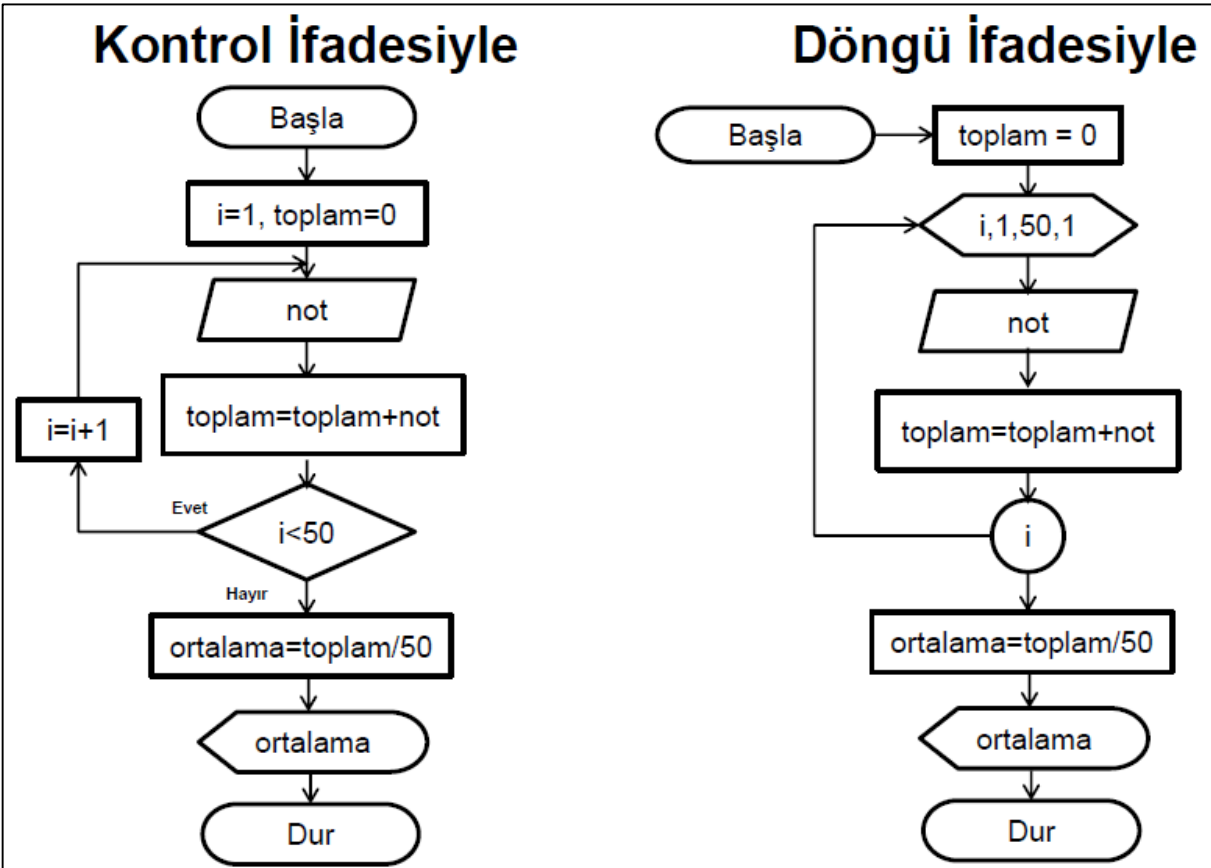
### Döngü İfadesiyle



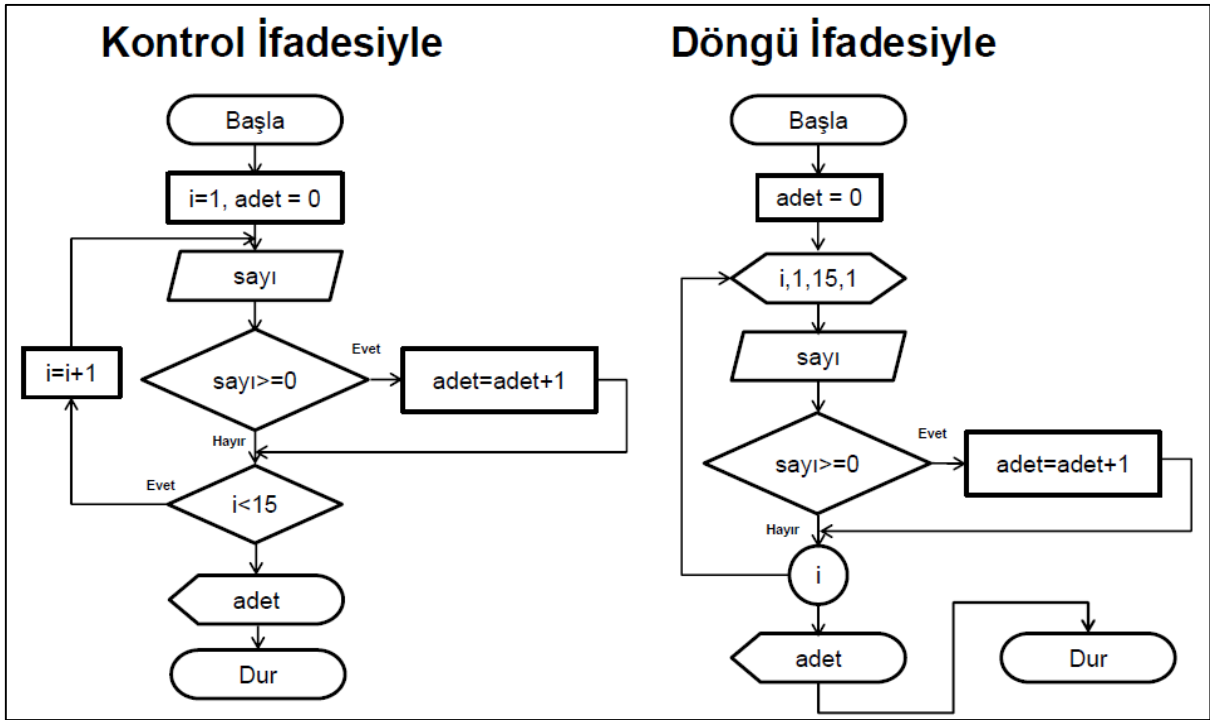
**Örnek:** Klavyeden girilen sayının faktöriyelini hesaplayan akış diyagramı.



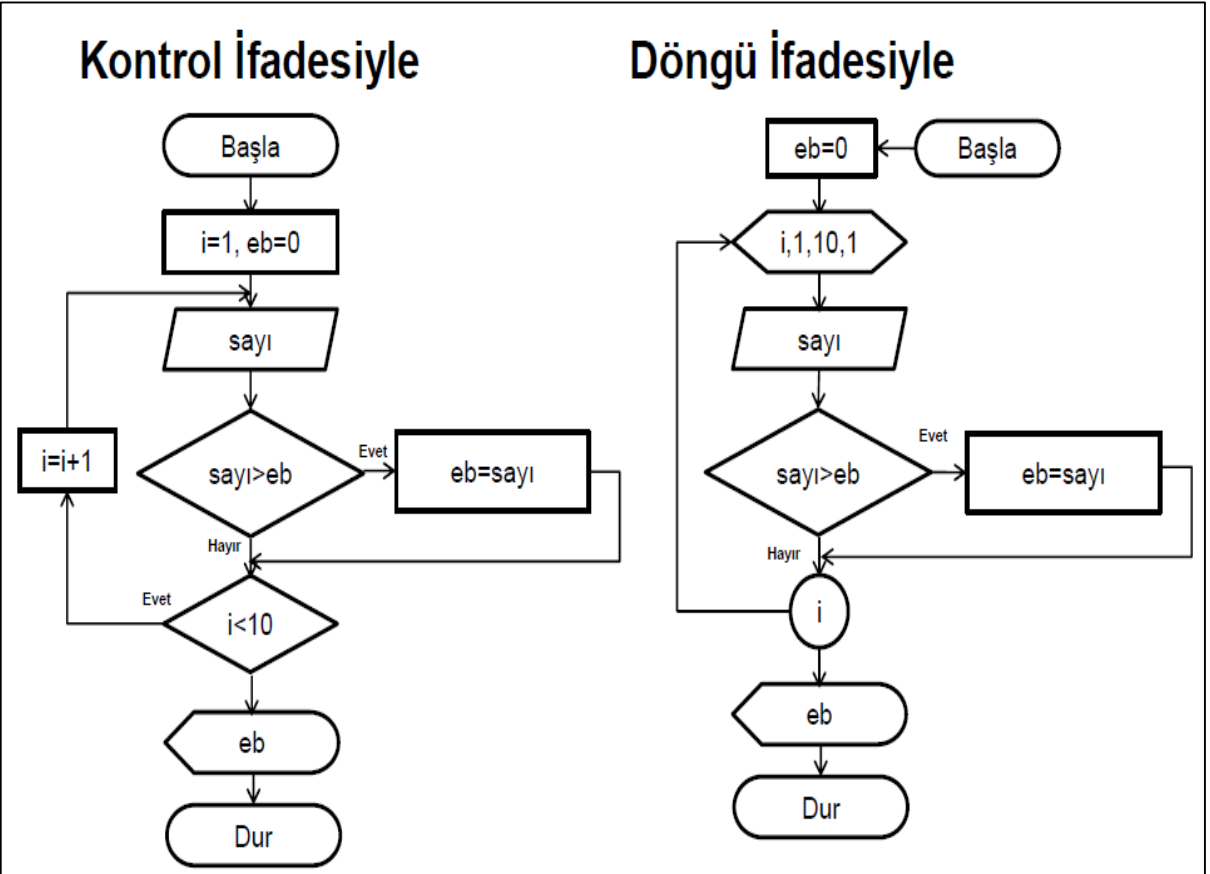
**Örnek:** Klavyeden girilen 50 adet notun ortalamasını bulduran akış diyagramını çiz.



**Örnek:** Klavyeden girilen 15 adet sayıdan pozitif olanların adedini bulan akış diyagramı.



**Örnek:** Klavyeden girilen 10 adet sayıdan en büyüğünü bulan akış diyagramı.



## **JAVA PROGRAMLAMA DİLİNE GİRİŞ**

Java, James Gosling tarafından 1995 yılında piyasaya sürülmüştür. Java hakkında araştırma yaptığınızda karşınıza çıkacak ilk slogan tartışmasız **"Write once, Run Anywhere - Bir kere yaz, her yerde çalıştır"** olacaktır. Bu da gösteriyor ki, java destekleyen her aygıtta yeniden düzenlemeye gerek kalmadan direkt çalıştırılabileceği anlamına gelmektedir.

Java'nın şu anda kullanmakta olduğumuz sürümü **JAVA SE (Standard Edition) 8**, 2014 yılının Mart ayında bizlerle buluştu. SE serisinin başlangıcı ise 2006 yılına dayanmaktadır. 13 Kasım 2006 yılında, Sun firması; JAVA SE 6 sürümünü büyük yeniliklerle ücretsiz ve açık kaynaklı olarak piyasaya sürmüştür. 2010 yılında ise Oracle, Sun firmasını satın alarak Java dilinin yeni sahibi olmuştur.

### **Neden Java Programlama Dilini Öğrenmeliyim?**

1. Java, birçok platformdan derlenen veriler incelendiğinde en fazla kullanılan dildir.
2. Java, JDK yüklü olduğu her yerde çalıştırılabilir. Herhangi bir platforma bağlı değildir.
3. Java, tamamıyla nesne yönelimli bir dildir.
4. Java, tamamen açık kaynak bir yazılım dilidir.
5. Java, yüksek güvenliği sayesinde virüssüz, şişmeyen sistemler yazılabilen bir dildir.
6. Java, direkt makine diline çevrildiğinden ve de hiç bir yerde saklanmadığından hızlıdır.
7. Java, aynı anda birden fazla işlem yapabilen projelerin yazılabildiği bir dildir.
8. Java, serbest uygulama mimarisi yapısıyla taşınabilir bir sistem dilidir.
9. Java, derleme ve çalışma zamanı hata denetimini kullandığından sağlam bir dildir.
10. Java, günümüzün mobil işletim sistemi olan Android için uygulama yazılan bir dildir.

### **Java'nın Kullanıldığı Alanlar:**

Java şu anda yaklaşık olarak 3 milyar cihazda çalışmaktadır. Javanın en çok kullanıldığı alanlardan bazıları şunlardır;

- ✓ Acrobat Reader, Media player, Antivirus gibi bir çok masaüstü uygulamaları,
- ✓ Bir çok Framework ve web uygulamaları,
- ✓ Kurumsal ve banka uygulamaları,
- ✓ Mobile uygulamaları,
- ✓ Akıllı kart uygulamaları,
- ✓ Robotik sistem uygulamaları,
- ✓ Oyun Uygulamaları,
- ✓ Gömülü sistemlerde ve daha bir çok alanda kullanılabilmektedir.



## Java Kurulumu:

Bir Java programı önce Notepad benzeri bir editör ile oluşturulan bir metin dosyası içine yazılır. Kaynak dosya adı verilen bu dosyanın uzantısı **.java**'dır. Kaynak dosya **javac.exe** programı kullanılarak derlenir ve **bytecode** adı verilen **.class** uzantılı bir dosya oluşturulur. Bu dosya herhangi bir işletim sisteminde yüklü olan **JVM** tarafından çalıştırılabilmektedir.

## JVM (Java Virtual Machine) :

Bildiğimiz üzere Java'nın en temel ve en cazip özelliği her işletim sisteminde çalışabilmesidir ( Bir kere yaz her yerde çalıştır). Her işletim sisteminde sanal makine çalışmasını sağlayan yapı ise **JVM**'dir. Bizim yazdığımız ".java" uzantılı java kodlarımız derlendiği zaman ".class" uzantılı **byte code**'lu dosyalar üretilir.

## JRE (Java Runtime Environment):

İçerisinde **JVM** barındıran ve Java uygulamalarını çalıştırabilmemizi sağlayan yazılım paketidir.

## JDK (Java Development Kit):

Programcıların yazdıkları kodları derleyip çalıştırmak için gerekli kurulumdur. Paketin içerisinde javac derleyicisi, Java Sanal Makinesi(Java Virtual Machine) ve diğer yardımcı paketler bulunur.JDK aşağıdaki bağlantı adresinden indirilebilir;

**<http://www.oracle.com/technetwork/java/javase/downloads/index.html>**

## Java Ayarları:

- \* Bilgisayarım/Özellikler/Gelişmiş Sistem Ayarları/Gelişim/Ortam Değişkenleri/Yeni seçilir.
- \* Değişken adı PATH, Değeri ise C:\Program Files\Java\jdk1.8.0\_221\bin yolu seçilir.
- \* Değişken adı CLASSPATH, Değeri ise C:\Program Files\Java\jre1.8.0\_251\lib yolu seçilir.

## IDE (Integrated Development Environment) :

Kodlarımızı yazarken bize yardımcı bir geliştirme ortamı gerekmektedir. Aslında kodlarımızı notepad ile yazıp da çalıştırabiliriz, fakat bu kullanışlı bir yöntem değildir. Dolayısıyla Java ile uygulama geliştiren kişilerin genel olarak kullandıkları ortak programlar Eclipse ve NetBeans'dir. İkisinde açık kaynak kodlu lisans gerektirmeyen programlardır. Netbeans paketini **<https://netbeans.apache.org/download>** adresinden indirebilirsiniz.

## Java Klasör Yapısı:

Netbeans Java Projesi **File/New Project/Java Application/Project Name** adımlarından proje isimlendirilerek oluşturulur. Bu projeye class eklemek için proje adı farenin sağ tuşuyla seçilerek **new/class** penceresinden **Java Main Class** seçilerek class isimlendirilir. Bir Java projesinde kaynak kodlarımız **src** adlı klasörde tutulur. Oluşturacağımız her türlü Java uzantılı dosya (Java sınıfları) bu klasörde saklanır ve uygulamayı çalıştırdığımızda bu klasör içinden derlenir.

Bir Java projesinin çalışabilmesi için en az şu bileşenleri içermesi gerekmektedir.

```
public class HelloWorld {  
    public static void main(String[] args) {  
        System.out.println("Hello World!");  
    }  
}
```

program ismi (sınıf)  
komutların ismi (method)  
komutlar

- ✓ Her proje en az bir sınıf içermelidir.
- ✓ Her sınıf en az bir metod içermelidir.
- ✓ Her metod en az bir komut içermelidir.

**System:** Sistem tarafından belirlenen sınıfın imkanları ölçüsünde girdi, standart çıktı ve hata çıktısı vermesi ile popülerdir. Bunlar dışında tanımlanmış özellik ve değişkenlere erişmesi, dosya ve kütüphanelerin yüklenebilmesi ve herhangi bir diziden bir bölümü hızla kopyalamak için bir yöntem olarak kullanılabilir.

**out:** Adından da anlaşıldığı üzere bilgisayarın standart çıktı bölümüdür.

**println:** Standart olarak ekrana yazdırma işlemini üstlenir.

<b>sout+tab:</b> System.out.println <b>soutv+tab:</b> System.out.println <b>psvm+tab:</b> public static void main <b>st+tab:</b> String değişken <b>db+tab:</b> double değişken <b>bo+tab:</b> boolean değişken <b>wh+tab:</b> while döngüsü <b>dowhile+tab:</b> do while döngüsü <b>fori+tab:</b> for döngüsü <b>forl+tab:</b> for list döngüsü <b>fore+tab:</b> foreach döngüsü <b>iff+tab:</b> if yapısı <b>ifelse+tab:</b> if else yapısı <b>sw+tab:</b> switch case yapısı <b>br+tab:</b> break <b>cn+tab:</b> continue <b>re+tab:</b> return	<b>pr+tab:</b> private <b>pu+tab:</b> public <b>pe+tab:</b> protected <b>trycatch+tab:</b> try-catch kalıbı <b>newo+tab:</b> Nesne kalıbı  <b>Alt + Insert :</b> Kod Ekle <b>Ctrl + Enter:</b> Yukarı satır açar. <b>Shift + Enter:</b> Aşağı satır açar. <b>Ctrl + Shift + Up:</b> Yukarı satır kopyalar. <b>Ctrl + Shift + Down:</b> Aşağı satır kopyalar. <b>Alt + Shift + Up:</b> Yukarı satır taşır. <b>Alt + Shift + Down:</b> Aşağı satır taşır. <b>Ctrl + E / Shift + Delete :</b> Satır siler. <b>Ctrl+Delete / Ctrl+Backspace:</b> Kelime siler. <b>Ctrl+Shift+C/Ctrl+ /:</b> //Yorum satırı <b>bcom:</b> /* */ Yorum blok
----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

## **JAVA PROGRAMLAMA DİLİNİN TEMEL ÖĞELERİ:**

### **Java İsimlendirme Kuralları:**

İsimlendirme kuralları hem bireysel hem de bir ekiple yazılım geliştirenler için küçük gibi görünen ama önemli bir konudur. Bunun için belirli isimlendirme kuralları oluşturulmuştur. Namespaceler, classlar, metodlar, değişkenler ve sabitler için yazılan programlama dilinin isimlendirme kurallarını uygulamak ve mantıklı isimler vermek yazılımcılara kolaylık sağlar. Yazılım geliştirdiğimiz dilin isimlendirme kurallarını öğrenip uygulamak önemli bir detaydır.

### **Camel Case Nedir ?**

Camel Case, bir tanım içindeki her kelimenin ilk kelime dışındaki kelimelerin ilk harfleri büyük harflerle yazıldığı bir adlandırma kuralıdır. Yazılım geliştiricileri genellikle kaynak kodu yazarken "Camel Case" kullanır.

Camel Case kullanımı zorunlu olmamakla beraber yazılım dilinin jargonudur ve genelde tüm yazılımcılar bu kurala uyar. Bu kural sayesinde değişken isimleri daha okunur olur.

### **Örnek : yemekVakti, vizeOrtalamaNotu, javaProgDersleri**

**Upper Camel Case:** Upper Camel Case isimlendirme kuralı ise, bir tanım içindeki bütün kelimelerin ilk harflerinin büyük olmasıdır.

**Lower Camel Case:** Lower Camel Case isimlendirme kuralı ise, bir tanım içindeki birinci kelimenin ilk harfi küçük, geri kalan kelimelerin ilk harflerinin büyük olmasıdır.

**Screaming Snake Case:** Bütün harfler büyük yazılır. Sabit isimlendirmede kullanılır.

**Snake Case:** Kelimeler alt tire ( \_ ) ile birbirine bağlanır.

**Upper snake case** → örnek: Hello\_World

**Lower snake case** → örnek: hello\_world

### **Java dilinin İsimlendirme Kuralları**

Sınıflar için upper camel case kullanılır.

**Örnek: HelloWorld**

Metotlar için lower camel case kullanılır.

**Örnek: helloWorld**

Değişkenler için lower camel case kullanılır.

**Örnek: helloWorld**

Sabitler için screaming snake case kullanılır.

**Örnek: HELLO\_WORLD**

## Tanıtıcı isimlendirme kuralları şunlardır;

- ✓ Latin alfabesi (Türkçe karakterler dahil) ve 0 – 9 arasındaki rakamlardan oluşmalı,
- ✓ Tanıtıcı isim rakamla başlamamalı veya sadece rakamdan oluşmamalı,
- ✓ Tanıtıcı isim Java anahtar kelimelerinden birisi olmamalı,
- ✓ Sembollerden sadece alt tire ( \_ ) ve \$ işaretleri kullanılabilir,
- ✓ Tırnak içindeki ifadelerde küçük veya büyük harf kullanılabilir,
- ✓ Matematiksel ifadelerde ondalıklı sayıların ayırıcı program yazımında nokta (.) iken, klavyeden değer girişlerinde virgül (,) olduğuna dikkat edilmeli,
- ✓ Java dilindeki komutların / tanıtıcıların küçük-büyük harf duyarlı olduğu unutulmamalı,

Geçerli Tanıtıcılar		Geçersiz Tanıtıcılar		
AD_Soy	Geçerli	Yaşıt!	Geçersiz	Sembol kullanılmış
vize1	Geçerli	6.sokak	Geçersiz	Rakam ve sembol kullanılmış
VIZE2	Geçerli	float	Geçersiz	Java kelimesi kullanılmış

Yazım hatalarını en aza indirmek için çok dikkatli olmalıyız;

- ✓ Genellikle java dizilimlerinin sonu ; ile biter.
- ✓ Method ya da sınıf bloklarının hepsi { } içerisinde yazılır.
- ✓ Çıktılar ' (tek tırnaklar) veya " (çift tırnaklar) arasında verilen değerlerdir.
- ✓ Çıktı verirken bir alt satıra inmek için enter tuşu işlevsizdir.

**Semboller:** Harf ve rakam dışında kalan karakterlerdir.

**a. Ayırıcılar: Boşluk** , . , , , : , ; , = , ' ' , " " , \

**b. Parantezler:** ( ) , [ ] , { }

**c. Operatörler:** + , - , \* , / , = , ! , && , || , % , ++ , -- , += , -= , \*= , /=

**d. Yorum Satırı:** / \* \*/ , // Yorum satırları derleyici tarafından dikkate alınmaz.

## Anahtar Kelimeler:

abstract		continue		for		new		switch
assert		default		goto		package		synchronized
boolean		do		if		private		this
break		double		implements		protected		throw
byte		else		import		public		throws
case		enum		instanceof		return		transient
catch		extends		int		short		try
char		final		interface		static		void
class		finally		long		strictfp		volatile
const		float		native		super		while

Aşağıdaki listede, kategorilere göre gruplandırılmış anahtar kelimeler gösterilmektedir:

**Erişim değişkenleri :** private, protected, public

**Sınıf, metot, değişkenler:** abstract, class, extends, final, implements, interface, native, new, static, strictfp, synchronized, transient, volatile

**Akış Kontrolü:** break, case, continue, default, do, else, for, if, return, switch, while

**Paket Ekleme:** import, package

**Veri Tipleri:** boolean, byte, char, double, float, int, long, short

**Hata İşleme:** assert, catch, finally, throw, throws, try

**Sayım:** enum

**Diğer:** super, this, void

**Kullanılmayan:** const, goto

## Java Operatörleri:

### 1. Aritmetik Operatörler

Operatör	İşlevi	Örnek
+	Toplama / birleştirme operatörüdür	a + b
-	Çıkartma operatörüdür	a - b
*	Çarpma operatörüdür	a * b
/	Bölme operatörüdür	a / b
%	Mod alma operatörüdür	a % b
++	1 arttırma operatörüdür	a ++
--	1 eksiltme operatörüdür	a --

### 2. ilişkisel operatörler

Operatör	İşlevi	Örnek
==	Soldaki ifadenin sağdaki ifadeye eşit olup olmadığını kontrol	a == b
!=	Soldaki ifadenin sağdaki ifadeden farklı olup olmadığını kontrol	a != b
>	Soldaki ifadenin sağdaki ifadeden büyük olup olmadığını kontrol	a > b
>=	Soldaki ifadenin sağdaki ifadeden büyük/eşit olup olmadığını kontrol	a >= b
<	Soldaki ifadenin sağdaki ifadeden küçük olup olmadığını kontrol	a < b
<=	Soldaki ifadenin sağdaki ifadedenküçük/eşit olup olmadığını kontrol	a <= b

### 3. Mantıksal Operatörler

Operatör	İşlevi	Örnek
&& (VE)	Bütün koşulların doğru olması gerekir	( A && B ) yanlış
(VEYA)	Koşullardan en az birisinin doğru olması gerekir	( A    B ) doğrudur
! (DEĞİL)	Doğrunun değil, yanlışın değil doğrudur	! ( A && B ) doğru

```
public static void main(String[] args)
{
    int A = 2 , B = 3 , C = 4;
    System.out.println ( ! ( A > B ) && A > 3 );           // 1.şart doğru, 2.şart yanlış    False
    System.out.println ( A > C || B <= A );                 // 1.şart yanlış, 2.şart yanlış    False
    System.out.println ( B >= A && A != C );                 // 1.şart doğru, 2.şart doğru      True
    System.out.println ( A >= B && ! ( C >= B ) );           // 1.şart yanlış, 2.şart yanlış    False
    System.out.println ( B > 3 || A != C );                 // 1.şart yanlış, 2.şart doğru     True
}
```

## Java Veri Türleri:

Java dilinde en fazla kullanılan tamsayı türü int iken, gerçel sayı türü ise **double** 'dır. Çünkü **double** türünün yuvarlama hatalarına direnci oldukça yüksektir. Çift tırnak içindeki karakterler **string** türündeyken, Tek tırnak içerisindeki karakter **char** türündendir.

	Tür	Sınır	Bellek	Tanımlama
<b>Tamsayılar</b>	byte	-128..+127	8 Bit	byte deger=100
	short	-32768..+32767	16 Bit	short deger =-23565
	int	-2.147.483.648.. +2.147.483.647	32 Bit	int deger=5000000
	long	-9.223.372.036.854.775.808 .. +9.223.372.036.854.775.807	64 Bit	long deger=-10000000
<b>Gerçel Sayılar</b>	float	$\pm 1.5 \times 10^{-45} \dots \pm 3.4 \times 10^{38}$	32 Bit	float deger=0.18
	double	$\pm 5.0 \times 10^{-324} \dots \pm 1.7 \times 10^{308}$	64 Bit	double deger=3.14
<b>Mantıksal</b>	boolean	true & false	1 Bit	boolean deger=true
<b>Karakter</b>	char	Unicode karakter	16 Bit	char deger='A' char deger=65
<b>Metin</b>	string	String Sınıfı	---	String str = "Karakter dizisi"

### Değişken Tanımlama

```
int vizenotu;
```

```
double kdv;
```

```
char cinsiyet;
```

```
String adres;
```

### Değişkene Değer Atama

```
vizenotu = 80;
```

```
kdv = 0.18;
```

```
cinsiyet='E';
```

```
adres = "Hatay";
```

### Değişken Tanımlama ve Değer Atama

```
int vizenotu = 80;
```

```
double kdv = 0.18;
```

```
char cinsiyet='E';
```

```
String adres = "Hatay";
```

### Aynı türden değişkenler Tanımlama:

```
int a,b,c;
```

```
double x,y,z;
```

```
double b;
```

```
string ad, soyad;
```

### Sabit Tanımlama

```
final byte AY_SAYISI=12;
```

```
final float PI = 3.1416f;
```

```
final String DUNYA_UYDU="Ay";
```

```
public static void main(String[] args)
```

```
{
```

```
System.out.println("\nByte Min Değeri:"+Byte.MIN_VALUE);
```

```
System.out.println("Byte Max Değeri:"+Byte.MAX_VALUE);
```

```
}
```

## Tür Dönüşümleri:

Farklı türdeki değişkenlerin aynı ifade içinde işlem görebilmesi için tür dönüşümünün yapılması gerekir. Dolayısıyla atama işleminde önce sağ taraf sol tarafın türüne dönüştürülür, daha sonra atama yapılır.

## String Türünü Sayısal Türlere Dönüştürmek:

**String x="10";**

byte=Byte.parseByte ( x )	byte=Byte.valueOf ( x )
short=Short.parseShort ( x )	short=Short.valueOf ( x )
int=Integer.parseInt ( x )	int=Integer.valueOf ( x )
long=Long.parseLong ( x )	long=Long.valueOf ( x )
float=Float.parseFloat ( x )	float=Float.valueOf ( x )
double=Double.parseDouble( x )	double=Double.valueOf ( x )

## Sayısal Türleri String Türüne Dönüştürmek:

**tür x=10;**

String=Byte.toString ( byte x )	String= String.valueOf ( byte x )
String=Short.toString ( short x )	String= String.valueOf ( short x )
String=Integer.toString ( integer x )	String= String.valueOf ( integer x )
String=Long.toString ( long x )	String= String.valueOf ( long x )
String=Float.toString ( float x )	String= String.valueOf ( float x )
String=Double.toString ( double x )	String= String.valueOf ( double x )

String x="10", y="20"; int a=Integer.parseInt(x) + Integer.parseInt(y); double b=Double.parseDouble(x) + Double.parseDouble(y); int c=Integer.valueOf(x) + Integer.valueOf(y); double d=Double.valueOf(x) + Double.valueOf(y); System.out.println(a); System.out.println(b); System.out.println(c); System.out.println(d);	int x=10, y=20;  String a=Integer.toString(x);  String b=String.valueOf(y);  String c = a + b;  System.out.println(c);
----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	------------------------------------------------------------------------------------------------------------------------------------------------



### Otomatik Dönüşümler:

Java dilinde düşük kapasiteli türler yüksek kapasiteli türlere otomatik olarak dönüştürülebilmektedir. Bu tür dönüşümlerde büyük türlerden dolayı eklenen bitler sıfırla beslenmektedir. Dolayısıyla küçük türdeki yüksek anlamlı bitlerin sıfırla beslenmesi değişkendeki değeri değiştirmede için herhangi bir veri kaybı olmamaktadır.

Ancak büyük kapasiteli türlerin küçük kapasiteli türlere otomatik dönüştürülmesi yasaklanmıştır. Çünkü bu tür dönüşümlere izin verilmesi halinde birtakım veri kayıplarına yol açabileceği bilinmelidir.

byte	➔	short, int, long, float, double
short	➔	int, long, float, double
int	➔	long, float, double
long	➔	float, double
float	➔	double

### Otomatik dönüşümlerde dikkat edilmesi gereken hususlar;

- ❖ Tamsayı türlerinden gerçel sayı türlerine doğrudan atama yapılabilirken, gerçel sayı türlerinden tamsayı türlerine atama yapılamaz.
- ❖ Küçük işaretli sayı türlerinden büyük işaretli sayı türlerine atama yapılabilirken, küçük işaretli sayı türlerinden büyük işaretli sayı türlerine atama yapılamaz.
- ❖ Bölme işlemlerinde kullanılan değişken türlerinin gerçel seçilmesi gerekir.

<pre>int i=25000; long L; L=i;</pre>	<pre>long i=25000;           // Hatalıdır, çünkü long veri tipi int L;                  // int veri tipinden büyüktür. L=i;</pre>
--------------------------------------	-----------------------------------------------------------------------------------------------------------------------------------

### Programcı Dönüşümleri:

Programcı dönüşümü genellikle derleyicinin izin vermediği dönüşümlerde kullanılır. Bu tür dönüşümlerde küçük türler büyük türlere ya da büyük türler küçük türlere dönüştürülebilir. Küçük türlerin büyük türlere dönüştürülmesi aynen otomatik dönüşümde olduğu gibiyken, büyük türlerin küçük türlere dönüşümü ise tür dönüştürme operatörüyle olabilmektedir.

<pre>long uzun=1500; int kısa= (int) uzun; System.out.println(kısa);</pre>	<pre>int s1 = 10 , s2=20; double s3 = (double) s1/s2; System.out.println(s3);</pre>
----------------------------------------------------------------------------	-------------------------------------------------------------------------------------

Ancak büyük türlerden küçük türlere dönüşümde değerin sığmaması halinde veri kaybının olacağı unutulmamalıdır.

<pre>int a=300; byte b=(byte) a;</pre>	Programı çalıştırdığımızda ekrana 44 yazacaktır.
----------------------------------------	--------------------------------------------------

**Program yazarken en sık yapılan hatalardan bazıları şunlardır;**

- Aynı satırda farklı türden değişkenler tanımlamak,

```
int a , string b;
```

- Değişkene uygunsuz değer vermeye çalışmak,

```
int a;  
a="metin";
```

- Değişkeni tanımlamadan ve/veya değişkene ilk değer vermeden değişkeni kullanmaya çalışmak,

```
int b;  
System.out.println(a);  
System.out.println(b);
```

- Değişkenlere yanlış şekilde değer vermek,

```
String a="deneme";  
// String a=deneme;  
System.out.println(a);
```

- Bazı değişken türlerindeki değişkenlere değer verirken eklenmesi gereken karakteri eklememek,

```
float a=12.5f;  
// float a=12.5;  
System.out.println(a);
```

- Ondalık sayıların ondalık kısmını ayırırken nokta (.) yerine virgül (,) kullanmak,

```
float a=12.5f;  
// float a=12,5f;  
System.out.println(a);
```

- Metinsel değişkenlerle matematiksel işlem yapmak,

```
String a="1" , b="2";  
// int a=1 , b=2;  
int c=a+b;  
System.out.println(a);
```

- Bir değişkeni birden fazla kez tanımlamak,

```
String a;  
// String a="deneme";  
System.out.println(a);
```

## Java Çıktı İşlemleri:

### 1. print / println Metotları:

```
// print1.java
public static void main(String[] args)
{
    System.out.println("Merhaba");
    System.out.print("Meslek ");
    System.out.print("Yüksek ");
    System.out.println("Okulu");
    System.out.println();
    System.out.println();
    System.out.print("Java Programlama ");
    System.out.println("Öğreniyorum");
}
```

```
// print2.java
public static void main(String[] args)
{
    System.out.println("Türkiye\ 'yi seviyorum");
    System.out.println("Herşey \" JAVA \" için...");
    System.out.println("c:\\Depo\\okul.doc");
    System.out.println("Numara \t ad \t soyad");
    System.out.println("MKÜ \n Antakya MYO");
}
```

```
// print3.java
public static void main(String[] args)
{
    int A=20, B=10, C=2, D=3;
    System.out.println ( A + B / C + D );
    System.out.println (A + ( B / C ) + D );
    System.out.println (( A + B ) / C + D );
    System.out.println ( A + B / ( C + D ));
    System.out.println (( A + B ) / ( C + D ));
}
```

### 2. printf Metodu:

Format	Anlamı
%d & %5d	Tamsayıları sağa hizalı görüntüler.
%f & %-5.2f	Gerçek sayıları sola hizalı görüntüler.
%o & %5o	İşaretsiz tamsayıları octal görüntüler.
%x & %5x	İşaretsiz tamsayıları hexa görüntüler.
%e & %5.2e	Gerçek sayıları üstel biçimde görüntüler.
%c & %5c	Karakterleri sağa hizalı görüntüler.
%s & %-5s	Metinleri sola hizalı görüntüler.
%b & %5b	Boolean ifadeleri görüntüler.

```
// printf1.java
public static void main(String[] args)
{
    int S=3, A=9, N=5;
    System.out.println(S + A + N);
    System.out.println(S + "" + A + "" + N);
    System.out.println(S + " " + A + " " + N);
    System.out.printf("%d%d%d\n", S, A, N);
    System.out.printf("%d %d %d\n", S, A, N);
    System.out.printf("%5d %5d %5d\n", S, A, N);
    System.out.printf("%-5d %-5d %-5d\n", S, A, N);
}
```

```
// printf2.java
public static void main(String[] args)
{
    int a=300, b=200;
    System.out.println(a + " + " + b + " = " + a+b);
    System.out.println(a + " + " + b + " = " + (a+b));
    System.out.println(a + " - " + b + " = " + (a-b));
    System.out.println(a + " * " + b + " = " + a*b);
    System.out.println(a + " % " + b + " = " + a%b);
    System.out.println(a + " / " + b + " = " + a/b);
    System.out.println("\n-----\n");
    System.out.printf("%3d + %3d = %3d\n", a, b, a+b);
    System.out.printf("%3d - %3d = %3d\n", a, b, a-b);
    System.out.printf("%3d * %3d = %3d\n", a, b, a*b);
    System.out.printf("%3d % %3d = %3d\n", a, b, a%b);
    System.out.printf("%3d / %3d=%3.1f\n", a, b, (double)a / b);
}
```

```
// printf3.java
public static void main(String[] args)
{
    int dtar , kilo ; float boy ;
    String ad , soyad ;
    ad = "Mustafa";
    soyad ="Korkmaz";
    dtar= 1978 ;
    boy = 1.75f ;
    kilo=80 ;
    System.out.println("Sayın " +ad + " " + soyad) ;
    System.out.printf("Sayın %s %s" , ad , soyad) ;
    System.out.printf( "\n-----" );
    System.out.printf("\nYaşınız:%d" , 2020 - dtar );
    System.out.printf( "\nBoyunuz:%5.2f" , boy) ;
    System.out.printf( "\nKilonuz:%d" , kilo) ;
}
```

```
// printf4.java
public static void main(String[] args)
{
    int a1=1, a2=2;
    System.out.printf("\nsayı1 ==>%d sayı2 ==>%d",a1,a2);
    System.out.print("\n-----");
    System.out.printf("\n%d == %d ==> %b" , a1 , a2 , a1==a2);
    System.out.printf("\n%d != %d ==> %b" , a1 , a2 , a1!=a2);
    System.out.printf("\n%d > %d ==> %b" , a1 , a2 , a1>a2);
    System.out.printf("\n%d < %d ==> %b" , a1 , a2 , a1<a2);
    System.out.printf("\n%d >= %d ==> %b" , a1 , a2 , a1>=a2);
    System.out.printf("\n%d <= %d ==> %b" , a1 , a2 , a1<=a2);

    System.out.print("\n.....");

    boolean b1=true, b2=false;
    System.out.printf("\nb1 ==> %b b2 ==> %b" , b1 , b2);
    System.out.print("\n-----");
    System.out.printf("\n!b1 ==>%b" , !b1);
    System.out.printf("\n!b2 ==>%b" , !b2);
    System.out.printf("\nb1 && b2 ==>%b" , b1&&b2);
    System.out.printf("\nb1 || b2 ==>%b" , b1||b2);
}
```

## Java Girdi İşlemleri:

Java dilinde klavyeden değer girebilmek için projemize scanner sınıfı eklenmelidir. Scanner sınıfı java.util kütüphanesinin içerisinde bulunmaktadır. Import anahtar kelimesini kullanarak uygulamamıza farklı kütüphaneler ve sınıflar dahil edebiliriz.

```
import java.util.*;           // Java Util kütüphanesini ekler.  
import java.util.Scanner;    // Java Util kütüphanesinin Scanner class.  
import java.util.List;       // Java Util kütüphanesinin List class  
import java.util.ArrayList;  // Java Util kütüphanesinin ArrayList class  
import java.util.Date;       // Java Util kütüphanesinin Date class.  
import java.swing.*;         // Java swing kütüphanesini ekler.  
import javax.swing.JOptionPane; //Java swing kütüphanesinin JOptionPane class
```

Java programlarında klavyeden veri girişi için "java.util" paketinde yer alan "Scanner" sınıfının metotları kullanılır. Bu yüzden programın başına "import java.util.Scanner" ile sınıf çağırılır. "Scanner" sınıfının önemli metotları ise aşağıdaki gibidir.

<b>nextByte()</b>	Girilen ifadeyi byte tipinde alır.
<b>nextShort()</b>	Girilen ifadeyi short tipinde alır.
<b>nextInt()</b>	Girilen ifadeyi int tipinde alır.
<b>nextLong()</b>	Girilen ifadeyi long tipinde alır.
<b>nextFloat()</b>	Girilen ifadeyi float tipinde alır.
<b>nextDouble()</b>	Girilen ifadeyi double tipinde alır.
<b>next()</b>	Girilen ifadeyi ilk boşluğa kadar alır.
<b>nextLine()</b>	Girilen tüm satırı alır.
<b>charAt(indis)</b>	Girilen ifadenin belirtilen indisteki karakteri alır.
<b>nextBoolean()</b>	Girilen ifadeyi boolean tipinde alır.

```
// KDV dahil ürün fiyatını hesaplar.
public static void main(String[] args)
{
    Scanner klavye=new Scanner(System.in);

    System.out.print("Ürün Kodunu Gir:");
    String urKod=klavye.nextLine();

    // int urKod=klavye.nextInt(); klavye.nextLine();

    System.out.print("Ürün Adını Gir:");
    String urAd=klavye.nextLine();
    System.out.print("Ürün Fiyatını Gir:");
    double urFiyat=klavye.nextDouble();

    urFiyat = urFiyat * 1.18;          // urFiyat*=1.18;

    System.out.printf("\n%s isimli ürünün", urAd);
    System.out.println("\n-----");
    System.out.printf("KDV Dahil Fiyat:%5.1f TL",urFiyat);
}

```

```
// Dairenin alanını hesaplar.
public static void main(String[] args)
{
    Scanner scan = new Scanner(System.in);

    final double PI = 3.1416;

    System.out.print("Dairenin Yarıçapını Gir: ");
    if (scan.hasNextDouble())
    {
        double yaricap=scan.nextDouble();

        double alan = PI * yaricap * yaricap;

        System.out.printf("\nDairenin Alanı:%5.2f", alan);
    }
    else
        System.out.print("\nSayınızı kontrol ediniz...");
}

```

```
// İki sayıyı toplar.
public static void main(String[] args)
{
    Scanner input=new Scanner(System.in);

    try          // İşlem yapılan blok
    {
        System.out.print("Birinci Sayıyı Gir:");
        int a=input.nextInt();
        System.out.print("İkinci Sayıyı Gir:");
        int b=input.nextInt();
        System.out.print(a+" + "+b+" = " +(a+b));
        // System.out.printf("\n%d + %d=%d\n", a, b, a+b);
    }
    catch (Exception hata) //Hata yakalayan blok
    {
        System.out.println("Sadece sayı girilebilir.");
        System.out.println("Sistem Hata Kodu :"+hata);
    }
    finally      // Her durumda çalışan blok
    {
        System.out.print("\nİşlem sonlandırıldı...");
    }
}

```

```
// Sıcaklık ortalamasını hesaplar.
public static void main(String[] args)
{
    Scanner klavye=new Scanner(System.in);

    String gunad;
    byte sabah , ogle , aksam;
    float ort ;

    System.out.print(" Bulunduğun günün adı:");
    gunad=klavye.next();

    System.out.print(" Sabah Sıcaklık Derecesi:");
    sabah=klavye.nextByte();

    System.out.print(" Sabah Sıcaklık Derecesi:");
    ogle=klavye.nextByte();

    System.out.print(" Sabah Sıcaklık Derecesi:");
    aksam=klavye.nextByte();

    ort = ( sabah + ogle + aksam ) / 3 ;

    System.out.printf (" %s gününün ",gunad);
    System.out.printf ("sıcaklık ortalama:%3.1f", ort);
}

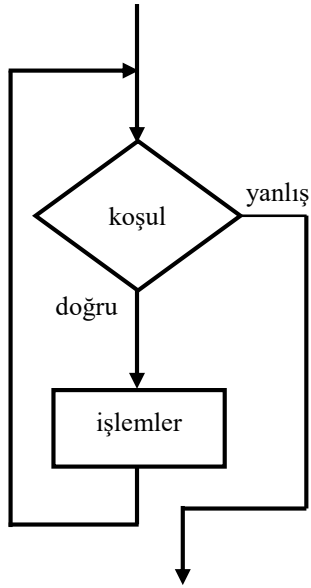
```

## Döngü Deyimleri:

Döngüler program içerisinde işlemleri defalarca tekrar edebilmemizi sağlayan komutlardır.

### while döngüler:

**while** koşulunun değeri **true** ise döngü sürekli olarak çalıştırılırken, **false** olduğu takdirde döngü sonlandırılarak program akışı döngü sonrasındaki ilk komutla devam eder.



```
while (koşul)
{
    işlem;
}

while (koşul)
{
    işlemler;
}

// sonsuz döngü
while (true)
{
    işlemler;
}
```

// 1 den 10 a kadar olan sayıları ekrana listeler.

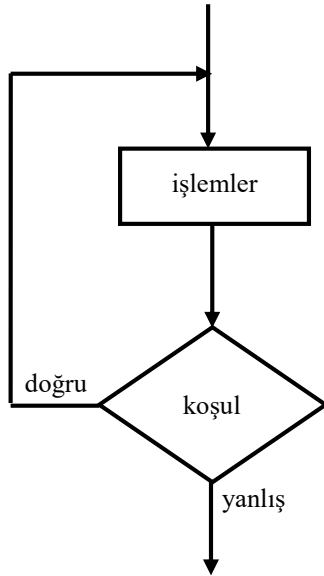
```
public static void main(String[] args)
{
    int sayac=0; //sayac=11;
    String sonuc="";
    while (sayac <10) // (sayac>1)
    {
        sayac++; // sayac--;
        System.out.print(sayac+" ");

        // System.out.println(sayac);

        // sonuc+=Integer.toString(sayac) + " ";
    }
    // System.out.println(sonuc);
}
```

## do-while döngüleri:

Bu döngünün while döngüsünden tek farkı koşulun döngünün sonunda kontrol ediliyor olmasıdır. Bu durum koşuldan bağımsız olarak döngünün en az bir kez çalışması demektir.



```
do
{
işlemler;
}
while (koşul);
```

```
// 1 den 10 a kadar olan sayıları ekrana listeler.
public static void main(String[] args)
{
    int sayac=0;                //sayac=11;
    String sonuc="";
    do
    {
        sayac++;                // sayac--;
        System.out.print(sayac+" ");

        //System.out.println(sayac);

        //sonuc+=Integer.toString(sayac)+" ";
    }
    while(sayac <10);            // sayac>1;
    // System.out.println(sonuc);
}
```

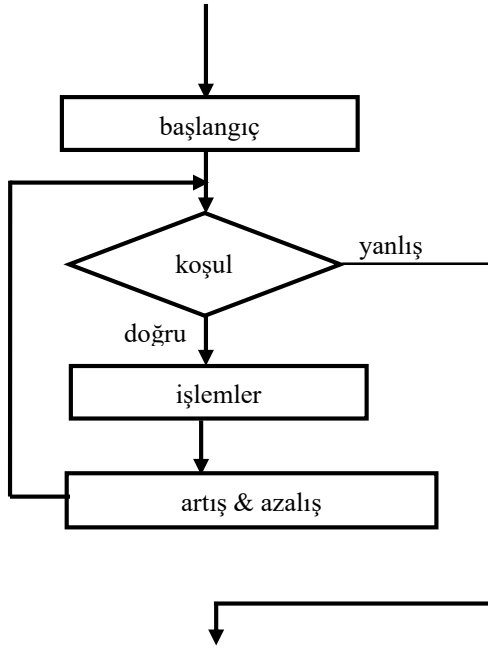


## for döngüleri:

for döngüleri daha çok tekrar sayısının belli olduğu işlemlerde tercih edilir.

### for döngüsü şöyle çalışır;

- ❖ Birinci kısımdaki parametre sadece döngüye ilk girişte çalışır.
- ❖ İkinci kısımdaki parametre true olduğu sürece döngü tekrar eder.
- ❖ Üçüncü kısımdaki parametre her işlemten sonra başa dönerek değeri artırır & eksiltir.
- ❖ Birinci ve üçüncü kısımdaki parametreler string haricinde herhangi bir türden olabilirken, ikinci kısımdaki parametre bool türünden olmalıdır.



### Artan Döngüler

```
for (başlangıç; koşul; artış)
{
    işlemler;
}
```

### Azalan Döngüler

```
for (başlangıç; koşul; azalış)
{
    işlemler;
}
```

// 1 den 10 a kadar olan sayıları ekrana listeler

```
public static void main(String[] args)
{
    String sonuc="";
    for (int sayac=1;sayac<=10;sayac++)
        System.out.print(sayac+" ");
    // System.out.println(sayac);
    // sonuc+=Integer.toString(sayac)+" ";
    // sonuc+="\n";
    for (int sayac=10;sayac>=1;sayac--)
        System.out.print(sayac+" ");
    // System.out.println(sayac);
    // sonuc+=Integer.toString(sayac)+" ";
    // System.out.println(sonuc);
}
```

Döngüleri kontrol etmek için **break** ve **continue** komutları kullanılmaktadır;

#### **break komutu:**

```
// 1..10 arası bütün sayılar

public static void main(String[] args)
{
    int x = 2;

    for (;;)           // sonsuz döngü
    {
        System.out.println(x);

        if (x == 10) break;

        x+=2;
    }
}
```

#### **continue komutu:**

```
// 1..10 arası tek sayılar

public static void main(String[] args)
{
    for (int i=1;i<=10;++i)
    {
        if (i % 2== 1)
        {
            System.out.println("-----> Tek sayı engellendi.");

            continue;
        }

        System.out.println(i);
    }
}
```

**// Klavyeden girilen n adet sayı toplamını bulur.**

```
public static void main(String[] args)
{
    int n, sayi, toplam = 0, saydir=0;
    Scanner klavye = new Scanner(System.in);
    System.out.print("\nSayı Adedini Gir: ");
    n = klavye.nextInt();
    System.out.print("-----\n");
    do
    {
        saydir++;
        System.out.printf("%d. Sayı: ",saydir);
        sayi = klavye.nextInt();
        toplam +=sayi;
    }
    while(saydir <n);
    System.out.println("\n\nToplam: "+toplam);
}
```

**// Fibonacci, her sayının kendisinden bir önceki sayı ile toplanması ile elde edilen sayı serisidir.**

```
public static void main(String[] args)
{
    Scanner scan=new Scanner(System.in);
    System.out.print("Fibonacci sınırını giriniz: ");
    int adet = scan.nextInt();
    long a = 0 , b = 1;
    System.out.printf("%d %d", a, b);
    for (int i = 3; i <= adet; i++)
    {
        long c = a + b;
        System.out.printf(" %d ",c);
        a = b;
        b = c;
    }
}
```

**// Klavyeden girilen sayının faktöriyelini hesaplar.**

```
public static void main(String[] args)
{
    byte son, say = 0;
    long faktor = 1;
    Scanner scan=new Scanner(System.in);
    System.out.print("Üst sınırı gir:");
    son=scan.nextByte();
    while (say < son)
    {
        say++;
        faktor = faktor * say;
        System.out.printf("\n%5d! = %5d", say, faktor);
    }
}
```

**// Üçgen görünümlü ekran çıktısı verir.**

```
public static void main(String[] args)
{
    for(int a=1; a<=10; a++)
    {
        System.out.printf("%2d",a);
        for(int b=1; b<=a; b++)
        {
            System.out.print(" X");
        }
        System.out.println();
    }
    System.out.println("\n-----\n");
    for(int a=10; a>=1; a--)
    {
        System.out.printf("%-2d",a);
        for(int b=1; b<=a; b++)
        {
            System.out.print(" x");
        }
        System.out.println();
    }
}
```

**// 10 sayısını bulana kadar çalışmaya devam eder.**

```
public static void main(String[] args)
{
    Random r = new Random();
    int uret=0, saydir=0;
    while (uret!=10)
    {
        saydir++;
        uret = r.nextInt(10)+1;
        System.out.println(uret);
    }
    System.out.println(saydir+".tekrarda program sonlandı");
}
```

**// Kullanıcının tuttuğu sayıyı bilgisayar tahmin eder.**

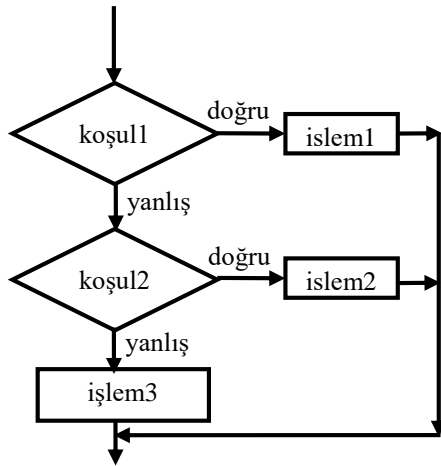
```
public static void main(String[] args)
{
    int uret, sayac = 0;
    Random r = new Random();
    Scanner klavye=new Scanner(System.in);
    System.out.print("Bir sayı tut(1-10 arası :)");
    int tut = klavye.nextInt();
    System.out.print("-----\n");
    do
    {
        sayac++;
        uret = r.nextInt(11);
        System.out.println(uret);
    }
    while (tut != uret);
    System.out.print("-----\n");
    System.out.print(tut + " sayısı " + sayac + ".tahminde bulundu");
}
```

## Kontrol Yapıları:

Program yazarken en çok karşılaşılan durumların başında elinizdeki verilere bakarak bir karar vermek ve programın akışını bu karara göre yönlendirmek gelir. İyi bir programda gerekli yerlerde doğru ve etkili karar yapıları oluşturmak çok önemlidir.

### 1. switch – case yapıları:

Birden fazla şartlı işlemlerde kullanılan bir yapı genellikle kompleks if-else bloklarının yerine tercih edilir. Switch deyimi ile yapılabilen bütün kontroller if deyimi ile de yapılabilir. Ancak bazı durumlarda switch deyimi daha sade bir yapıda görünür.



```
switch (koşul )
{
    case sabit1: işlem_1; break;
    case sabit2: işlem_2; break;
    case sabit3: işlem_3; break;
    .
    case sabit_n: işlem_n; break;
    default: işlem_m; break;
}
```

Derleyici switch parantezi içerisindeki koşulun eşdeğeri olan bir case bölümü araştırır. Bulursa akışı ilgili case bölümüne atlar. Akış o noktadan komutları çalıştırarak devam eder, break anahtar sözcüğü görüldüğünde akış switch dışındaki ilk komutla devam eder. Eğer switch parantezi içerisindeki koşula tam eşit bir case bölümü bulunamazsa ve switch deyiminin default bölümü varsa akış default bölümüne aktarılır. Eğer default bölümü de yoksa akış switch dışındaki ilk komutla devam eder.

Switch-case yapısını kullanırken dikkat edilmesi gereken hususlardan bazıları şunlardır;

- ✓ switch yapısı { paranteziyle başlar } paranteziyle de sona ermeli,
- ✓ switch koşulu parantez içinde verilmeli,
- ✓ switch değişkeni, gerçel tipler haricindeki tiplerden birisi (int, char,string) olmalı,
- ✓ case blokları koşul ile aynı tipte sabit değer olmalı, ifade veya değişken olmamalı,
- ✓ switch koşuluyla case sabitleri arasında sadece eşitlik kontrolü yapılabilir,
- ✓ case sabitleri birden fazla değerden oluşmamalı, tek değer kullanılmalı,
- ✓ **case** ve **default** blokları mutlaka **break;** komutuyla sonlandırılmalı,
- ✓ case bloklarının { } parantezleri arasına alınması zorunlu değildir.
- ✓ Case bloklarının sıralı olması ya da default bloğunun en sonda olması zorunlu değildir.

**// Girilen rakama karşılık gelen gün ismini verir.**

```
public static void main(String[] args)
{
    Scanner input=new Scanner(System.in);
    System.out.print("Gün numarasını gir:");
    int gun=input.nextInt();
    switch (gun)
    {
        case 1: System.out.print("Bugün: PAZARTESİ"); break;
        case 2: System.out.print("Bugün: SALI"); break;
        case 3: System.out.print("Bugün: ÇARŞAMBA"); break;
        case 4: System.out.print("Bugün: PERŞEMBE"); break;
        case 5: System.out.print("Bugün: CUMA"); break;
        case 6: System.out.print("Bugün: CUMARTESİ"); break;
        case 7: System.out.print("Bugün: PAZAR"); break;
        default: System.out.print("HATA"); break;
    }
}
```

**// Girilen rakama karşılık hafta içi / hafta sonu durumunu bulur.**

```
public static void main(String[] args)
{
    Scanner input=new Scanner(System.in);
    System.out.print("Gün numarasını gir:");
    int gun=input.nextInt();
    switch (gun)
    {
        case 1:
        case 2:
        case 3:
        case 4:
        case 5: System.out.print("Hafta İçi "); break;
        case 6:
        case 7: System.out.print("Hafta Sonu"); break;
        default: System.out.print("HATA"); break;
    }
}
```

```

// Dikdörtgenin alanını veya çevresini hesaplar

public static void main(String[] args)
{
    Scanner klavye=new Scanner(System.in);

    System.out.println("\n***DİKDÖRTGENİN***");
    System.out.println("-----");
    System.out.print("Kısa kenarını gir:");
    int kk = klavye.nextInt();
    System.out.print("\nUzun kenarını gir:");
    int uk = klavye.nextInt();

    System.out.println("\n\n***ANA MENÜ***");
    System.out.println("\n 1. Alan");
    System.out.println("\n 2. Çevre");
    System.out.println("\n 3. Çıkış");
    System.out.print("\n Seçiminiz:");
    int sec = klavye.nextInt();

    switch(sec)
    {
        case 1: int alan = kk * uk;
                System.out.printf("\nAlan=%d", alan);
                break;
        case 2: int cevre = 2 * kk + 2 * uk;
                System.out.printf("\nÇevre=%d", cevre);
                break;
        case 3: System.out.println("\nÇıktınız...");
                return;
        default: System.out.println("\nGeçersiz");
                break;
    }
}

```



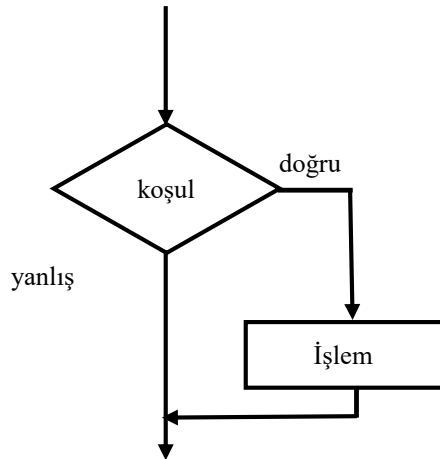
**// 1..99 arası rakamların okunuş karşılığını bulur**

```
public static void main(String[] args)
{
    int sayi; String birler="",onlar="";
    Scanner input= new Scanner(System.in);
    do
    {
        System.out.print("Bir Sayı[1..99] :");
        sayi = input.nextInt();
    } while (!(sayi>0 && sayi<100));

    System.out.print(sayi+" Sayısı ");
    switch (sayi / 10)
    {
        case 1: onlar="On"; break;
        case 2: onlar="Yirmi"; break;
        case 3: onlar="Otuz"; break;
        case 4: onlar="Kırk"; break;
        case 5: onlar="Elli"; break;
        case 6: onlar="Altmış"; break;
        case 7: onlar="Yetmiş"; break;
        case 8: onlar="Seksen"; break;
        case 9: onlar="Doksan"; break;
    }
    switch ((sayi % 10))
    {
        case 1: birler="Bir"; break;
        case 2: birler="İki"; break;
        case 3: birler="Üç"; break;
        case 4: birler="Dört"; break;
        case 5: birler="Beş"; break;
        case 6: birler="Altı"; break;
        case 7: birler="Yedi"; break;
        case 8: birler="Sekiz"; break;
        case 9: birler="Dokuz"; break;
    }
    System.out.printf("\n%d sayısı %s %s şeklinde okunur..." , sayi , onlar , birler);
}
```

## 2. if Kontrol Yapısı:

### a. Basit if Yapısı:



```
if (koşul)
    işlem; //doğruysa
```

```
if (koşul)
{
    işlemler; //doğruysa
}
```

```
public static void main(String[] args)
{
    int sayi;

    Scanner klavye=new Scanner(System.in);

    System.out.print("Lütfen bir sayı gir: ");

    sayi=klavye.nextInt();

    if (sayi==0)

        System.out.printf("%d sayısı sıfır", sayi);

    if (sayi<0)

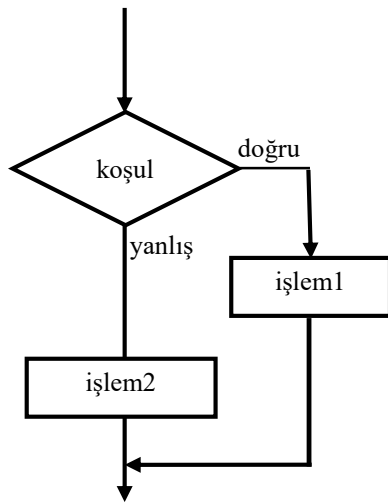
        System.out.printf("%d sayısı negatif", sayi);

    if (sayi>0)

        System.out.printf("%d sayısı pozitif", sayi);

}
```

## b. if else Yapısı:



```
if (koşul)

işlem1; //doğruysa

else

işlem2; //yanlışsa
```

```
if (koşul)

{

İşlemler1 ; //doğruysa

}

else

{
```

```
public static void main(String[] args)

{

int sayi;

Scanner klavye=new Scanner(System.in);

System.out.print("Lütfen bir sayı gir: ");

sayi=klavye.nextInt();

if (sayi<0)

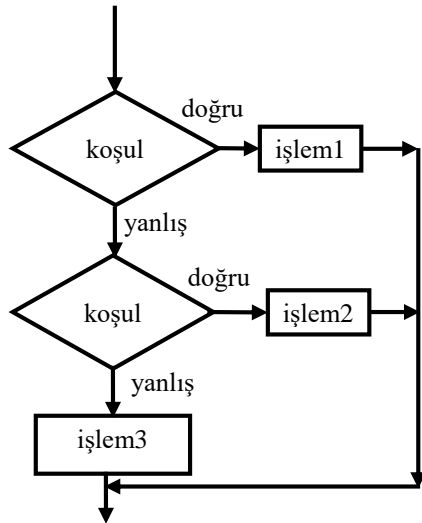
System.out.printf("%d sayısı negatif",sayi);

else

System.out.printf("%d sayısı pozitif",sayi);

}
```

### c. if else if else Yapısı:



```
if (koşul1)
```

```
işlem1;
```

```
else if (koşul2)
```

```
işlem2;
```

```
else işlem3;
```

```
if (koşul1)
```

```
{
```

```
işlemler1;
```

```
}
```

```
else if (koşul2)
```

```
{
```

```
işlemler2;
```

```
}
```

```
else
```

```
{
```

```
işlemler3;
```

```
}
```

```
public static void main(String[] args)

{

    int sayi;

    Scanner klavye=new Scanner(System.in);

    System.out.print("Lütfen bir sayı gir: ");

    sayi=klavye.nextInt();

    if (sayi==0)

        System.out.printf("%d sayısı sıfır",sayi);

    else if (sayi<0)

        System.out.printf("%d sayısı negatif",sayi);

    else

        System.out.printf("%d sayısı pozitif",sayi);

}
```

```
// Geçme Kalma Durumunu kontrol eder

public static void main(String args[])

{

    do

    {

        System.out.print("\nNotunuzu Giriniz:");

        Scanner klavye = new Scanner(System.in);

        int not= klavye.nextInt();

        if (not < 60)

            System.out.println("\nGeçme Notu 60 KALDINIZ");

        else

            System.out.println("\nGeçme Notu 60 GEÇTİNİZ");

        System.out.print("\nTekrar Edilsinmi?[E/H]?");

        char cevap=klavye.next().charAt(0);

        if (cevap=='E' || cevap=='e') continue;

        else break;

    }

    while (true);

}
```

```
// Girilen sayıları tek ve çift şeklinde ayırarak toplar

public static void main(String[] args)

{

    int sayi, tekTop=0, ciftTop=0;

    byte tekSay=0, ciftSay=0, n;

    Scanner klavye=new Scanner(System.in);

    System.out.print("Sayı adedi gir:");

    n = klavye.nextByte();

    for (byte i = 1; i <= n; i++)

    {

        System.out.print(i+" .Sayıyı gir:");

        sayi=klavye.nextByte();

        if (sayi % 2 == 1)

        {

            tekSay++;

            tekTop +=sayi;

        }

        else

        {

            ciftSay++;

            ciftTop += sayi;

        }

    }

    System.out.println("\n-----");

    System.out.printf("\n%d tek sayı top:%d", tekSay, tekTop);

    System.out.printf("\n%d çift sayı top:%d", ciftSay, ciftTop);

}
```

**// Rastgele üretilen iki sayının toplamını tek-çift durumu**

```
public static void main(String args[])
{
    int r1 = (int)(Math.random() * 101);
    int r2 = (int)(Math.random() * 101);
    System.out.printf("Random1:%d\nRandom2:%d\n",r1,r2);
    int toplam = r1 + r2;
    if (toplam % 2 == 0) System.out.printf("Toplam :%d Çift Sayı", toplam);
    else System.out.printf("Toplam :%d Tek Sayı", toplam);
}
```

**// Üçgenin çeşidini bulur**

```
public static void main(String[] args)
{
    int a = 0, b = 0, c = 0;
    Scanner konsol=new Scanner(System.in);
    System.out.print("a kenarını gir:");
    a = konsol.nextInt();
    System.out.print("b kenarını gir:");
    b = konsol.nextInt();
    System.out.print("c kenarını gir:");
    c = konsol.nextInt();
    if (a == b && b == c) System.out.println("eş kenar üçgen");
    else if (a == b || a == c || b==c) System.out.println("ikiz kenar üçgen");
    else System.out.println("çeşit kenar üçgen");
}
```

**// Girilen sayılardan en büyüğünü ve en küçüğünü bulur**

```
public static void main(String args[])

{

    int n, sayi, enb=0, enk=0, sayac=0;

    Scanner klavye=new Scanner(System.in);

    System.out.print("Sayı adedini gir:");

    n = klavye.nextInt();

    while(sayac<n)

    {

        sayac++;

        System.out.print(sayac+" .sayıyı gir:");

        sayi = klavye.nextInt();

        if (sayac == 1) enb=enk=sayi;

        if (sayi>enb) enb=sayi;

        else if (sayi<enk) enk = sayi;

    }

    System.out.print("\nEn Büyük Sayı:"+enb);

    System.out.print("\nEn Küçük Sayı:" + enk);

}
```



**//Tersten kendisine eşit olan sayılara palindrom sayı denir.**

```
public static void main(String[] args)
```

```
{
```

```
Scanner scan = new Scanner(System.in);
```

```
System.out.print("Sayı Giriniz : ");
```

```
int sayi = scan.nextInt();
```

```
int girilen = sayi;
```

```
int uzunluk = String.valueOf(sayi).length();
```

```
int kalan, palindrom=0;
```

```
for (int i = 0; i<uzunluk; i++)
```

```
{
```

```
kalan = sayi % 10;
```

```
palindrom = palindrom * 10 + kalan;
```

```
sayi = sayi /10;
```

```
}
```

```
if (palindrom == girilen)
```

```
System.out.println(girilen + " Palindrom bir sayıdır.");
```

```
else
```

```
System.out.println(girilen + " Palindrom bir sayı değildir.");
```

```
}
```

```
// Armstrong, sayı değerlerinin küpleri toplamı,kendisine eşit
```

```
// Armstrong Sayılar :1 , 153 , 370 , 371 , 407
```

```
public static void main(String[] args)

{

Scanner scan = new Scanner(System.in);

System.out.print("Bir Sayı Giriniz:");

int sayi = scan.nextInt();

int girilen = sayi;

int kalan = 0;

int amstrong = 0;

int uzunluk = String.valueOf(sayi).length();

for (int i=1; i<=uzunluk; i++)

{

kalan = sayi % 10;

amstrong = amstrong + (int)Math.pow(kalan, uzunluk);

sayi = sayi/10;

}

if (amstrong == girilen)

System.out.print("Armstrong:"+girilen+"="+amstrong);

else

System.out.println("Armstrong sayı değildir");

}
```

```
//sesli, sessiz ve kelime adedini bulur
```

```
public static void main(String[] args)
```

```
{
```

```
Scanner klavye=new Scanner(System.in);
```

```
System.out.print(" Mesajını gir:");
```

```
String mes = klavye.nextLine().toLowerCase();
```

```
int sesli=0, sessiz=0, kelime=1;
```

```
for (int i=0; i<mes.length(); i++)
```

```
{
```

```
if (mes.charAt(i)==' ') kelime++;
```

```
else if (mes.charAt(i)=='a' || mes.charAt(i)=='e' || mes.charAt(i)=='i' || mes.charAt(i)=='i' ||
```

```
mes.charAt(i)=='o' || mes.charAt(i)=='ö' || mes.charAt(i)=='u' || mes.charAt(i)=='ü') sesli++;
```

```
else sessiz++;
```

```
}
```

```
System.out.println(kelime+" kelimedenden oluşuyor");
```

```
System.out.println(sesli+" sesli harf bulundu");
```

```
System.out.println(sessiz+" sessiz harf bulundu");
```

```
}
```

## DİZİLER (İNDİSLİ DEĞİŞKENLER):

Bir değişken aynı anda tek bir bilgiyi tutabilir. Bu kuralın istisnası, değişkenin dizi olarak tanımlanmasıdır. Elemanları aynı türden olan ve bir index yardımıyla erişilebilen veri yapılarına dizi(array) denilmektedir.

Program içerisinde aynı türden çok sayıda bilginin olması ve bilgiler üzerinde toplu işlem yapılacağı durumlarda dizilerden yararlanılır. Veriler dizilere yerleştirilirken 0. indisten başlanarak yerleştirilir ve diziden veriler alınırken 0 . indisten başlanarak alınır.

Örnek olarak, 50 kişilik bir sınıftaki öğrencilerin adlarını ve vizelerini almak isteyelim. Bu işlem için değişken kullanacak olursak, 50 adet adlar ve 50 adet vizeler için olmak üzere 100 değişken kullanmamız gerekmektedir. Bu işlemin sınıf için değil de bir okul için yapıldığını düşünün, içinden çıkılamayacak bir hal alacaktır.

**X vektörü tek boyutlu dizi**

$$X = \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ \vdots \\ \vdots \\ X_n \end{bmatrix}$$

**A matrisi iki boyutlu dizi**

$$A = \begin{bmatrix} A_{11} & A_{12} & \dots & A_{1N} \\ A_{21} & A_{22} & \dots & A_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ A_{N1} & A_{N2} & \dots & A_{NN} \end{bmatrix}$$

- **Doğal sayılar(1,2,3, ..) , günler (pzt,sal,çrş...) , aylar (oc,şub,mart,...) dizidir.**
- **Haftanın günleri tek boyutlu bir dizi olarak düşünülebilir.**  
İlk eleman (0. eleman), Pazartesi, son eleman da (6. eleman) Pazar günü olacaktır.
- **Bir ayda bulunan günler iki boyutlu bir dizi olarak düşünülebilir.**  
Yatayda haftanın günleri bulunurken, dikeyde hafta sayısı olabilir.
- **Bir yıldaki günleri de üç boyutlu bir dizi olarak düşünülebilir.**  
1. boyut aylar, 2. boyut hafta numarası, 3. boyut ise haftanın günleri olur.

## Tek Boyutlu Diziler:

Java dilinde tek boyutlu bir dizi, normal bir değişken gibi tanımlanır, sadece değişken ifadesinden sonra köşeli parantez kullanılır. Bir dizi nesnesi **new** operatörü ile yaratıldığında dizinin tüm elemanları sıfırlanır.

Bu tür diziler aşağıdaki gibi tanımlanabilir;

```
String[] ad1=new String[50];
int[] vize1=new int[50];

int n=10;

String[] ad2=new String[n];
int[] vize2=new int[n];

String [] ad3; ad3=new String[50];
int [] vize3;vize3=new int[50];

String [] ad4; ad4=new String[n];
int [] vize4;vize4=new int[n];
```

Diziye ilk değer verilirken **[]** parantezlerinin içi boş bırakılmalıdır. Bu durumda derleyici verilen ilk değerleri sayar ve sanki **[]** parantez içerisine o değer yazılmış gibi işlem yapar.

```
int [] dizi1 = new int [] {1,2,3,4,5}; // Geçerli
String dizi2=new String [] {"pzt","sal","çrş","prş","cum"}; // Geçerli

int [] dizi5 = new int [5] {1,2,3,4,5}; // Geçersiz
String dizi6=new String [5] {"pzt","sal","çrş","prş","cum"}; // Geçersiz

int [] dizi3={1,2,3,4,5}; // Geçerli
String dizi4={"pzt","sal","çrş","prş","cum"}; // Geçerli

int [] dizi7; dizi7={1,2,3,4,5}; // Geçersiz
int [] dizi8; dizi8=new int []{1,2,3,4,5}; // Geçerli
```

Diziye ilk değer verilirken **{}** içerisine ilk değerler sabit olmak zorunda değildir.

```
int x = 5;
int [ ] dizi5=new int [ ] {x, x *2, x *3, x *4}; // Geçerli
```

Dizilerin ekrana listelenmesinde sıkça döngülerin kullanıldığı görülecektir.

```
int [ ] notlar=new int[ ] {50, 40, 70, 10, 25, 90, 55};  
for (int i=0; i<7; i++)  
    System.out.println (notlar[i]);
```

Dizilerin uzunluklarını yani eleman sayılarını bulurken **length** fonksiyonu da kullanılabilir. **Length** readonly olduğundan sadece okunabilir, değer ataması yapılamaz. **i<7** ile **i<gunler.length** eş değerdir.

```
String [] gunler={"Pt", "Sl", "Çrş", "Prş", "Cm", "Ct", "Pz"};  
for (int i=0; i <gunler.length; i++)  
    System.out.println(gunler[i]);
```

for döngüsünün sadece diziler geliştirilmiş bir başka kullanım şeklide aşağıdaki gibidir. Ancak bu yapı giriş işlemlerinde kullanılamaz, sadece çıkış işlemlerinde kullanılabilir.

```
foreach döngüsünün genel biçimi şöyledir:  
for (tür değişken : dizi)  
    değişken;
```

```
int [] notlar=new int[] {50,40,70,10,25,90,55};  
for (int i : notlar)  
    System.out.println (i);
```

```
String [] gunler={"Pt", "Sl", "Çrş", "Prş", "Cm", "Ct", "Pz"};  
for (String i : gunler)  
    System.out.println (i);
```

```
public static void main(String[] args)  
{  
    String [] gunler;  
    gunler=new String[7];  
    Scanner klavye=new Scanner(System.in);  
    for (int i=0; i<7;i++)  
    {  
        System.out.printf("%d. gün:", i+1);  
        gunler[i]=klavye.next();  
    }  
    for (int i=0; i<gunler.length; i++)  
        System.out.println (gunler[i]);  
}
```

```
public static void main(String[] args)  
{  
    Scanner klavye=new Scanner(System.in);  
    System.out.print ("Öğrenci Sayısını Gir:");  
    int n=klavye.nextInt();  
    int[] notlar=new int[n];  
    for (int i=0; i<n; i++)  
    {  
        System.out.printf ("%d. notunu gir:", i+1);  
        notlar[i]=klavye.nextInt();  
    }  
    for (int i: notlar)  
        System.out.println (i);  
}
```

```
// Öğrencilerin vize notlarını ve sınıf ortalamasını bulur
```

```
public static void main(String[] args)

{

Scanner input=new Scanner(System.in);

System.out.println("Sınıf mevcudunu Gir :");

int n=input.nextInt();

int toplam=0;

int [] vizeler=new int[n];

for (int i=0; i<n; i++)

{

System.out.printf("%d. öğrencinin vize notunu gir:",i+1);

vizeler[i]=input.nextInt();

toplam+=vizeler[i];

}

double ort=toplam/n;

System.out.println("\n\nVize Sınıf Ortalaması:"+ort);

System.out.println("-----");

for(int i:vizeler)

    System.out.println(i); }
```

```
// En yüksek vize notunu alan öğrencilerin isimlerini bulur
```

```
public static void main(String[] args)
{
    Scanner input=new Scanner(System.in);

    System.out.print("Sınıf mevcudunu Gir :");

    int n=input.nextInt();

    String [] ad=new String[n];

    int [] vize=new int[n];

    for (int i=0; i<n; i++)
    {
        System.out.printf("\n%d.öğrencinin",i+1);

        System.out.print("\n-----");

        System.out.print("\nAdını gir:");

        ad[i]=input.next();

        System.out.print("Vize notunu gir:");

        vize[i]=input.nextInt();

    }

    int eny=vize[0];

    for (int i=1; i<n; i++)

        if (vize[i]>eny) eny=vize[i];

    System.out.print("\nEn Yüksek Not:"+eny);

    System.out.print("\n-----\n");

    for (int i=0;i<n;i++)

        if (eny==vize[i]) System.out.println(ad[i]);

}
```



```
// Para tutarını uygun banknotlara ayırır.

public static void main(String[] args)

{

Scanner input=new Scanner(System.in);

int [] banknot={ 200,100,50,20,10,5};

String [] para={"ikiyüz","yüz","elli","yirmi","on","beş"};

int adet,tutar;

System.out.print("Para Tutarını Gir:");

tutar=input.nextInt();

System.out.print("-----");

for (int i = 0; i < 6; i++)

{

adet=tutar/banknot[i];

tutar-=adet*banknot[i];

if (adet>0)

System.out.printf("\n%5d adet %10s TL\n",adet,para[i]);

}

if (tutar>0)

System.out.printf("\n%5d TL madeni para\n",tutar);

}
```

```
// Harf Harf kelimeyi tahmin etmeye çalışır
```

```
public static void main(String[] args)
{
    String [] dizi = {"computer", "program", "java", "ayrık", "matematik"};
    Random rnd = new Random();
    int tutulanIndex = rnd.nextInt(dizi.length);
    Scanner scn = new Scanner(System.in);
    String tutulanKelime = dizi[tutulanIndex];
    StringBuilder gosterilenKelime = new StringBuilder();

    for(int i=0; i<tutulanKelime.length(); i++)
        gosterilenKelime.append("*");
    System.out.println(gosterilenKelime);
    int yanlisSayisi = 0;
    while (true)
    {
        String tahmin = scn.next();
        if (tutulanKelime.contains(tahmin))           //Var ise
        {
            for(int i=0; i<tutulanKelime.length(); i++)
                if (tutulanKelime.charAt(i) == tahmin.charAt(0))
                    gosterilenKelime.setCharAt(i, tahmin.charAt(0));
        }
        else yanlisSayisi++;

        System.out.println(gosterilenKelime);

        if (!gosterilenKelime.toString().contains("*")) break;
    }
    System.out.println("Tebrikler Bildiniz...");
    System.out.println("Yanlış tahmin sayısı : " + yanlisSayisi);
}
```

## İki Boyutlu Diziler:

Birden fazla boyutu olan dizilere çok boyutlu diziler denmektedir. Bilimsel uygulamalarda sık sık kullanılan işlemlerden birisi olan matrisler iki boyutlu dizilere en güzel örnektir. Matrisler, bir çok veriyi bünyesinde taşıyan ve her bir satırı bir dizi olarak düşünülen geniş bilgi alanlarıdır. 3 veya daha çok boyutlu diziler teoride dilin yapısına uygun olmasına rağmen gerçek hayatta çok fazla kullanılmazlar. 3 ve daha fazla boyutlu dizilerde işlem yapılması oldukça zordur.

		Sütunlar							
		1	2	3	4	5	6	7	8
Satırlar	1	B(1,1)	B(1,2)	B(1,3)	B(1,4)	B(1,5)	B(1,6)	B(1,7)	B(1,8)
	2	B(2,1)	B(2,2)	B(2,3)	B(2,4)	B(2,5)	B(2,6)	B(2,7)	B(2,8)
	3	B(3,1)	B(3,2)	B(3,3)	B(3,4)	B(3,5)	B(3,6)	B(3,7)	B(3,8)
	4	B(4,1)	B(4,2)	B(4,3)	B(4,4)	B(4,5)	B(4,6)	B(4,7)	B(4,8)

değişken tipi[,] dizi={ {eleman1,eleman2,eleman3,...},{eleman1,eleman2,eleman3,...},...}

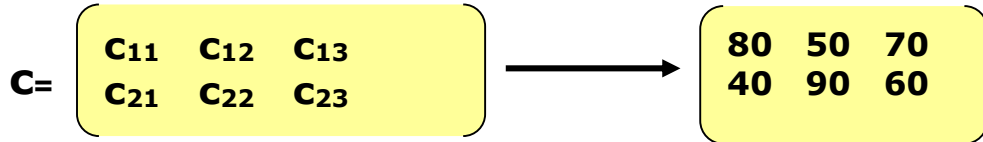
int [][] ikiboyut= { {1,2},{3,4} }; // int [][][] ucboyut= { { {1,2},{3,4} }, { {1,2},{3,4} } };

### Örnek : 3 kişilik bir sınıftaki öğrencilerin 2 ayrı dersten

**JAVA Dersi için : A = [ a1 , a2 , a3 ] → [80, 50, 70]**

**ASP:NET Dersi için : B = [ b1 , b2 , b3 ] → [40, 90, 60]**

Yukarıdaki her bir dersi ayrı dizide tutmak yerine aşağıdaki gibi matris adını verdiğimiz iki boyutlu dizide tutmak daha mantıklıdır.Çünkü verileri ayrı ayrı değişkenlerde tutmak zordur



2 boyutlu 3 elemanlı bir dizinin tanımlanması;

int [ ][ ] ikiboyut = { {80,50,70},{40,90,60} };

```
public static void main(String[] args)
{
    int [][] dizi =new int [2][3];
    dizi [0][0]=80;
    dizi [0][1]=50;
    dizi [0][2]=70;
    dizi [1][0]=40;
    dizi [1][1]=90;
    dizi [1][2]=60;
    for (int i=0; i<2; i++)
        for (int j=0; j<3; j++)
            System.out.printf("\nDizi[%d][%d]=%d",i,j,dizi[i][j]);
}
```

```
public static void main(String[] args)
{
    int [][] dizi =new int [2][3];
    Scanner input=new Scanner(System.in);
    for (int i=0; i<2; i++)
        for (int j=0; j<3; j++)
        {
            System.out.printf("%d. Dersin %d. Notu:", i+1,j+1);
            dizi[i][j]=input.nextInt();
        }
    for (int i=0; i<2; i++)
        for (int j=0; j<3; j++)
            System.out.printf("\nDizi[%d][%d]=%d",i,j,dizi[i][j]);
}
```

```
// birim matris köşegenler=1, diğerleri=0
```

```
public static void main(String[] args)

{

    Scanner klavye=new Scanner(System.in);

    System.out.print("Matris Boyutunu Gir:");

    int n=klavye.nextInt();

    int[ ][ ] matris= new int[n][n];

    for (int i = 0; i < matris.length; i++)

        for (int j = 0; j < matris.length; j++)

            if (i == j) matris[i][j]=1;

            else matris[i][j]=0;

    for (int i = 0; i <matris.length; i++)

    {

        for (int j = 0; j < matris.length; j++)

            System.out.printf(" %d ", matris[i][j]);

        System.out.println();

    }

}
```

```
// nxn boyutunda oluşturulan matrisi toplamını bulur
```

```
public static void main(String[] args)
{
    Scanner klavye=new Scanner(System.in);

    int toplam=0;

    System.out.print("Matris Boyutunu Gir:");

    int n=klavye.nextInt();

    int[ ][ ] dizi= new int[n][n];

    for (int i = 0; i < dizi.length; i++)
        for (int j = 0; j < dizi.length; j++)
        {
            System.out.printf("\nDizi[%d][%d]:",i,j);

            dizi[i][j]=klavye.nextInt();

        }

    for (int i = 0; i < dizi.length; i++)
    {
        for (int j = 0; j < dizi.length; j++)
        {
            System.out.printf(" %d ",dizi[i][j]);

            toplam+=dizi[i][j];

        }

        System.out.println();

    }

    System.out.printf("Matrisin Toplamı: " + toplam);

}
```

```

// Matris izi köşegen elemanlarının toplamı

public static void main(String[] args)

{

    Scanner klavye=new Scanner(System.in);

    System.out.print("Matris Boyutunu Gir:");

    int n=klavye.nextInt();

    int[ ][ ] matris= new int[n][n];

    for (int i = 0; i < matris.length; i++)

        for (int j = 0; j < matris.length; j++)

            {

                System.out.printf("\nMatris[%d][%d]:",i,j);

                matris[i][j]=klavye.nextInt();

            }

    System.out.print("\n-----\n\n");

    int tizMatris=0;

    for (int i = 0; i < matris.length; i++)

        tizMatris += matris[i][i];

    for (int i = 0; i < matris.length; i++)

        {

            for (int j = 0; j < matris.length; j++)

                System.out.printf(" %d ",matris[i][j]);

            System.out.println();

        }

    System.out.print("\n-----\n");

    System.out.printf("Matrisin İzi:" + tizMatris);

}

```

```
// Transpoz satır sütunun yer deęiřtirmesidir
```

```
public static void main(String[] args)
{
    Scanner klavye=new Scanner(System.in);
    System.out.print("Matris Boyutunu Gir:");
    int n=klavye.nextInt();
    int[ ][ ] matris= new int[n][n];
    int[ ][ ] transpoz= new int[n][n];

    for (int i = 0; i < matris.length; i++)
        for (int j = 0; j < matris.length; j++)
        {
            System.out.printf("\nMatris[%d][%d]:",i,j);
            matris[i][j]=klavye.nextInt();
        }
    System.out.print("\n-----\n\n");
    for (int i = 0; i <matris.length; i++)
    {
        for (int j = 0; j <matris.length; j++)
            System.out.printf(" %d ",matris[i][j]);
        System.out.println();
    }
    System.out.print("\n-----\n\n");
    for (int i = 0; i <matris.length; i++)
    {
        for (int j = 0; j <matris.length; j++)
        {
            transpoz[i][j]=matris[j][i];
            System.out.printf(" %d ", transpoz[i][j]);
        }
        System.out.println();
    }
}
```

## Arrays Sınıfı:

```
public static void main(String[] args)
{
    String [] adlar={ "ali","can","ali","elif","ece","veli","harun","ece","can","alp"};
    int [] notlar = { 60 , 80 , 40 , 90 , 70 , 50 , 85 , 30 , 65 , 70 };

    System.out.println("\n-----Temel Diziler-----");
    for (int i = 0; i < gunler.length; i++)
        System.out.print(gunler[i]+" ");

    System.out.println();

    for(int i:notlar)
        System.out.print(i+" ");

    System.out.print("\n\n-----toString-----\n");
    System.out.println(Arrays.toString(gunler));
    System.out.println(Arrays.toString(notlar));

    System.out.print("\n-----Copy-----\n");
    int[] not1 = Arrays.copyOf(notlar, 3);
    System.out.println(Arrays.toString(not1));

    int[] not2=Arrays.copyOfRange(notlar,2,5);
    System.out.println(Arrays.toString(not2));

    int[] not3=Arrays.copyOfRange(notlar,6,9);
    System.out.println(Arrays.toString(not3));

    System.out.print("\n-----Fill-----\n");
    Arrays.fill(not1,50);
    System.out.println(Arrays.toString(not1));

    Arrays.fill(not2,0,2,100);
    System.out.println(Arrays.toString(not2));

    Arrays.fill(not3,50);
    System.out.println(Arrays.toString(not3));

    System.out.print("\n-----equals-----\n");
    System.out.println(Arrays.equals(not1, not2));
    System.out.println(Arrays.equals(not1, not3));
}
```



## **ArrayList Sınıfı:**

Dizilerin en büyük dezavantajlarından birisi diziler için bizim belirlediğimiz bir alan ayrılıyor ve bu alan dışında diziye eleman eklemesinde bulunamıyorduk. Örneğin 100 eleman alabilecek bir diziye 101. elemanı atamak istediğimizde hata alıyorduk. Bu sıkıntıdan kurtulmanın en kolay yolu **ArrayList** aracılığı ile sınırsız bir liste oluşturulabilir ve istediğimiz kadar veriyi bu listeye ekleyebiliriz. Ekledikçe genişleyen bir yapısı vardır.

## **ArrayList Metotları:**

**add(değer)** :Listenin en sonuna yeni bir değer eklemek için kullanılır.

**add(indis, değer)** :Belirlenen indis (index) noktasına yeni bir değer ekler.

**addAll(list)** :Bir listeyi başka bir listenin sonuna ekler.

**addAll(indis, list)** : Bir listeyi başka bir listenin belirlenen indisten itibaren ekler.

**clear()** :Listedeki tüm elemanları temizler.

**contains(değer)** :Girilen değer listede var ise TRUE, yoksa FALSE döndürür.

**equals(list)** :Listeler üzerinden karşılaştırma yapmamızı sağlar.

**indexOf(değer)** :Listede aranan ilk değerın indis (index) değerini döndürür.

**lastIndexOf(değer)** : Listede aranan son değerın indis (index) değerini döndürür.

**get(indis)** : Listede girilen indisin tuttuğu değeri döndürür.

**remove(indis)** : Listede girilen indisin tuttuğu değeri siler.

**remove(değer)** :Listede girilen değeri arayarak siler.

**removeAll(list)** :Belirtilen listedeki tüm elemanları siler.

**set(indis, değer)** :Listede belirtilen indis değerine yeni bir değer atar.

**size()** : Listenin toplam eleman sayısını verir.

**subList(başlangıç, bitiş)** :Listenin belirtilen aralıktaki değerleri döndürür.

**toArray()** :ArrayList'i bir normal bir diziye dönüştürür.

**toString()** :Listedeki elemanları "[5, 7, 10, 2]" şeklinde ekrana yazdırır.

## ArrayList Oluşturmak

```
public static void main(String[] args)
{
    ArrayList <String> ulkeler = new ArrayList<String>();
    // ArrayList ulkeler = new ArrayList();

    ulkeler.add("Türkiye");
    ulkeler.add("Almanya");
    ulkeler.add("İtalya");
    ulkeler.add("Fransa");
    ulkeler.add("İngiltere");

    System.out.println(ulkeler);
    System.out.println("1: " + ulkeler.get(0));
    System.out.println("4: " + ulkeler.get(3));

    ArrayList yeni = new ArrayList();

    yeni.addAll(ulkeler);

    yeni.add("Japonya");
    yeni.add(0,"Çin");
    yeni.add(4,"Fransa");

    yeni.set(2, "İspanya");

    yeni.remove(3);

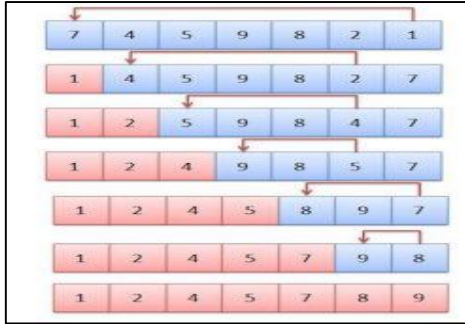
    System.out.println(yeni);
    System.out.println(yeni.size());

    System.out.println(yeni.contains("Japonya"));
    System.out.println(yeni.indexOf("Amerika"));

    System.out.println(yeni.indexOf("Fransa"));
    System.out.println(yeni.lastIndexOf("Fransa"));

    yeni.clear();
    System.out.println(yeni);
    System.out.println(yeni.isEmpty());
}
```

## Dizilerde Sıralama İşlemleri:



**Selection Sort:** Öncelikle bu tekniğin sıralama mantığını anlatalım. Dizinin ilk elemanından başlayarak dizinin sonuna kadar tüm elemanlarla kontrol edilerek en küçük eleman dizinin ilk sırasına alınır. Daha sonra 2.sıradaki elemana geçilir ve aynı işlem tekrarlanarak son elemana kadar gidilir.

// **Array.Sort:** Dizileri sadece artan sırada sıralayabilir.

//Array.Sort birden çok dizilerde kullanılamaz

```
public static void main(String[] args)
{
    String [] adlar= {"ali","zeynep","ali","ceylan","ece"};
    int[] notlar = { 60, 30, 80, 70, 20 };

    for (int i = 0; i < 5; i++)
        System.out.printf("\n%10s %5d",adlar[i],notlar[i]);

    System.out.print("\n-----");

    Arrays.sort(adlar);
    Arrays.sort(notlar);

    System.out.println(Arrays.toString(adlar));
    System.out.println();
    System.out.println(Arrays.toString(notlar));

    System.out.println("\n-----");

    for (int i = adlar.length-1; i >= 0 ; i--)
        System.out.printf("\n%10s %5d",adlar[i],notlar[i]);
}
```

// **Selection Sort**

```
public static void main(String[] args)
{
    String [] adlar= {"ali","zeynep","ali","ceylan","ece"}
    int [] notlar={ 60,30,80,70,20};
    int x;
    String y;

    for (int i=0; i<notlar.length; i++)
        System.out.printf("\n%10s %5d",adlar[i],notlar[i]);

    System.out.print("\n-----\n");
    System.out.println(Arrays.toString(adlar));
    System.out.println(Arrays.toString(notlar));

    for (int i=0; i<notlar.length-1; i++)
        for (int j=i+1;j<notlar.length;j++)
            if (notlar[i]>notlar[j])
                // if (adlar[i].compareToIgnoreCase(adlar[j])>0)
                {
                    x=notlar[i]; notlar[i]=notlar[j]; notlar[j]=x;
                    y=adlar[i]; adlar[i]=adlar[j]; adlar[j]=y;
                }

    System.out.println();
    System.out.println(Arrays.toString(adlar));
    System.out.println(Arrays.toString(notlar));
}
```

## Dizilerde Arama İşlemleri:

**1. Linear(doğrusal) Search:** Sıralı olmayan ve daha çok küçük boyutlu dizilerde kullanılır. Çok büyük ve sıralı olan dizilerde ise Binary(ikili) arama tekniği kullanılmalıdır.

```
// Linear (Doğrusal) Search
```

```
public static void main(String[] args)

{

    String[] adlar={ "ali","can","ali","elif","ece","veli","harun","ece","can","alp" };

    int[] notlar = { 60 , 80 , 40 , 90 , 70 , 50 , 85 , 30 , 65 , 70 };

    System.out.println(Arrays.toString(adlar));

    System.out.println(Arrays.toString(notlar));

    Scanner input=new Scanner(System.in);

    System.out.print("\nAradığınız bilgiyi giriniz:");

    String ara = input.next();

    for (int i=0;i<adlar.length;i++)

        if (ara.equalsIgnoreCase(adlar[i]))

        {

            System.out.printf("\n%s  %d .sıradadır", ara, i + 1);

            System.out.printf("\nVize Notu:%d", notlar[i]);

            System.out.print("\n\nAradığınız Kayıt Bu mu?");

            String cevap = input.next();

            if (cevap.compareToIgnoreCase("Evet")==0) return;

            // if (cevap=="Evet") return;

            // char cevap = input.next().charAt(0);

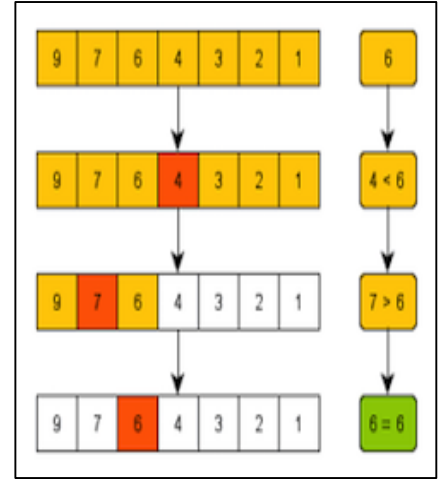
            // if (cevap=='e' || cevap=='E') return;

        }

    System.out.printf("\n%s isimli kişi bulunamadı",ara);

}
```

**2. Binary Search:** Çalışma mantığını kısaca anlatmaya çalışalım. Sadece sıralı dizilerde kullanılan bu teknikte ilk olarak dizinin orta değeri bulunur. Dizi sıralı olduğundan orta değer , aranan değerden büyük / küçük / eşit kontrolü yapılır. Eğer orta değer aranan değerden büyükse dizinin orta değerden sonraki değerlerine bakılmaz. Çünkü değerimiz artık başlangıç değeri ile orta değer arasındadır. Tam tersi olarak orta değer aranan değerden küçük ise dizinin orta değerinden önceki değerlerine bakılmaz. Bu işlem orta değer aranan değere eşit olana kadar devam eder.



### binarySearch Java Metodu

```
public static void main(String[] args)
{
    // Arama yapılacak dizinin sıralı olması gerekir.

    String [] adlar={ "ali","can","ali","elif","ece","veli","harun","ece","can","alp" };

    Arrays.sort(adlar);

    System.out.println(Arrays.toString(adlar));

    Scanner input=new Scanner(System.in);

    System.out.print("\nAradığınız bilgiyi giriniz:");

    String ara = input.next().toLowerCase();

    int index = Arrays.binarySearch(adlar,ara);

    if(index >-1)

        System.out.printf("\n%s %d .sıradadır",ara,index+1);

    else

        System.out.println("\nAranan bilgi bulunamadı.");

}
```

## binarySearch Programlama Mantığı

```
public static void main(String[] args)

{

String [] adlar={ "ali","can","ali","elif","ece","veli","harun","ece","can","alp"};

Arrays.sort(adlar);

System.out.println(Arrays.toString(adlar));

Scanner scan=new Scanner(System.in);

System.out.print("\nAranan bilgiyi giriniz:");

String ara=scan.next();

int konum = binarySearch(adlar,ara);

if (konum != -1) System.out.println("\nBilginin konumu :"+(konum+1));

else System.out.println("\nAranan bilgi bulunamadı");

}

private static int binarySearch(String[] list, String key)

{

int ilk = 0;

int son = list.length-1;

while (ilk<=son)

{

int orta = (int)(ilk+son)/2; // orta değ er bulunacak

if (list[orta].equalsIgnoreCase(key)) return orta; // arama ger çekleřti

else if (list[orta].compareToIgnoreCase(key)>0) son = orta - 1; // orta değ er aranan değ erden büyükse

else ilk = orta + 1; // orta değ er aranan değ erden küçükse

}

return -1; // aranan bulunamazsa -1 dönecek

}
```

## HAZIR FONKSİYONLAR :

### 1. Matematiksel(Math) Fonksiyonlar :

**Math.PI** // 3.14

Math.PI → 3.1416

**Math.pow(x,y)** // x'in y. Kuvveti

Math.pow(2,5) → 32

**Math.sqrt(x)** // x'in karekökü

Math.sqrt(25) → 5

**Math.abs(x)** // x'in mutlak değeri

Math.pow(1) → 1

Math.pow(-1) → 1

**Math.min(x,y)** // İki sayıdan küçük olanı bulur.

Math.min(5,9) → 5

Math.min(-5,-9) → -9

**Math.max(x,y)** // İki sayıdan büyük olanı bulur.

Math.max(5,9) → 9

Math.max(-5,-9) → -5

**Math.round(x)** // Sayıyı en yakın sayıya yuvarlar.

Math.round(8.1) → 8

Math.round(8.9) → 9

**Math.floor(x)** // Sayıyı aşağıya yuvarlar.

Math.floor(8.1) → 8

Math.floor(8.9) → 8

**Math.ceil()** // Sayıyı yukarıya yuvarlar.

Math.ceil(8.1) → 9

Math.ceil(8.9) → 9

**Math.random();** // 0 ile 1 arasında double bir değer üretir

Math.random();

```

public static void main(String[] args)
{
    double x , y;
    String s="";
    Scanner konsol=new Scanner(System.in);
    System.out.println("Birinci gerçel sayıyı gir:");
    x=konsol.nextDouble();
    System.out.println("İkinci gerçel sayıyı gir:");
    y=konsol.nextDouble();

    s+="Matematik kütüphanesi Fonksiyonları";
    s+="\n-----\n" ;

    s+= "Math.max("+x+", "+y+" ) = "+Math.max(x,y)+"\n";

    s+= "Math.min("+x+", "+y+" ) = "+Math.min(x,y)+"\n";
    s+="Math.abs("+x+" ) = "+Math.abs(x)+"\n";

    s+= "Math.PI      = "+Math.PI+"\n";

    s+= "Math.sqrt("+x+" ) = "+Math.sqrt(x)+"\n";

    s+= "Math.pow("+x+", "+y+" ) = "+Math.pow(x,y)+"\n";
    s+="Math.ceil("+x+" ) = "+Math.ceil(x)+"\n";

    s+="Math.floor("+x+" ) = "+Math.floor(x)+"\n";

    s+="Math.round("+x+" ) = "+Math.round(x)+"\n";
    s+="Math.random() = "+Math.random()+"\n";

    s+= "Math.sin("+x+" ) = "+Math.sin(x)+"\n";

    s+= "Math.cos("+x+" ) = "+Math.cos(x)+"\n";

    s+= "Math.tan("+x+" ) = "+Math.tan(x)+"\n";

    s+= "Math.asin("+y+" ) = "+Math.asin(y)+"\n";

    s+= "Math.acos("+y+" ) = "+Math.acos(y)+"\n";

    s+= "Math.atan("+y+" ) = "+Math.atan(y)+"\n";

    s+= "Math.E = "+Math.E+"\n";

    s+= "Math.log("+x+" ) = "+Math.log(x)+"\n";

    s+= "Math.exp("+x+" ) = "+Math.exp(x)+"\n";

    System.out.println(s);
    System.exit(0);
}

```



## 2. KarakterSel(String) Fonksiyonlar:

**length:** String ifadenin karakter sayısını verir.

**int uzunluk=**str.length();

**toLowerCase:** String ifade içindeki büyük karakterleri küçük karakterlere dönüştürür.

**String str=**strmetin.toLowerCase();

**toUpperCase:** String ifade içindeki küçük karakterleri büyük karakterlere dönüştürür.

**String str=**strmetin.toUpperCase();

**trim metodu:** String ifadelerin başındaki ve sonundaki tab yada boşlukları siler.

**String str=**strmetin.trim();

**concat:** String ifadeleri birleştirir.

**String str =** strmetin.concat(String str2);

**charAt:** String içerisinde belirtilen index değerindeki karakter değeri döner.

**char str.charAt(int indis)**

**substring:** String ifadeden karakter kopyalamamızı sağlar.

**String str=**strmetin.substring(int start, int adet);

**replace:** String ifade içerisinde bulunan karakteri yenisi ile değiştirir.

**String str=**strmetin.replace(char oldChar, char newChar);

**split:** String ifadeleri parçalamamızı sağlar.

**String strMetin =** "Ahmet, Ayşe, Nalan, Yasin, Sinan";

**String [] str =** strMetin.split(", ");

**startsWith/endsWith:**String ifadenin belirtilen değerle başlayıp başlamadığını kontrol eder.

**boolean str =** strmetin.startsWith(String str1);

**boolean str =** strmetin.endsWith(String str1);

**equals / equalsIgnoreCase:** Nesne olarak verilen ifadeyi başka bir değerle eşitliğini kontrol eder.

**boolean str=(string str1.equals(string str2))**

**indexOf:** String ifade içerisinde belirtilen karakterin hangi sırada olduğunu döndürür. Yoksa -1 döner

**int str=**strmetin.indexOf(String str1) // Soldan aramaya başlar

**int str=**strmetin.lastIndexOf(String str1) // Sağdan aramaya başlar

**compareTo / compareToIgnoreCase:** İki string ifadeyi alfabetik olarak karşılaştırır

**int str=**strmetin.compareTo(String str1)

**contains metodu:** Bir string içerisinde arama işlemi yapmamızı sağlar.

**String str=**strmetin.contains(String aranan);

**isEmpty():** String değer boş mu dolumu kontrolü yapar. true/false döner.

**if( strmetin != null && !strmetin.trim().isEmpty())**

```

public static void main(String[] args)
{
// String Sınıfının karakter sayısını bulmak
String string1="Mustafa Kemal Üniversitesi Antakya Meslek Yüksekokulu";
int strUzunluk=string1.length();
System.out.println("\n(1).Karakter Sayısı:"+strUzunluk);

// toUpperCase ve toLowerCase metodları ile stringi büyük veya küçük yazdırma
String string2="Mustafa Kemal Üniversitesi";
String string3="ANTAKYA MESLEK YÜKSEKOKULU";
System.out.println("\n(2)."+string2.toUpperCase());
System.out.println("(2)."+string3.toLowerCase());

// + operatörü ile string birleştirmek
String a1="Çanak";
String a2="kale";
String a3=a1+a2;
System.out.println("\n(3)."+a3);

//Concat metodu ile string birleştirmek
String b1="Gazi";
String b2="antep";
String b3=b1.concat(b2);
System.out.println("\n(4)."+b3);

// charAt metodu ile karaktere erişmek
String c1="Mustafa Kemal Üniversitesi Antakya Meslek Yüksekokulu";
System.out.println("\n(5). "+c1.charAt(9));

//trim metodu ile bir string ifadesinin yanındaki boşlukları silme
String d1="      Antakya Meslek Yüksekokulu      ";
System.out.println("\n(6)."+d1.trim());

// substring metodu ile string içerisinde herhangi bir bölüm seçme
String e1="Java ile String işlemleri yapmak";
String e2="";
System.out.println("\n(7)."+(e2=e1.substring(0,4)));

// replace metodu ile string içerisinde herhangi bir karakteri değiştirme
String f1="İstanbul Türkiye'nin en büyük şehridir.";
System.out.println("\n(8)."+f1.replace("ü", "u"));

```

```

// startsWith/endsWith metodları ile başlangıç ve bitiş karakterlerini kontrol etme
String g1="İstanbul Turkiyenin en büyük şehridir.";
System.out.println("\n(9)."+g1.startsWith("İ"));
System.out.println("(9)."+ g1.endsWith("ilidir"));

// String ifadeleri karşılaştırma
String h1="Hatay";
String h2="Hatay";
String h3="hatay";

System.out.println("\n(10)."+h1+"-"+h2+" ==> "+h1.equals(h2));
System.out.println("(10)."+h1+"-"+h2+" ==> "+h1.equalsIgnoreCase(h2));
System.out.println("(10)."+h1+"-"+h3+" ==> "+h1.equals(h3));
System.out.println("(10)."+h1+"-"+h3+" ==> "+h1.equalsIgnoreCase(h3));

System.out.println("\n(10)."+h1+"-"+h2+" ==> "+h1.compareTo(h2));
System.out.println("(10)."+h1+"-"+h2+" ==> "+h1.compareToIgnoreCase(h2));
System.out.println("(10)."+h1+"-"+h3+" ==> "+h1.compareTo(h3));
System.out.println("(10)."+h1+"-"+h3+" ==> "+h1.compareToIgnoreCase(h3));

// IndexOf ve LastIndexOf metodları ile baştan ve sondan Karakter arama
String k1="Mustafa Kemal Üniversitesi Antakya Meslek Yüksekokulu";
System.out.println("\n(11). e karakteri için ilk konum: "+k1.indexOf("e"));
System.out.println("(11). e karakteri için son konum: "+k1.lastIndexOf("e"));
System.out.println("(11). b karakteri için ilk konum: "+k1.indexOf("b"));
System.out.println("(11). Antakya kelimesi için ilk konum: "+k1.indexOf("Antakya"));
System.out.println("(11). e karakteri 7 indisten sonraki ilk konumu:"+k1.indexOf("e",7));
System.out.println("(11). e karakteri 10 indisten önceki son konumu:"+k1.lastIndexOf("e",10)+"\n");

System.out.println("\n(12). Dizi Listesi");
String str = "Fatma,Kemal,Hakan,Hüseyin,Emre"; String[] dizi = str.split(",");
// String str = "Fatma Kemal Hakan Hüseyin Emre"; String[] dizi = str.split("\\s");
for (String indis : dizi) System.out.println(indis);
}

```

### 3. Dosya (File) Fonksiyonları:

```
// klasor.java

import java.io.File;

import java.util.Date;

import java.util.Scanner;

public static void main(String[] args)

{

    System.out.println("\nKlasör Özellikleri");

    System.out.println("-----");

    File klasor = new File("C:\\Projeler");

    // File klasor = new File("C:\\Projeler\\Java");

    if (klasor.exists()) System.out.println(klasor.getName()+" isimli klasör var...");

    else if (klasor.mkdir()) System.out.println(klasor.getName()+" isimli klasör Oluşturuldu.");

    else System.out.println(klasor.getName()+" isimli klasör Oluşturulamadı.");

    if (klasor.isDirectory()) System.out.println(klasor.getName()+" Bir Klasördür.");

    else System.out.println(klasor.getName()+" Bir Klasör Değildir.");

    System.out.println("Klasör Adresi : " + klasor.getAbsolutePath());

    System.out.println("Path Ayırıcı : " + klasor.separator);

    System.out.println("Klasör Uzunluğu:"+klasor.length());

    Date erisim = new Date(klasor.lastModified());

    System.out.println("Klasör Son Erişim Tarihi:"+erisim);

    System.out.println("\nKlasör İçerik Listesi");

    System.out.println("-----");

    File liste[] = klasor.listFiles();

    for (File indis : liste)

    {

        if (indis.isFile()) System.out.println(indis);

        if (indis.isDirectory())System.out.println(indis);

    }

}
```

```

// dosya.java

import java.io.File;

public static void main(String[] args)
{
    System.out.println("\nDosya Özellikleri");
    System.out.println("-----");

    File dosya = new File("C:\\Projeler\\ogrenciler.txt");

    if (dosya.exists()) System.out.println(dosya.getName()+" isimli dosya var.");
    else if (dosya.createNewFile()) System.out.println(dosya.getName()+" isimli dosya oluşturuldu.");
    else System.out.println(dosya.getName()+" isimli dosya oluşturulamadı.");

    if (dosya.isFile()) System.out.println(dosya.getName()+" Bir Dosyadır.");
    else System.out.println(dosya.getName()+" Bir Dosya Değildir.");

    dosya.setReadOnly();

    System.out.println("\nDosya Gizlimi : " + dosya.isHidden());
    System.out.println("Dosya Okunabilir mi : " + dosya.canRead());
    System.out.println("Dosya Yazılabilir mi : " + dosya.canWrite());

    dosya.setWritable(true);

    System.out.println("\nDosya Okunabilir mi : "+dosya.canRead());
    System.out.println("Dosya Yazılabilir mi : "+dosya.canWrite());
    // System.out.println("Dosya Silindimi : " + dosya.delete());
}

```

```

// ReadWriteFile.java

import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileReader;
import java.io.FileWriter;
import java.util.Scanner;

public static void main(String[] args) throws Exception
{
    System.out.println("\nDosyaya Yazma İşlemleri");
    System.out.println("-----");

    Scanner input=new Scanner(System.in);
    String path = "c:\\Projeler\\ogrenciler.txt";

    File dosya = new File(path);
    if (!dosya.exists()) dosya.createNewFile();

    FileWriter fw = new FileWriter(dosya,false);    // dosyayı yazma amaçlı tanımlar.
    //FileWriter fw = new FileWriter(dosya,true);    // dosyayı ekleme amaçlı tanımlar.

    BufferedWriter bw = new BufferedWriter(fw);    // dosyaya kayıt yazdırır.

    String ad,soyad; int not, i=0;

    while (true)
    {
        i++;
        System.out.println(i+" . Ogrencinin");
        System.out.println("-----");
        System.out.print("Adı:"); ad=input.next();
        if (ad.equalsIgnoreCase("exit")) break;
        System.out.print("Soyadı:"); soyad=input.next();
        System.out.print("Vize Notu:"); not=input.nextInt();
        bw.write(ad+" "+" "+soyad+" "+not);
        bw.newLine(); System.out.println();
    }
    bw.flush(); bw.close(); fw.close();

    System.out.println("\nDosyadan Okuma İşlemleri");
    System.out.println("-----");

    FileReader fr=new FileReader(path);    // dosyayı okuma amaçlı tanımlar.
    BufferedReader br=new BufferedReader(fr);    // dosyadan kayıt okur.
    String str;
    while((str=br.readLine())!=null)
        System.out.println(str+"\n");
    fr.close();
}

```

## JAVA NESNE PROGRAMLAMA:

### FORM TASARIMI:

**Müşteri Bilgileri**

Adı:  Soyadı:

Cep Tel:  E-Posta:

Adresi:

**Sipariş Bilgileri...**

Ürün Adı:

Seçenekler: ☐ Ketçap ☐ Mayonez ☐ Patates

Ürün Adedi:

Birim Fiyatı:

Ödeme Türü: ☐ Kredi Kartı ☐ Kapıda Ödeme

Sipariş Özeti:

- Item 1
- Item 2
- Item 3
- Item 4
- Item 5

Sipariş Ver...  
Sipariş Değiştir...  
Sipariş İptal...

### Dialog(JOptionPane) Pencereleeri:

Java'da JOptionPane kullanarak veri girişlerini ekrana çıkan kutudan yapabilir sonuçları mesaj kutuları ile ekrana verebilir , hata mesajları, bilgi mesajları oluşturabilirsiniz. JOptionPane ile projelerinizin görsel açıdan daha güzel bir platforma taşıyabilirsiniz.

**JOptionPane** sınıfını kullanabilmemiz için projeye **javax.swing.JOptionPane** kütüphanesini eklememiz gerekiyor.

**JOptionPane.showInputDialog(null, "mesaj","başlık",simge);** // Giriş

**JOptionPane.showMessageDialog(null, "mesaj","başlık",düğme,simge);** // Çıkış

**JOptionPane.showConfirmDialog(null, "mesaj","başlık",düğme,simge);** // Onay

Kod	Simgeler	Kod	Düğmeler
-1	JOptionPane.PLAIN_MESSAGE	-1	JOptionPane.DEFAULT_OPTION
0	JOptionPane.ERROR_MESSAGE	0	JOptionPane.YES_NO_OPTION
1	JOptionPane.INFORMATION_MESSAGE	1	JOptionPane.YES_NO_CANCEL_OPTION
2	JOptionPane.WARNING_MESSAGE	2	JOptionPane.OK_CANCEL_OPTION
3	JOptionPane.QUESTION_MESSAGE		
		<b>Onay</b>	<b>Düğme</b>
		Ok	JOptionPane.OK_OPTION
		Yes	JOptionPane.YES_OPTION
		No	JOptionPane.NO_OPTION
		Cancel	JOptionPane.CANCEL_OPTION
		Close	JOptionPane.CLOSE_OPTION

## Örnek: Dialog

```
// Fatura Hesaplama
import javax.swing.JOptionPane;

public class Dialog
{
    public static void main(String[] args)
    {
        int fiyat , pesinat , taksit , aylikOdeme ;
        String musteriAd, urunAd, fatura="";

        do
        {
            musteriAd=JOptionPane.showInputDialog(null,"Müşteri Adını Gir:", "müşteri", -1);
            urunAd=JOptionPane.showInputDialog(null,"Ürün Adını Gir:", "ürün",0);
            fiyat=Integer.parseInt(JOptionPane.showInputDialog(null,"Fiyatını Gir: ", "fiyat", 1));
            pesinat=Integer.parseInt(JOptionPane.showInputDialog(null,"Peşinatı Gir:", "peşinat", 2));
            taksit=Integer.parseInt(JOptionPane.showInputDialog(null,"Taksit Sayısını Gir: ", "Taksit", 3));

            aylikOdeme = ( fiyat - pesinat ) / taksit ;

            fatura="";
            fatura+="Müşteri Adı:"+musteriAd;
            fatura+="\nÜrün Adı:"+urunAd;
            fatura+="\nSatış Fiyatı:"+fiyat;
            fatura+="\nÖdenen Peşinat:"+pesinat;
            fatura+="\nTaksit Sayısı:"+taksit;
            fatura+="\nAylık Taksit Tutarı:"+aylikOdeme;

            JOptionPane.showMessageDialog(null,fatura,"Fatura Ekranı",JOptionPane.INFORMATION_MESSAGE);
            int cevap=JOptionPane.showConfirmDialog(null, "Devam mı?", "X", JOptionPane.YES_NO_OPTION);
            if (cevap!=JOptionPane.YES_OPTION) break;
        }
        while (true);
    }
}
```



## Örnek:

### // Form1.java

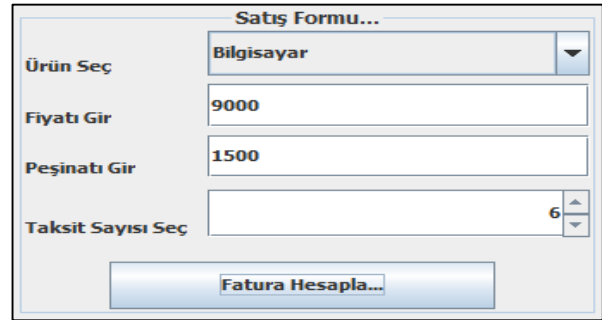
```
public class Form1 extends javax.swing.JFrame
{
    private javax.swing.JComboBox<String> cmbUrun;
    private javax.swing.JTextField txtFiyat;
    private javax.swing.JTextField txtPesinat;
    private javax.swing.JSpinner spnTaksit;
    private javax.swing.JButton btnHesapla;

    public Form1()
    {
        initComponents();
        txtFiyat.setText("");
        txtPesinat.setText("");
        cmbUrun.addItem("Bilgisayar");
        cmbUrun.addItem("Televizyon");
        cmbUrun.addItem("Cep Telefonu");
    }

    private void btnHesaplaActionPerformed(java.awt.event.ActionEvent evt)
    {
        String urunAd=cmbUrun.getSelectedItem().toString();
        int fiyat=Integer.parseInt(txtFiyat.getText());
        int pesinat=Integer.parseInt(txtPesinat.getText());
        int taksit=Integer.parseInt(spnTaksit.getValue().toString());

        String fatura="";
        double aylıkOdeme = ( fiyat - pesinat ) / taksit ;
        fatura+="Ürün Adı:"+urunAd;
        fatura+="\nSatış Fiyatı:"+fiyat;
        fatura+="\nÖdenen Peşinat:"+pesinat;
        fatura+="\nTaksit Sayısı:"+taksit;
        fatura+="\nAylık Taksit Tutarı:"+aylıkOdeme;

        Form2 frm2=new Form2();
        frm2.setVisible(true);
        frm2.txtFatura.setText(fatura);
        // frm2.setLocationRelativeTo(null);
        // frm2.setDefaultCloseOperation(Form2.DISPOSE_ON_CLOSE);
    }
}
```



### // Form2.java

```
public class Form2 extends javax.swing.JFrame
{
    public javax.swing.JTextArea txtFatura;

    public Form2()
    {
        initComponents();
    }
}
```



## METOTLAR:

Java Programlama dilinde temel düşünce programın bütün kısımlarının bloklar halinde yazılmasıdır. Metodlar java programlarının ana parçalarıdır. Metodlar sınıfların(class) içinde yer alan küçük program parçacıklarıdır. Metodların çoğunda değişken parametreler metotlar ve sınıflar arasında iletişimi sağlar. Ayrıca her metodun kendine özgü değişkenleri bulunur. Metod yapısının ana sebebi programları modüler hale getirmektir.

Ana program (**main**) metodu ise özel bir sınıfta bulunması gerekmediği gibi herhangi bir sınıfta bulunabilir veya belirtilmeyebilir. Erişim belirleyicisi **public**, **protected** ve **private** olarak seçilebileceği gibi hiçte yazılmayabilir. Erişim belirleyicisi belirtilmediği takdirde **default** olarak işlem görecektir. Metodun **private** olarak seçilmesi, sadece tanımlandığı sınıfın içerisinden kullanılabileceği, **public** olarak seçilmesi ise herkes tarafından kullanılabileceği anlamına gelir.

Java metodları programın bütünü üzerinde geçerlidir. Dolayısıyla kullanıcı tanımlı metodlar **main()** bloğundan önce veya sonra tanımlanabilir. Ancak bir metod başka bir metod içerisinde tanımlanamaz. Metodlar **main()** bloğunun içinden, başka bir metodun içinden veya kendi kendini çağırabilir.

Metodun geri dönüş tipinin verilmesindeki amaç, metodun kendisini çağıran noktaya geri götüreceği değerin tipinin belirlenmesidir. Tanımlanan metod hangi türde (int, float, double, char, string, bool, array) değer gönderecekse bunu metodu tanımlarken mutlaka belirtmeliyiz. Eğer metodun içinden dışarıya değer çıkarılmayacaksa metod void (değer aktarmayan) türünden seçilebilir.

Metodun içinden dışarıya çıkarılacak değer return komutuna aktarılması gerekir. Bu komut kendisine aktarılan değeri metod ismine vererek dışarı çıkarır. Metodun çağrıldığı noktadan metod içerisine değer aktarımı söz konusu ise parametreler kullanılmaktadır.

**DİKKAT!** Bir sınıf içerisinde sınırsız sayıda methot oluşturulabilir fakat bir methot içerisinde farklı bir methot oluşturulamaz!



## Metot Türleri:

**Metotlar...**

**Birinci Değer:**

**İkinci Değer:**

**jLabel3**

**Sonuçlandır...**

### 1. Parametresiz Metotlar

#### Geriye Değer Döndürmeyen Metotlar

```
// Geri Değer Döndürmeyen Metotlar

private void islem()
{
    int x=Integer.parseInt(veri1.getText());
    int y=Integer.parseInt(veri2.getText());
    label3.setText("Sonuç:"+String.valueOf(x+y));
}

private void sonucActionPerformed()
{
    islem();
}
```

#### Geriye Değer Döndüren Metotlar

```
// Geri Değer Döndüren Metotlar

private int islem()
{
    int x=Integer.parseInt(veri1.getText());
    int y=Integer.parseInt(veri2.getText());
    return(x+y);
}

private void sonucActionPerformed()
{
    label3.setText("Sonuç:"+String.valueOf(islem()));
}
```

### 2. Parametrelili Metotlar

```
// Geri Değer Döndürmeyen Metotlar

private void islem(int a,int b)
{
    label3.setText("Sonuç:"+String.valueOf(a+b));
}

private void sonucActionPerformed()
{
    int x=Integer.parseInt(veri1.getText());
    int y=Integer.parseInt(veri2.getText());
    islem(x,y);
}
```

```
// Geri Değer Döndüren Metotlar

private int islem(int a,int b)
{
    return(a+b);
}

private void sonucActionPerformed()
{
    int x=Integer.parseInt(veri1.getText());
    int y=Integer.parseInt(veri2.getText());
    label3.setText("Sonuç:"+String.valueOf(islem(x,y)));
}
```

### 3. Overloading (Aşırı Yükleme) Metotlar

```
static int islem(int a, int b)
{
    return a + b;
}
static double islem(double a, double b)
{
    return a + b;
}
static String islem(String a, String b)
{
    return a + b;
}

private void sonucActionPerformed()
{
    int x=Integer.parseInt(veri1.getText());
    int y=Integer.parseInt(veri2.getText());

    // double x=Double.parseDouble(veri1.getText());
    // double y=Double.parseDouble(veri2.getText());

    // String x =veri1.getText();
    // String y =veri2.getText();

    goster.setText("Sonuç:"+String.valueOf(islem(x,y)));
}
```

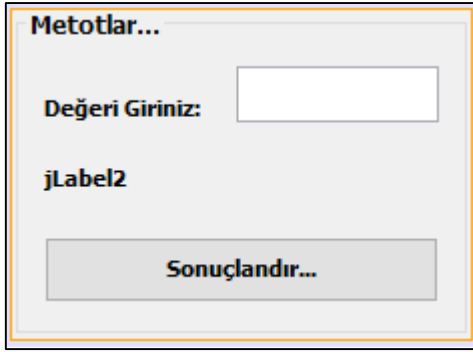
### 4. Kendi kendini çağıran Metotlar

```
static int islem(int a)
{
    if (a == 1) return 1;
    else return a + islem(a - 1);
}
static double islem(int a)
{
    if (a == 1) return 1;
    else return a * islem(a - 1);
}

private void sonucActionPerformed()
{
    int x=Integer.parseInt(veri1.getText());
    goster.setText("Sonuç:"+String.valueOf(islem(x)));

    // double x=Double.parseDouble(veri1.getText());
    //goster.setText("Sonuç:"+String.format("%.0f",islem(x)));
}
```

## Uygulamalar:



// Klavyeden girilen sayının rakamlarını toplar.

```
public static String rTopla(int sayi)
{
    int toplam = 0;
    while(sayi > 0)
    {
        toplam += (sayi % 10);
        sayi /= 10;
    }
    return String.valueOf(toplam);
}

private void sonucActionPerformed()
{
    int x=Math.abs(Integer.parseInt(veri.getText()));
    goster.setText("Sonuç:"+rTopla(x));
}
```

// Klavyeden girilen sayının bölenlerini listeler.

```
public static int bolen(int a,int b)
{
    if (a % b == 0) return (b);
    else return 0;
}

private void sonucActionPerformed()
{
    int x=Integer.parseInt(veri.getText());
    String text="";
    for (int i=1; i<=x; i++)
        if (bolen(x,i)==i) text+=i+" ";
    goster.setText(x+" sayısının bölenleri:" + text);
}
```

// Klavyeden girilen mesajı ters çevirir.

```
public static String tersCevir(String str)
{
    String strTers = "";
    for(int i=str.length()-1; i>=0; i--)
        strTers=strTers+str.charAt(i);
    return(strTers);
}

private void sonucActionPerformed()
{
    String text=veri.getText();
    goster.setText(tersCevir(text));
}
```

// Mesaj içindeki a karakterinin sayısını bulur

```
public static int saydir(String kelime)
{
    int adet=0;
    for(int i=0; i<kelime.length(); i++)
        if (kelime.charAt(i)=='a') adet++;
    return adet;
}

private void sonucActionPerformed()
{
    String text=veri.getText().toLowerCase();
    goster.setText(String.valueOf(saydir(text)));
}
```

## Örnek:

```
public class Form extends javax.swing.JFrame
{
    private javax.swing.JTextField txtPara;
    private javax.swing.JButton btnPara;
    private javax.swing.JTextArea txtEkran;

    private void btnParaActionPerformed()
    {
        int miktar=0, adet;
        String txtAktor="";
        int [] para = { 200, 100, 50, 20, 10, 5 };
        txtEkran.setText("");
        txtEkran.setEditable(false);

        try
        {
            miktar = Integer.parseInt(txtPara.getText());
        }
        catch(NumberFormatException e)
        {
            JOptionPane.showMessageDialog(null,"Geçersiz tutar...", "X",0);
        }

        for (int i = 0; i < 6; i++)
        {
            adet = (int)(miktar / para[i]);
            miktar = miktar - adet * para[i];
            if (adet > 0) txtAktor+=adet + " tane " + para[i]+" TL\n";
        }

        if (miktar > 0)
            txtAktor+="Kalan " + miktar + " TL madeni para\n";

        txtEkran.setText(txtAktor);
    }
}
```

**Bankamatik**

Para Miktarı: 789 Para Çek...

3 tane 200 TL  
1 tane 100 TL  
1 tane 50 TL  
1 tane 20 TL  
1 tane 10 TL  
1 tane 5 TL  
Kalan 4 TL madeni para

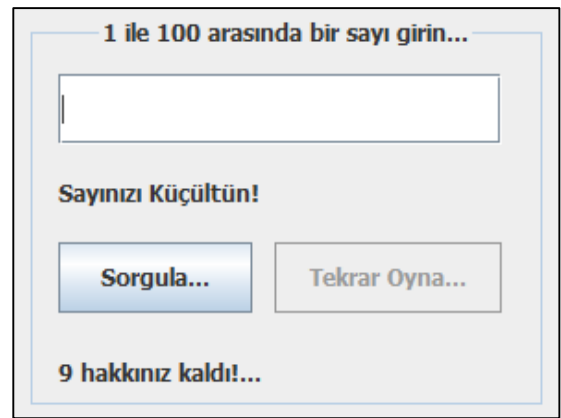
## Örnek:

```
public class Form1 extends javax.swing.JFrame
{
    private javax.swing.JTextField txtSayi;
    private javax.swing.JLabel lblMesaj;
    private javax.swing.JLabel lblHak;
    private javax.swing.JButton btnSorgula;
    private javax.swing.JButton btnTekrar;

    private int hak=0;
    private double Rastgele=Math.random()*101;

    private void btnSorgulaActionPerformed()
    {
        btnTekrar.setEnabled(false);
        hak++;
        int sayi=Integer.parseInt(txtSayi.getText());
        Rastgele=Math.floor(Rastgele);
        if (sayi<Rastgele)
        {
            lblMesaj.setText("Sayınızı Büyütün!");
            txtSayi.setText("");
            txtSayi.requestFocus();
        }
        else if (sayi>Rastgele)
        {
            lblMesaj.setText("Sayınızı Küçültün!");
            txtSayi.setText("");
            txtSayi.requestFocus();
        }
        else if (sayi==Rastgele)
        {
            btnSorgula.setEnabled(false);
            btnTekrar.setEnabled(true);
            lblMesaj.setText(hak + " defada bildiniz!");
        }

        lblHak.setText((10 - hak) + " hakkınız kaldı!...");
        if (hak==10)
        {
            lblHak.setText(Rastgele +" sayısını bilemedin!");
            btnSorgula.setEnabled(false);
            btnTekrar.setEnabled(true);
        }
    }
}
```



1 ile 100 arasında bir sayı girin...

Sayınızı Küçültün!

Sorgula... Tekrar Oyna...

9 hakkınız kaldı!...

```
private void btnTekrarActionPerformed()
{
    Rastgele=Math.random()*101;

    hak = 0; txtSayi.setText("");

    txtSayi.requestFocus();

    btnSorgula.setEnabled(true);

    lblMesaj.setText("");

    lblHak.setText("10 Hakkınız Var...");
}
```

## Örnek:

```
public class Form1 extends javax.swing.JFrame
{
    private javax.swing.JTextField txtVize;
    private javax.swing.JTextField txtFinal;
    private javax.swing.JTextField txtBut;
    private javax.swing.JLabel lblGNotu1;
    private javax.swing.JLabel lblGNotu2;
    private javax.swing.JButton btnHesap;
    private javax.swing.JPanel pnlBut;

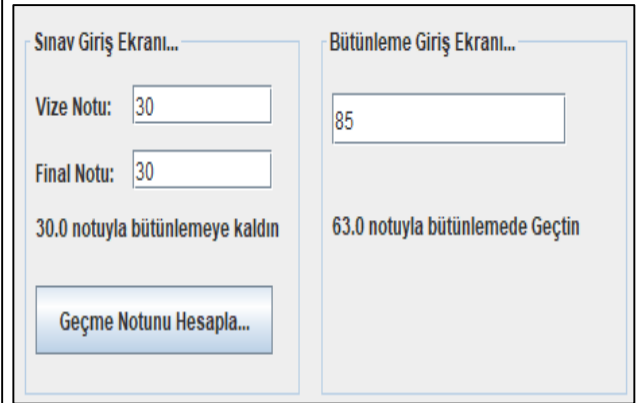
    private int vnotu, fnotu, bnotu;
    public double ort;

    // Form arası veri aktarımı
    // public static int vnotu;

    public Form1()
    {
        initComponents();
        lblVize.setText("Vize Notu:");
        lblFinal.setText("Final Notu:");
        txtVize.setText("");
        txtFinal.setText("");
        lblGNotu1.setText("");
        btnHesap.setText("Geçme Notunu Hesapla...");
        pnlBut.hide();
    }

    private void btnHesapActionPerformed()
    {
        vnotu=Integer.parseInt(txtVize.getText());
        fnotu=Integer.parseInt(txtFinal.getText());
        ort=Math.round(vnotu*0.4+fnotu*0.6);
        lblGnotu1.show();
        if (ort<60)
        {
            lblGnotu1.setText(ort+" notuyla büt kaldın");
            pnlBut.show();
            txtBut.setText("");
            lblGNotu2.setText("");
            txtBut.requestFocus();

            // Formlar arası geçiş
            // new Form2().setVisible(true);
            // new Form2().show();
        }
        else
        {
            lblGnotu1.setText(ort+" notuyla finalde Geçtin");
        }
    }
}
```



```
private void txtButKeyReleased()
{
    int bnotu=Integer.parseInt(txtBut.getText());
    double ort=Math.round(vnotu*0.4+bnotu*0.6);

    // Farklı formdan veri çekme
    // double ort=Math.round(Form1.vnotu*0.4+bnotu*0.6);

    if (ort<60)
    {
        lblGNotu2.setText(ort+" notuyla ders tekrar...");
    }
    else
    {
        lblGNotu2.setText(ort+" notuyla büt Geçtin...");
    }
}

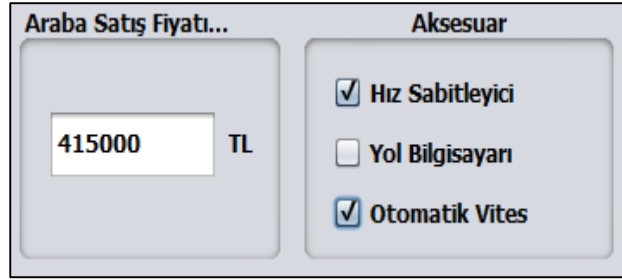
private void txtVizeMouseClicked()
{
    pnlBut.hide();
    txtVize.selectAll();
}

private void txtFinalMouseClicked()
{
    pnlBut.hide();
    txtFinal.selectAll();
}
```



## Örnek: CheckBox1

```
public class Form extends javax.swing.JFrame
{
    private javax.swing.JTextField txtFiyat;
    private javax.swing.JCheckBox chkHız;
    private javax.swing.JCheckBox chkYol;
    private javax.swing.JCheckBox chkVites;
```



```
private void txtFiyatKeyReleased(java.awt.event.KeyEvent evt)
{
    chkHız.setSelected(false);
    chkYol.setSelected(false);
    chkVites.setSelected(false);
}

private void chkHızActionPerformed(java.awt.event.ActionEvent evt)
{
    if (chkHız.isSelected())
        txtFiyat.setText(String.valueOf(Integer.parseInt(txtFiyat.getText()+10000)));
    else
        txtFiyat.setText(String.valueOf(Integer.parseInt(txtFiyat.getText()-10000)));
}

private void chkYolActionPerformed(java.awt.event.ActionEvent evt)
{
    if (chkYol.isSelected())
        txtFiyat.setText(String.valueOf(Integer.parseInt(txtFiyat.getText()+15000)));
    else
        txtFiyat.setText(String.valueOf(Integer.parseInt(txtFiyat.getText()-15000)));
}

private void chkVitesActionPerformed(java.awt.event.ActionEvent evt)
{
    if (chkVites.isSelected())
        txtFiyat.setText(String.valueOf(Integer.parseInt(txtFiyat.getText()+20000)));
    else
        txtFiyat.setText(String.valueOf(Integer.parseInt(txtFiyat.getText()-20000)));
}
```

## Örnek: CheckBox2

```
public class Form extends javax.swing.JFrame
{
    private javax.swing.JTextField txtUrunAd;
    private javax.swing.JTextField txtFiyat;
    private javax.swing.JTextField txtAdet;
    private javax.swing.JCheckBox chkBayi;
    private javax.swing.JCheckBox chkKdv;
    private javax.swing.JTextField txtFatura;

    protected void temizle()
    {
        txtFatura.setText("");
        txtFatura.setEditable(false);
        chkBayi.setSelected(false);
        chkKdv.setSelected(false);
    }

    public Form()
    {
        initComponents();
        temizle();
    }

    private void txtFiyatFocusGained(java.awt.event.FocusEvent evt)
    {
        temizle();
        txtFiyat.setText("");
        txtAdet.setText("");
    }

    private void txtAdetFocusGained(java.awt.event.FocusEvent evt)
    {
        temizle();
        txtAdet.setText("");
    }

    private void txtFiyatKeyTyped(java.awt.event.KeyEvent evt)
    {
        char karakter = evt.getKeyChar();
        if ((karakter < '0' || karakter > '9') && (karakter != '\b')) evt.consume();
    }

    private void txtAdetKeyTyped(java.awt.event.KeyEvent evt)
    {
        char karakter = evt.getKeyChar();
        if ((karakter < '0' || karakter > '9') && (karakter != '\b')) evt.consume();
    }
}
```

Ürün Sipariş Ekranı...

Ürün Adı: Bilgisayar

Ürün Fiyatı: 7500

Ürün Adedi: 3

☒ Bayi İndirimi ☒ KDV Dahil

Fatura Tutarı: 19912

```

protected void hesapla()
{
    int adet, fiyat, fatura;
    fiyat = Integer.parseInt(txtFiyat.getText());
    adet = Integer.parseInt(txtAdet.getText());
    fatura = fiyat * adet;
    if (chkBayi.isSelected()) fatura = (int)(fatura - fatura * 0.25);
    if (chkKdv.isSelected()) fatura = (int)(fatura + fatura * 0.18);
    txtFatura.setText(String.valueOf(fatura));
}

private void txtFiyatKeyReleased(java.awt.event.KeyEvent evt)
{
    hesapla();
}

private void txtAdetKeyReleased(java.awt.event.KeyEvent evt)
{
    hesapla();
}

private void chkBayiActionPerformed(java.awt.event.ActionEvent evt)
{
    hesapla();
}

private void chkKdvActionPerformed(java.awt.event.ActionEvent evt)
{
    hesapla();
}

```

## Örnek: CheckBox3

```
public class Form extends javax.swing.JFrame
{
    private javax.swing.JTextField txtSayi1;
    private javax.swing.JTextField txtSayi2;
    private javax.swing.JButton btnHesap;
    private javax.swing.JButton btnClear;
    private javax.swing.JCheckBox chkTopla;
    private javax.swing.JCheckBox chkCikar;
    private javax.swing.JCheckBox chkCarp;
    private javax.swing.JCheckBox chkBol;
    private javax.swing.JTextArea txtAreaSonuc;
```

```
public Form()
{
```

```
    initComponents();
    txtAreaSonuc.setEditable(false);
}
```

```
private void btnHesapActionPerformed(java.awt.event.ActionEvent evt)
{
```

```
    double S1,S2,sonuc;
    String txtSonuc="";
    S1=Double.parseDouble(txtSayi1.getText());
    S2=Double.parseDouble(txtSayi2.getText());
    if (chkTopla.isSelected())
    {
        sonuc = S1 + S2;
        txtSonuc+=txtSayi1.getText() + " + " + txtSayi2.getText() + " = " +sonuc+"\n";
    }
    if (chkCikar.isSelected())
    {
        sonuc = S1 - S2;
        txtSonuc+=txtSayi1.getText() + " - " + txtSayi2.getText() + " = " +sonuc+"\n";
    }
    if (chkCarp.isSelected())
    {
        sonuc = S1 * S2;
        txtSonuc+=txtSayi1.getText() + " X " + txtSayi2.getText() + " = " +sonuc+"\n";
    }
    if (chkBol.isSelected())
    {
        sonuc = S1 / S2;
        txtSonuc+=txtSayi1.getText() + " / " + txtSayi2.getText() + " = " +sonuc+"\n";
    }
    txtAreaSonuc.setText(txtSonuc.toString());
}
```

```
private void btnClearActionPerformed(java.awt.event.ActionEvent evt)
{
```

```
    txtAreaSonuc.setText(""); txtSayi1.setText(""); txtSayi2.setText("");
    chkTopla.setSelected(false); chkCikar.setSelected(false);
    chkCarp.setSelected(false); chkBol.setSelected(false);
}
```

## Örnek: RadioButton1

```
public class Form extends javax.swing.JFrame
{
    private javax.swing.JRadioButton rdbBenzin;
    private javax.swing.JRadioButton rdbDizel;
    private javax.swing.ButtonGroup btnGrupArac;
    private javax.swing.JRadioButton rdbManuel;
    private javax.swing.JRadioButton rdbOtomatik;
    private javax.swing.ButtonGroup btnGrupVites;
    private javax.swing.JTextField txtMesafe;
    private javax.swing.JButton btnHesapla;
    private javax.swing.JLabel lblGoster;

    public Form()
    {
        initComponents();
        btnGrupArac.add(rdbBenzin);
        btnGrupArac.add(rdbDizel);
        btnGrupVites.add(rdbManuel);
        btnGrupVites.add(rdbOtomatik);
        rdbBenzin.setSelected(true);
        rdbManuel.setSelected(true);
        lblGoster.setText("");
    }

    private void btnHesaplaActionPerformed(java.awt.event.ActionEvent evt)
    {
        double km = Double.parseDouble(txtMesafe.getText());
        double litre = 0, tutar = 0;

        if (rdbBenzin.isSelected() && rdbManuel.isSelected())
        {
            litre = km * 8 / 100;
            tutar = 10 * litre;
        }
        else if (rdbBenzin.isSelected() && rdbOtomatik.isSelected())
        {
            litre = km * 8 / 100 * 1.05;
            tutar = 10 * litre;
        }
        else if (rdbDizel.isSelected() && rdbManuel.isSelected())
        {
            litre = km * 6 / 100;
            tutar = 9 * litre;
        }
        else if (rdbDizel.isSelected() && rdbOtomatik.isSelected())
        {
            litre = km * 6 / 100 * 1.05;
            tutar = 9 * litre;
        }
        lblGoster.setText(litre + " litre karşılığı " + tutar + " TL");
    }
}
```

Yakıt Hesaplama Ekranı...

Araç Türü: ☐ Benzin ☒ Dizel

Vites Türü: ☒ Manuel ☐ Otomatik

Mesafe(Km):

Hesapla...

6.0 litre karşılığı 54.0 TL

## Örnek: RadioButton2

```
import javax.swing.JOptionPane;
public class Form extends javax.swing.JFrame
{
    private javax.swing.JTextField txtMusteri;
    private javax.swing.JTextField txtFiyat;
    private javax.swing.JTextField txtAdet;
    private javax.swing.JTextField txtİskonto;
    private javax.swing.JTextField txtOdenecek;
    private javax.swing.JRadioButton rdbDiger;
    private javax.swing.JRadioButton rdbGazi;
    private javax.swing.JRadioButton rdbMemur;
    private javax.swing.JRadioButton rdbOgrenci;
    private javax.swing.ButtonGroup btnGrupRezervasyon;

    private int adet, fiyat;
    private double fatura, indirim=0;

    public Form()
    {
        initComponents();
        txtİskonto.setEditable(false);
        txtOdenecek.setEditable(false);
        txtAdet.setText("1");
    }

    private void txtFiyatFocusLost(java.awt.event.FocusEvent evt)
    {
        try
        {
            fiyat = Integer.parseInt(txtFiyat.getText());
            txtAdet.selectAll();
        }
        catch (NumberFormatException e)
        {
            JOptionPane.showMessageDialog(null, "Geçersiz","X",0);
            txtFiyat.requestFocus(); txtFiyat.selectAll();
        }
    }

    private void txtAdetFocusLost(java.awt.event.FocusEvent evt)
    {
        try
        {
            adet = Integer.parseInt(txtAdet.getText());
        }
        catch (NumberFormatException s)
        {
            JOptionPane.showMessageDialog(null, "Geçersiz","X",0);
            txtAdet.requestFocus(); txtAdet.selectAll();
        }
    }
}
```

The screenshot shows a Java Swing window titled "HavaYolu Bilet Rezervasyon...". The window contains a form with two main sections. The left section has labels and text fields for "Müşteri Adı" (Ahmet Demir), "Bilet Fiyatı" (450), "Bilet Adedi" (3), "İskonto" (337.5), and "Ödenecek Tutar" (1012.5). The right section is titled "Rezervasyon Türü..." and contains four radio buttons: "Öğrenci" (selected), "Memur", "Gazi", and "Diğer...".

```

protected void rezervasyon()
{
    fiyat = Integer.parseInt(txtFiyat.getText());
    adet = Integer.parseInt(txtAdet.getText());
    fatura = fiyat * adet;
    if (rdbOgrenci.isSelected()) indirim = fatura * 0.25d;
    else if (rdbMemur.isSelected()) indirim = fatura * (double)0.20;
    else if (rdbGazi.isSelected()) indirim = fatura * 0.15d;
    else indirim=0;
    fatura -= indirim;
    txtIskonto.setText(String.valueOf(indirim));
    txtOdenecek.setText(String.valueOf(fatura));
    buttonGroup1.clearSelection();
}

private void rdbOgrenciActionPerformed(java.awt.event.ActionEvent evt)
{
    rezervasyon();
}

private void rdbMemurActionPerformed(java.awt.event.ActionEvent evt)
{
    rezervasyon();
}

private void rdbGaziActionPerformed(java.awt.event.ActionEvent evt)
{
    rezervasyon();
}

private void rdbDigerActionPerformed(java.awt.event.ActionEvent evt)
{
    rezervasyon();
}

```

### Örnek: RadioButton3

```
public class Form extends javax.swing.JFrame
{
    private javax.swing.JRadioButton rdbJava;
    private javax.swing.JRadioButton rdbPhp;
    private javax.swing.JRadioButton rdbPython;
    private javax.swing.ButtonGroup btnGrupKurs;
    private javax.swing.JCheckBox chkEvet;
    private javax.swing.JRadioButton rdbKdv18;
    private javax.swing.JRadioButton rdbKdv8;
    private javax.swing.ButtonGroup btnGrupKdv;
    private javax.swing.JTextField txtFiyat;
    private javax.swing.JButton btnHesap;
    private javax.swing.JLabel lblSonuc;

    public Form()
    {
        initComponents();
        rdbPhp.setSelected(true);
        rdbKdv8.setSelected(true);
        chkEvet.setSelected(true);
        txtFiyat.requestFocus();
        lblFatura.setVisible(false);
    }

    private void btnHesapActionPerformed(java.awt.event.ActionEvent evt)
    {
        double fatura,kdv=0,oran=0;
        if (txtFiyat.getText().equals(""))
        {
            JOptionPane.showMessageDialog(rootPane, "Kurs Fiyatını Girmelisiniz...");
            return;
        }
        fatura = Double.parseDouble(txtFiyat.getText());
        if (chkEvet.isSelected())
        {
            if (rdbPhp.isSelected()) oran = 20;
            else if (rdbJava.isSelected()) oran= 15;
            else if (rdbPython.isSelected()) oran=10;
            if (rdbKdv8.isSelected()) kdv = 8;
            else kdv = 18;

            fatura-= fatura * oran/100 ;
            fatura+= fatura * kdv / 100;

            lblFatura.setVisible(true);
            lblFatura.setText("Kurs Fiyatı: "+ fatura+ " TL") ;
        }
    }
}
```



## Örnek: Spinner

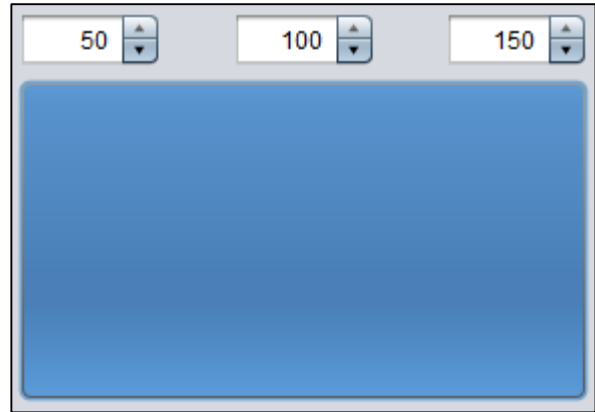
```
import java.awt.Color;

public class Form extends javax.swing.JFrame
{
    private javax.swing.JSpinner spn1Red;
    private javax.swing.JSpinner spn2Blue;
    private javax.swing.JSpinner spn3Green;
    private javax.swing.JButton btnRenk;
    static int r , g , b;
```

```
private void spn1RedStateChanged(javax.swing.event.ChangeEvent evt)
{
    r = (int) spn1Red.getValue();
    btnRenk.setBackground(new Color(r,g,b));
}

private void spn2GreenStateChanged(javax.swing.event.ChangeEvent evt)
{
    g = (int) spn2Green.getValue();
    btnRenk.setBackground(new Color(r,g,b));
}

private void spn3BlueStateChanged(javax.swing.event.ChangeEvent evt)
{
    b = (int) spn3Blue.getValue();
    btnRenk.setBackground(new Color(r,g,b));
}
```



## Örnek:ComboBox1

comboBox.addItem(Eleman);	Eleman ekler.
int getSelectedIndex( )	Seçili elemanın indis numarasını verir.
void setSelectedIndex(int)	Nesnenin başlığını belirler.
Object getSelectedItem( )	Seçili öğenin adını verir.
void removeItem(Object)	Adı belirtilen öğeyi siler.
void removeAllItems( )	Listedeki bütün öğeleri siler.
int getItemCount( )	Listedeki öğelerin sayısını dönderir.
int getMaximumRowCount( )	Pencerede görünen eleman sayısını dönderir.
void setMaximumRowCount(int)	Pencerede görünecek eleman sayısını belirler.
void setEnabled(boolean)	Nesnenin seçilebilme durumunu belirler.

```
public class Form extends javax.swing.JFrame
{
    private javax.swing.JComboBox<String> cmbBolum;
    private javax.swing.JLabel lblGoster;
    private javax.swing.JButton btnGoster;

    public Form()
    {
        initComponents();
        cmbBolum.addItem("Bilgisayar");
        cmbBolum.addItem("Elektrik");
        cmbBolum.addItem("Elektronik");
        cmbBolum.addItem("İnşaat");
    }

    private void btnGosterActionPerformed(java.awt.event.ActionEvent evt)
    {
        lblGoster.setText(String.valueOf(cmbBolum.getSelectedIndex()));
        // cmbBolum.setSelectedIndex(1);
        // lblGoster.setText(cmbBolum.getSelectedItem().toString());
        // cmbBolum.removeItem("Bilgisayar");
        // cmbBolum.removeAllItems();
        // lblGoster.setText(String.valueOf(cmbBolum.getItemCount()));
        // lblGoster.setText(String.valueOf(cmbBolum.getMaximumRowCount()));
        // cmbBolum.setMaximumRowCount(20);
        // cmbBolum.setEnabled(false);
    }
}
```



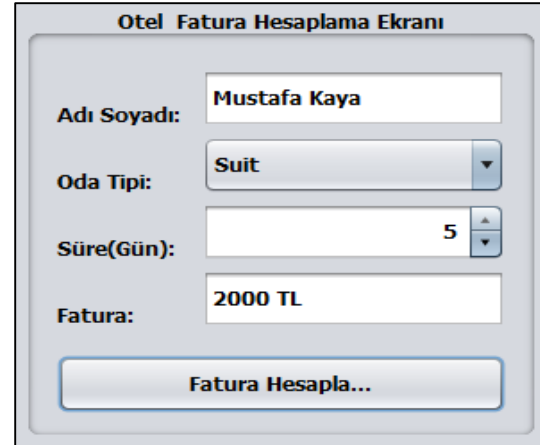
## Örnek: ComboBox2

```
import javax.swing.JOptionPane;

public class Form javax.swing.JFrame
{
    private javax.swing.JTextField txtAd;
    private javax.swing.JComboBox<String> cmbOda;
    private javax.swing.JSpinner spnGun;
    private javax.swing.JTextField txtFatura;
    private javax.swing.JButton btnHesap;

    public Form()
    {
        initComponents();
        cmbOda.addItem("Standard");
        cmbOda.addItem("Suit");
        cmbOda.addItem("Family");
        spnGun.setValue(1);
        txtFatura.setEditable(false);
    }

    private void btnHesapActionPerformed(java.awt.event.ActionEvent evt)
    {
        int fatura = 0;
        switch (cmbOda.getSelectedIndex())
        {
            case 0: fatura = Integer.parseInt(spnGun.getValue().toString()) * 250; break;
            case 1: fatura = Integer.parseInt(spnGun.getValue().toString()) * 400; break;
            case 2: fatura = Integer.parseInt(spnGun.getValue().toString()) * 500; break;
            default: JOptionPane.showMessageDialog(null, "Oda tipini seç...", "X", 0); break;
        }
        txtFatura.setText(fatura+" TL");
    }
}
```



### Örnek: ComboBox3

```
public class Form extends javax.swing.JFrame
{
    private javax.swing.JComboBox<String> cmbYer;
    private javax.swing.JButton btnHesap;
    private javax.swing.JTextField txtMesafe;
    private javax.swing.JSpinner spnHiz;
    private javax.swing.JTextField txtSure;

    public Form()
    {
        initComponents();
        cmbYer.addItem("Güzergahı Seçiniz...");
        cmbYer.addItem("Hatay-Adana");
        cmbYer.addItem("Hatay-Ankara");
        cmbYer.addItem("Hatay-İstanbul");
        cmbYer.addItem("Hatay-İzmir");
        cmbYer.addItem("Hatay-Diyarbakır");
        cmbYer.addItem("Hatay-Trabzon");
    }

    private void cmbYerActionPerformed(java.awt.event.ActionEvent evt)
    {
        txtMesafe.setEditable(false);
        txtSure.setEditable(false);
        spnHiz.setValue(90);
        switch (cmbYer.getSelectedIndex())
        {
            case 1: txtMesafe.setText("190");break;
            case 2: txtMesafe.setText("700");break;
            case 3: txtMesafe.setText("1200");break;
            case 4: txtMesafe.setText("1000");break;
            case 5: txtMesafe.setText("600");break;
            case 6: txtMesafe.setText("1500");break;
        }
    }

    private void btnHesapActionPerformed(java.awt.event.ActionEvent evt)
    {
        double mesafe, hiz,zaman;
        int saat, dakika; String mesaj="";
        mesafe = Double.parseDouble(txtMesafe.getText());
        hiz = Integer.parseInt(spnHiz.getValue().toString());
        zaman = mesafe / hiz*60;
        saat = (int) zaman / 60;
        dakika = (int) zaman % 60;
        if (saat>0) mesaj=saat+" saat ";
        if (dakika>0) mesaj+=dakika+" dakika";
        txtSure.setText(mesaj);
    }
}
```

The screenshot shows a Java Swing window with a light gray background. It is divided into two main panels. The left panel, titled 'Güzergah', contains a JComboBox with 'Hatay-İstanbul' selected and a 'Hesapla...' button. The right panel, titled 'Varış Bilgileri...', contains three input fields: 'Mesafe(km)' with '1200', 'Hız(km/h)' with '90', and 'Süre(h)' with '13 saat 20 dakika'.

## Örnek: ComboBox4

```
public class Form extends javax.swing.JFrame
{
    private javax.swing.ButtonGroup btnGrup;
    private javax.swing.JRadioButton rdbElektronik;
    private javax.swing.JRadioButton rdbGiyim;
    private javax.swing.JRadioButton rdbGida;
    private javax.swing.JComboBox<String> cmbUrun;
    private javax.swing.JSpinner spnAdet;
    private javax.swing.JTextField txtFatura;
    private javax.swing.JTextField txtFiyat;
    private javax.swing.JTextField txtKdv;
    private javax.swing.JButton btnHesap;
    private double adet, fiyat, fatura = 0, kdv = 0;

    public Form()
    {
        initComponents();
        btnGrup.add(rdbElektronik);
        btnGrup.add(rdbGiyim);
        btnGrup.add(rdbGida);
        cmbUrun.removeAllItems();
        cmbUrun.addItem("Lütfen Ürünü Seçiniz...");
        cmbUrun.addItem("Bilgisayar");
        cmbUrun.addItem("Televizyon");
        cmbUrun.addItem("Buzdolabı");
        rdbElektronik.setSelected(true);
        spnAdet.setValue(1);
    }

    private void temizle()
    {
        txtKdv.setText(""); txtFatura.setText("");
        txtFiyat.setText(""); spnAdet.setValue(1);
    }

    private void rdbElektronikActionPerformed(java.awt.event.ActionEvent evt)
    {
        temizle(); cmbUrun.removeAllItems();
        cmbUrun.addItem("Lütfen Ürünü Seçiniz...");
        cmbUrun.addItem("Bilgisayar");
        cmbUrun.addItem("Televizyon");
        cmbUrun.addItem("Buzdolabı");
    }

    private void rdbGiyimActionPerformed(java.awt.event.ActionEvent evt)
    {
        temizle(); cmbUrun.removeAllItems();
        cmbUrun.addItem("Lütfen Ürünü Seçiniz...");
        cmbUrun.addItem("Pantolon");
        cmbUrun.addItem("Gömlek");
        cmbUrun.addItem("Ayakkabı");
    }

    private void rdbGidaActionPerformed(java.awt.event.ActionEvent evt)
    {
        temizle(); cmbUrun.removeAllItems();
        cmbUrun.addItem("Lütfen Ürünü Seçiniz...");
        cmbUrun.addItem("Pirinç");
        cmbUrun.addItem("Şeker");
        cmbUrun.addItem("Çay");
    }
}
```

```

private void cmbUrunActionPerformed(java.awt.event.ActionEvent evt)
{
    temizle();
    if (rdbElektronik.isSelected())
    {
        kdv = 0.25;
        switch (cmbUrun.getSelectedIndex())
        {
            case 1: txtFiyat.setText("5000");break;
            case 2: txtFiyat.setText("3000");break;
            case 3: txtFiyat.setText("4000");break;
        }
    }
    else if (rdbGiyim.isSelected())
    {
        kdv = 0.18;
        switch (cmbUrun.getSelectedIndex())
        {
            case 1: txtFiyat.setText("500");break;
            case 2: txtFiyat.setText("300");break;
            case 3: txtFiyat.setText("400");break;
        }
    }
    else
    {
        kdv = 0.08;
        switch (cmbUrun.getSelectedIndex())
        {
            case 1: txtFiyat.setText("50");break;
            case 2: txtFiyat.setText("30");break;
            case 3: txtFiyat.setText("40");break;
        }
    }
}

private void btnHesapActionPerformed(java.awt.event.ActionEvent evt)
{
    adet = Double.parseDouble(spAdet.getValue().toString());
    fiyat = Double.parseDouble(txtFiyat.getText());
    fatura = fiyat * adet;
    kdv *= fatura;
    fatura += kdv;
    txtKdv.setText(String.valueOf(kdv));
    txtFatura.setText(String.valueOf(fatura));
}

```

## Örnek:ListBox1

```
import java.util.ArrayList;
import java.util.Random;
import javax.swing.DefaultListModel;

public class Form extends javax.swing.JFrame
{
    private javax.swing.JList<String> lstMac;
    private javax.swing.JList<String> lstHakem;
    private javax.swing.JButton btnHakem;

    public Form()
    {
        initComponents();
        DefaultListModel dlmMac=new DefaultListModel();
        dlmMac.addElement("Fenerbahçe-Antalya");
        dlmMac.addElement("Sivas-Galatasaray");
        dlmMac.addElement("Beşiktaş-Konya");
        dlmMac.addElement("Kasımpaşa-Trabzon");
        dlmMac.addElement("Kayseri-Bursa");
        lstMac.setModel(dlmMac);
    }

    private void btnHakemActionPerformed(java.awt.event.ActionEvent evt)
    {
        String[] hakemler = { "Cüneyt Çakır", "Halis Tokat", "Fırat Demir", "Hüseyin Göçek",
            "Yunus Ay", "Demir Yılmaz","Ali Yavuz","Serkan Çınar"," Mete Tok","Koray Gencer"};

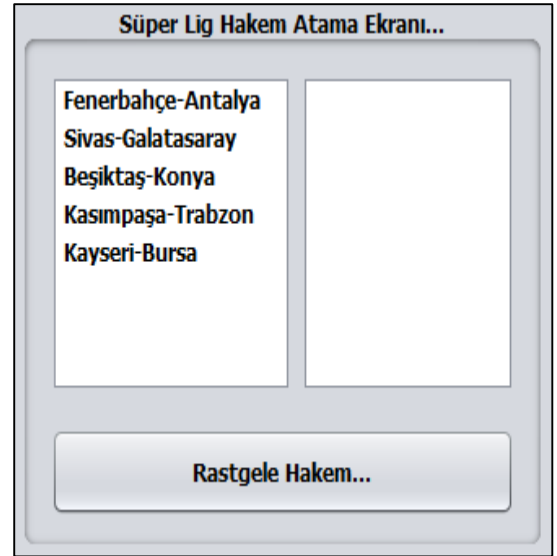
        Random r=new Random();
        int uret, sayac=0;

        ArrayList<Integer> maclar=new ArrayList<Integer>();

        DefaultListModel dlmHakem=new DefaultListModel();

        while(sayac<5)
        {
            {
                uret=r.nextInt(10);
                if (maclar.indexOf(uret) == -1)                //dizinin içinde aynı rakam yoksa
                {
                    maclar.add(uret);                        //Rastgele üretilen rakam sıradaki takıma atanacak
                    sayac++;
                }
            }
        }
        dlmHakem.clear();
        lstHakem.setModel(dlmHakem);
        for (int indis:maclar)                // Atanan hakem listesi
            dlmHakem.addElement(hakemler[indis]);

        lstHakem.setModel(dlmHakem);
    }
}
```



## Örnek: ListBox2

```
import java.awt.Color;
import java.awt.Component;
import javax.swing.DefaultListCellRenderer;
import javax.swing.DefaultListModel;
import javax.swing.JList;
import javax.swing.JOptionPane;

public class Form extends javax.swing.JFrame
{
    private javax.swing.JTextField txtInput;
    private javax.swing.JButton btnAdd;
    private javax.swing.JButton btnUpdate;
    private javax.swing.JButton btnDelete;
    private javax.swing.JButton btnRight;
    private javax.swing.JButton btnRightAll;
    private javax.swing.JButton btnLeft;
    private javax.swing.JButton btnLeftAll;
    private javax.swing.JButton btnUp;
    private javax.swing.JButton btnDown;
    private javax.swing.JList<String> lstLeft;
    private javax.swing.JList<String> lstRight;

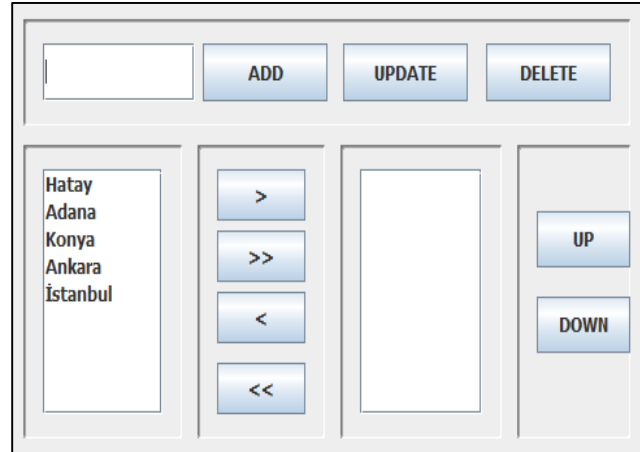
    private DefaultListModel modelLeft=new DefaultListModel();
    private DefaultListModel modelRight=new DefaultListModel();
    private int index;
    private [] secilen;

    public Form()
    {
        initComponents();
        modelLeft.addElement("HATAY");
        modelLeft.addElement("ADANA");
        modelLeft.add(modelLeft.getSize(), "MERSİN");
        listLeft.setModel(modelLeft);

        modelRight.addElement("ANKARA");
        modelRight.addElement("İSTANBUL");
        modelRight.add(0, "İZMİR");
        listRight.setModel(modelRight);

        listLeft.setSelectedIndex(modelLeft.getSize()-1);
        listRight.setSelectedIndex(modelRight.getSize()-1);
    }

    private void listLeftMouseClicked(java.awt.event.MouseEvent evt)
    {
        txtInput.setText(listLeft.getSelectedValue());
    }
}
```





```

private void btnAddActionPerformed(java.awt.event.ActionEvent evt)
{
    if (txtInput.getText().equals(""))
    {
        JOptionPane.showMessageDialog(null,"Boş Geçilemez...");
        txtInput.requestFocus(); return;
    }
    index=modelLeft.indexOf(txtInput.getText().trim().toUpperCase());
    if (index>-1)
    {
        listLeft.setSelectedIndex(index);
        JOptionPane.showMessageDialog(null,"Kayıt Var...");
        txtInput.setText(null); txtInput.requestFocus(); return;
    }
    modelLeft.addElement(txtInput.getText().trim().toUpperCase());
    listLeft.setModel(modelLeft);
    listLeft.setSelectedIndex(modelLeft.getSize()-1);
    txtInput.setText(null); txtInput.requestFocus();
}

private void btnUpdateActionPerformed(java.awt.event.ActionEvent evt)
{
    index=modelLeft.indexOf(txtInput.getText().toUpperCase());
    if (index>-1)
    {
        listLeft.setSelectedIndex(index);
        JOptionPane.showMessageDialog(null,"Kayıt Var..."); return;
    }
    index=listLeft.getSelectedIndex();
    modelLeft.setElementAt (txtInput.getText().trim().toUpperCase(), index);
    txtInput.setText(""); txtInput.requestFocus();
}

private void btnDeleteActionPerformed(java.awt.event.ActionEvent evt)
{
    if (listLeft.getSelectedIndex()>-1) modelLeft.remove(listLeft.getSelectedIndex());
    /* Çoklu Kayıt Silme
    secilen=listLeft.getSelectedIndices();
    if (secilen.length > 0)
    {
        for (int i = secilen.length - 1; i >= 0; i--)
            modelLeft.removeElementAt(secilen[i]);
    }
    */
    listLeft.setSelectedIndex(modelLeft.getSize()-1);
}

```

```

private void btnRightActionPerformed(java.awt.event.ActionEvent evt)
{
    secilen=listLeft.getSelectedIndices();
    if (secilen.length > 0)
    {
        for (int i = 0; i < secilen.length; i++)
            modelRight.addElement (listLeft.getModel().getElementAt(secilen[i]));

        for (int i = secilen.length - 1; i >= 0; i--)
            modelLeft.removeElementAt(secilen[i]);
    }
    listLeft.setSelectedIndex(modelLeft.getSize()-1);
    listRight.setSelectedIndex(modelRight.getSize()-1);
}

private void btnRightAllActionPerformed(java.awt.event.ActionEvent evt)
{
    for (int i = 0; i < modelLeft.getSize(); i++)
        modelRight.addElement(modelLeft.elementAt(i));

    modelLeft.removeAllElements();
    listRight.setSelectedIndex(modelRight.getSize()-1);
}

private void btnLeftActionPerformed(java.awt.event.ActionEvent evt)
{
    secilen=listRight.getSelectedIndices();
    if (secilen.length > 0)
    {
        for (int i = 0; i < secilen.length; i++)
            modelLeft.addElement (listRight.getModel().getElementAt(secilen[i]));

        for (int i = secilen.length - 1; i >= 0; i--)
            modelRight.removeElementAt(secilen[i]);
    }
    listLeft.setSelectedIndex(modelLeft.getSize()-1);
    listRight.setSelectedIndex(modelRight.getSize()-1);
}

private void btnLeftAllActionPerformed(java.awt.event.ActionEvent evt)
{
    for (int i = 0; i < modelRight.getSize(); i++)
        modelLeft.addElement(modelRight.elementAt(i));

    modelRight.removeAllElements();
    listLeft.setSelectedIndex(modelLeft.getSize()-1);
}

```

```

private void btnUpActionPerformed(java.awt.event.ActionEvent evt)
{
    index = listRight.getSelectedIndex();
    if (index == -1) JOptionPane.showMessageDialog(null, "Kayıt Seç...");
    else if (index > 0)
    {
        String temp = (String) modelRight.remove(index);
        modelRight.add(index-1, temp);
        listRight.setSelectedIndex(index - 1);
    }
}

private void btnDownActionPerformed(java.awt.event.ActionEvent evt)
{
    index = listRight.getSelectedIndex();
    if (index == -1) JOptionPane.showMessageDialog(null, "Kayıt Seç...");
    else if (index < modelRight.size() - 1)
    {
        String temp = (String) modelRight.remove(index);
        modelRight.add(index+1, temp);
        listRight.setSelectedIndex(index + 1);
    }
}

class SelectedListCellRenderer extends DefaultListCellRenderer
{
    @Override
    public Component getListCellRendererComponent (JList list, Object value, int index, boolean isSelected, boolean cellHasFocus)
    {
        Component c = super.getListCellRendererComponent (list, value, index, isSelected, cellHasFocus);
        if (isSelected) c.setBackground(Color.ORANGE);
        return c;
    }
}

private void listLeftValueChanged(javax.swing.event.ListSelectionEvent evt)
{
    listLeft.setCellRenderer(new SelectedListCellRenderer());
}

private void listRightValueChanged(javax.swing.event.ListSelectionEvent evt)
{
    listRight.setCellRenderer(new SelectedListCellRenderer());
}

```

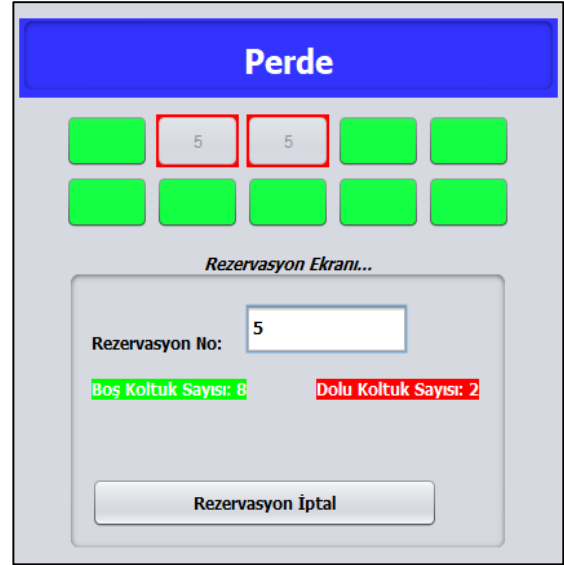
## Uygulama1:

```
import javax.swing.JOptionPane;
public class Form extends javax.swing.JFrame
{
    private javax.swing.JPanel pnlPerde;
    private javax.swing.JButton btn1;
    private javax.swing.JButton btn2;
    private javax.swing.JButton btn3;
    private javax.swing.JButton btn4;
    private javax.swing.JButton btn5;
    private javax.swing.JButton btn6;
    private javax.swing.JButton btn7;
    private javax.swing.JButton btn8;
    private javax.swing.JButton btn9;
    private javax.swing.JButton btn10;
    private javax.swing.JPanel pnlRezervasyon;
    private javax.swing.JLabel lblRezervasyon;
    private javax.swing.JTextField txtRezervasyon;
    private javax.swing.JLabel lblBos;
    private javax.swing.JLabel lblDolu;
    private javax.swing.JButton btnIptal;

    private int bos=10 , dolu=0;

    public Form()
    {
        initComponents();
        lblBos.setForeground(Color.WHITE);
        lblBos.setBackground(Color.GREEN);
        lblBos.setOpaque(true);
        lblBos.setText("Boş Koltuk Sayısı: "+bos);
        lblDolu.setForeground(Color.WHITE);
        lblDolu.setBackground(Color.RED);
        lblDolu.setOpaque(true);
        lblDolu.setText("Dolu Koltuk Sayısı: "+dolu);
        txtRezervasyon.requestFocus();
    }

    protected void koltukRezervasyon(javax.swing.JButton btn)
    {
        {
            if (!txtRezervasyon.getText().equals(""))
            {
                btn.setEnabled(false); btn.setText(txtRezervasyon.getText());
                btn.setBackground(Color.RED); btn.setOpaque(true);
                bos++; dolu--;
                lblBos.setText("Boş Koltuk Sayısı: " + bos);
                lblDolu.setText("Dolu Koltuk Sayısı: " + dolu);
                txtRezervasyon.requestFocus();
            }
            else
            {
                JOptionPane.showMessageDialog(null, "Rezervasyon Boş Geçilemez...", "", 0);
                txtRezervasyon.requestFocus();
            }
        }
    }
}
```



```

protected void koltukIptal(javax.swing.JButton btn)
{
    btn.setText("");
    btn.setBackground(Color.GREEN);
    btn.setOpaque(true);
    btn.setEnabled(true);
    bos++; dolu--;
    lblBos.setText("Boş Koltuk Sayısı: " + bos);
    lblDolu.setText("Dolu Koltuk Sayısı: " + dolu);
}

private void btn1ActionPerformed(java.awt.event.ActionEvent evt)
{ koltukRezervasyon(btn1); }

private void btn2ActionPerformed(java.awt.event.ActionEvent evt)
{ koltukRezervasyon(btn2); }

private void btn3ActionPerformed(java.awt.event.ActionEvent evt)
{ koltukRezervasyon(btn3); }

private void btn4ActionPerformed(java.awt.event.ActionEvent evt)
{ koltukRezervasyon(btn4); }

private void btn5ActionPerformed(java.awt.event.ActionEvent evt)
{ koltukRezervasyon(btn5); }

private void btn6ActionPerformed(java.awt.event.ActionEvent evt)
{ koltukRezervasyon(btn6); }

private void btn7ActionPerformed(java.awt.event.ActionEvent evt)
{ koltukRezervasyon(btn7); }

private void btn8ActionPerformed(java.awt.event.ActionEvent evt)
{ koltukRezervasyon(btn8); }

private void btn9ActionPerformed(java.awt.event.ActionEvent evt)
{ koltukRezervasyon(btn9); }

private void btn10ActionPerformed(java.awt.event.ActionEvent evt)
{ koltukRezervasyon(btn10); }

private void btnIptalActionPerformed(java.awt.event.ActionEvent evt)
{
    if (txtRezervasyon.getText().equals(btn1.getText())) koltukIptal(btn1);
    if (txtRezervasyon.getText().equals(btn2.getText())) koltukIptal(btn2);
    if (txtRezervasyon.getText().equals(btn3.getText())) koltukIptal(btn3);
    if (txtRezervasyon.getText().equals(btn4.getText())) koltukIptal(btn4);
    if (txtRezervasyon.getText().equals(btn5.getText())) koltukIptal(btn5);
    if (txtRezervasyon.getText().equals(btn6.getText())) koltukIptal(btn6);
    if (txtRezervasyon.getText().equals(btn7.getText())) koltukIptal(btn7);
    if (txtRezervasyon.getText().equals(btn8.getText())) koltukIptal(btn8);
    if (txtRezervasyon.getText().equals(btn9.getText())) koltukIptal(btn9);
    if (txtRezervasyon.getText().equals(btn10.getText())) koltukIptal(btn10);
}

```

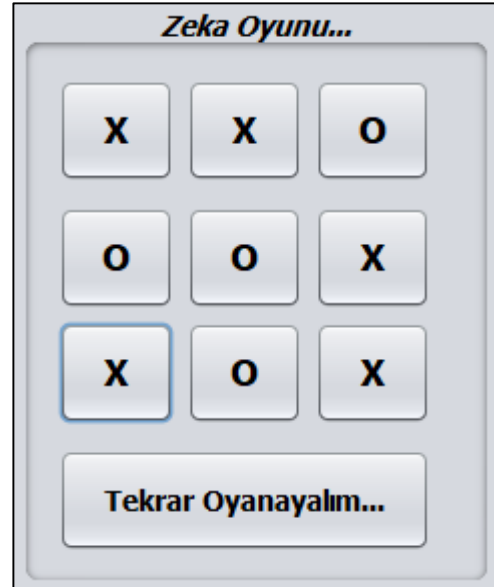
## Uygulama2:

```
import javax.swing.JOptionPane;
public class Form extends javax.swing.JFrame
{
    private javax.swing.JButton btn1;
    private javax.swing.JButton btn2;
    private javax.swing.JButton btn3;
    private javax.swing.JButton btn4;
    private javax.swing.JButton btn5;
    private javax.swing.JButton btn6;
    private javax.swing.JButton btn7;
    private javax.swing.JButton btn8;
    private javax.swing.JButton btn9;
    private javax.swing.JButton btnTekrar;
    private int [ ] button = {0,0,0,0,0,0,0,0,0};
    private int [ ] p1 = {0,0,0,0,0,0,0,0,0};
    private int [ ] p2 = {0,0,0,0,0,0,0,0,0};
    private int hamle=0;

    public int player1()
    {
        if (p1[1]==1 && p1[2]==1 && p1[3]==1) return 1;
        if (p1[4]==1 && p1[5]==1 && p1[6]==1) return 1;
        if (p1[7]==1 && p1[8]==1 && p1[9]==1) return 1;
        if (p1[1]==1 && p1[4]==1 && p1[7]==1) return 1;
        if (p1[2]==1 && p1[5]==1 && p1[8]==1) return 1;
        if (p1[3]==1 && p1[6]==1 && p1[9]==1) return 1;
        if (p1[1]==1 && p1[5]==1 && p1[9]==1) return 1;
        if (p1[3]==1 && p1[5]==1 && p1[7]==1) return 1;
        return 0;
    }

    public int player2()
    {
        if (p2[1]==1 && p2[2]==1 && p2[3]==1) return 1;
        if (p2[4]==1 && p2[5]==1 && p2[6]==1) return 1;
        if (p2[7]==1 && p2[8]==1 && p2[9]==1) return 1;
        if (p2[1]==1 && p2[4]==1 && p2[7]==1) return 1;
        if (p2[2]==1 && p2[5]==1 && p2[8]==1) return 1;
        if (p2[3]==1 && p2[6]==1 && p2[9]==1) return 1;
        if (p2[1]==1 && p2[5]==1 && p2[9]==1) return 1;
        if (p2[3]==1 && p2[5]==1 && p2[7]==1) return 1;
        return 0;
    }

    public void sec(javax.swing.JButton btn,int indis)
    {
        if (button[indis]==0)
        {
            if (hamle%2==0)
            {
                hamle++; btn.setText("X"); button[indis]=1; p1[indis]=1;
                if (player1()==1) JOptionPane.showMessageDialog(null, "Birinci Oyuncu Kazandı");
                else if (player2()==1) JOptionPane.showMessageDialog(null, "İkinci Oyuncu Kazandı");
                else if (hamle==9) JOptionPane.showMessageDialog(null, "Oyun Berabere...");
            }
            else
            {
                hamle++; btn.setText("O"); button[indis]=1; p2[indis]=1;
                if (player1()==1) JOptionPane.showMessageDialog(null, "Birinci Oyuncu Kazandı");
                else if (player2()==1) JOptionPane.showMessageDialog(null, "İkinci Oyuncu Kazandı");
                else if (hamle==9) JOptionPane.showMessageDialog(null, "Oyun Berabere...");
            }
        }
        else JOptionPane.showMessageDialog(null, "Zaten Seçili...");
    }
}
```



```

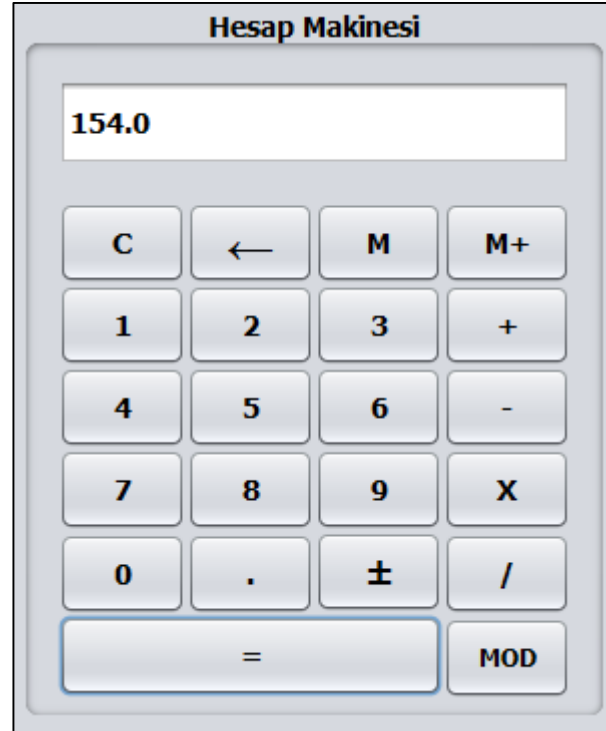
public Form()
{
    btn1.addActionListener(new ActionListener()
    {
        sec(btn1, 1);
    }
    btn2.addActionListener(new ActionListener()
    {
        sec(btn2, 2);
    }
    btn3.addActionListener(new ActionListener()
    {
        sec(btn3, 3);
    }
    btn4.addActionListener(new ActionListener()
    {
        sec(btn4, 4);
    }
    btn5.addActionListener(new ActionListener()
    {
        sec(btn5, 5);
    }
    btn6.addActionListener(new ActionListener()
    {
        sec(btn6, 6);
    }
    btn7.addActionListener(new ActionListener()
    {
        sec(btn7, 7);
    }
    btn8.addActionListener(new ActionListener()
    {
        sec(btn8, 8);
    }
    btn9.addActionListener(new ActionListener()
    {
        sec(btn9, 9);
    }

    btnTekrar.addActionListener(new ActionListener()
    {
        btn1.setText("");
        btn2.setText("");
        btn3.setText("");
        btn4.setText("");
        btn5.setText("");
        btn6.setText("");
        btn7.setText("");
        btn8.setText("");
        btn9.setText("");
        hamle=0;
        for(int i=0;i<10;i++)
        {
            button[i]=0;
            p1[i]=0;
            p2[i]=0;
        }
    }
}

```

### Uygulama3:

```
public class form extends javax.swing.JFrame
{
    private javax.swing.JTextField txtEkran;
    private javax.swing.JButton btnC;
    private javax.swing.JButton btnBackSpace;
    private javax.swing.JButton btnM1;
    private javax.swing.JButton btnM2;
    private javax.swing.JButton btn1;
    private javax.swing.JButton btn2;
    private javax.swing.JButton btn3;
    private javax.swing.JButton btnTopla;
    private javax.swing.JButton btn4;
    private javax.swing.JButton btn5;
    private javax.swing.JButton btn6;
    private javax.swing.JButton btnCikar;
    private javax.swing.JButton btn7;
    private javax.swing.JButton btn8;
    private javax.swing.JButton btn9;
    private javax.swing.JButton btnCarp;
    private javax.swing.JButton btn0;
    private javax.swing.JButton btnNokta;
    private javax.swing.JButton btnEksi;
    private javax.swing.JButton btnBol;
    private javax.swing.JButton btnEsit;
    private javax.swing.JButton btnMod;
```



// Ara/Karakter Eşlem/MS Gothic/BackSpase ve Eksi ekle

```
double s1, s2, m;
String islem;
```

```
public Form()
```

```
{
    initComponents();
    txtEkran.setEditable(false);
}
```

```
private void btn1ActionPerformed(java.awt.event.ActionEvent evt)
{
    txtEkran.setText(txtEkran.getText()+btn1.getText());
}
```

```
private void btn2ActionPerformed(java.awt.event.ActionEvent evt)
{
    txtEkran.setText(txtEkran.getText()+btn2.getText());
}
```

```
private void btn3ActionPerformed(java.awt.event.ActionEvent evt)
{
    txtEkran.setText(txtEkran.getText()+btn3.getText());
}
```

```
private void btn4ActionPerformed(java.awt.event.ActionEvent evt)
{
    txtEkran.setText(txtEkran.getText()+btn4.getText());
}
```

```
private void btn5ActionPerformed(java.awt.event.ActionEvent evt)
{
    txtEkran.setText(txtEkran.getText()+btn5.getText());
}
```



```

private void btn6ActionPerformed(java.awt.event.ActionEvent evt)
{
    txtEkran.setText(txtEkran.getText()+btn6.getText());
}

private void btn7ActionPerformed(java.awt.event.ActionEvent evt)
{
    txtEkran.setText(txtEkran.getText()+btn7.getText());
}

private void btn8ActionPerformed(java.awt.event.ActionEvent evt)
{
    txtEkran.setText(txtEkran.getText()+btn8.getText());
}

private void btn9ActionPerformed(java.awt.event.ActionEvent evt)
{
    txtEkran.setText(txtEkran.getText()+btn9.getText());
}

private void btn0ActionPerformed(java.awt.event.ActionEvent evt)
{
    txtEkran.setText(txtEkran.getText()+btn0.getText());
}

private void btnNoktaActionPerformed(java.awt.event.ActionEvent evt)
{
    txtEkran.setText(txtEkran.getText()+btnNokta.getText());
}

private void btnEksiActionPerformed(java.awt.event.ActionEvent evt)
{
    s1=Double.parseDouble(String.valueOf(txtEkran.getText()));
    s1=s1*(-1); txtEkran.setText(String.valueOf(s1));
}

private void btnCActionPerformed(java.awt.event.ActionEvent evt)
{
    txtEkran.setText("");
}

private void btnBackSpaceActionPerformed(java.awt.event.ActionEvent evt)
{
    String sil=null;
    if (txtEkran.getText().length()>0)
    {
        StringBuilder sb=new StringBuilder(txtEkran.getText());
        sb.deleteCharAt(txtEkran.getText().length()-1);
        sil=sb.toString(); txtEkran.setText(sil);
    }
}

private void btnM1ActionPerformed(java.awt.event.ActionEvent evt)
{
    m=Double.parseDouble(txtEkran.getText());
    txtEkran.setText("");
}

```

```

private void btnM2ActionPerformed(java.awt.event.ActionEvent evt)
{
    txtEkran.setText(Double.toString(m));
}

private void btnToplaActionPerformed(java.awt.event.ActionEvent evt)
{
    s1=Double.parseDouble(txtEkran.getText());
    islem="+"; txtEkran.setText("");
}

private void btnCikarActionPerformed(java.awt.event.ActionEvent evt)
{
    s1=Double.parseDouble(txtEkran.getText());
    islem="-"; txtEkran.setText("");
}

private void btnCarpActionPerformed(java.awt.event.ActionEvent evt)
{
    s1=Double.parseDouble(txtEkran.getText());
    islem="*"; txtEkran.setText("");
}

private void btnBolActionPerformed(java.awt.event.ActionEvent evt)
{
    s1=Double.parseDouble(txtEkran.getText());
    islem="/"; txtEkran.setText("");
}

private void btnModActionPerformed(java.awt.event.ActionEvent evt)
{
    s1=Double.parseDouble(txtEkran.getText());
    islem="%"; txtEkran.setText("");
}

private void btnEsitActionPerformed(java.awt.event.ActionEvent evt)
{
    double sonuc=0;
    s2=Double.parseDouble(txtEkran.getText());
    switch (islem)
    {
        case "+" : sonuc=s1+s2; break;
        case "-" : sonuc=s1-s2; break;
        case "*" : sonuc=s1*s2; break;
        case "/" : sonuc=s1/s2; break;
        case "%" : sonuc=s1%s2; break;
    }

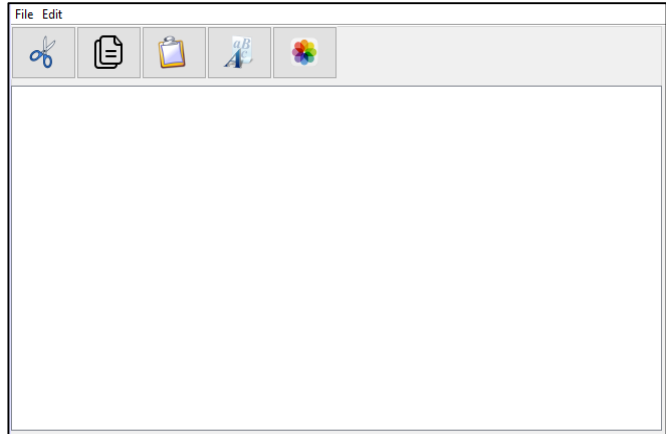
    txtEkran.setText(String.valueOf(sonuc));
    //txtEkran.setText(String.format("%.2f", sonuc));
}

```

## Uygulama4:

```
import com.ozten.font.JFontChooser;
import javax.swing.JColorChooser;
import java.awt.Color;
import javax.swing.JOptionPane;
import java.awt.FileDialog;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.BufferedReader;
import java.io.StringReader;
import java.io.FileOutputStream;
import java.io.DataOutputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.awt.event.KeyEvent;
```

```
public class Form extends javax.swing.JFrame
{
    private javax.swing.JMenuBar MenuBar;
    private javax.swing.JMenu fileMenu;
    private javax.swing.JMenu editMenu;
    private javax.swing.JMenuItem newMenu;
    private javax.swing.JMenuItem openMenu;
    private javax.swing.JMenuItem saveMenu;
    private javax.swing.JMenuItem saveAsMenu;
    private javax.swing.JMenuItem exitMenu;
    private javax.swing.JMenuItem cutMenu;
    private javax.swing.JMenuItem copyMenu;
    private javax.swing.JMenuItem pasteMenu;
    private javax.swing.JMenuItem fontMenu;
    private javax.swing.JMenuItem colorMenu;
    private javax.swing.JButton btnCut;
    private javax.swing.JButton btnCopy;
    private javax.swing.JButton btnPaste;
    private javax.swing.JButton btnFont;
    private javax.swing.JButton btnColor;
    private javax.swing.JPopupMenu popupMenu;
    private javax.swing.JMenuItem cutPopup;
    private javax.swing.JMenuItem copyPopup;
    private javax.swing.JMenuItem pastePopup;
    private javax.swing.JMenuItem fontPopup;
    private javax.swing.JMenuItem colorPopup;
    private javax.swing.JTextArea txtArea;
```



```

private String program="myNotePad";
private String path="";
private String metin;
private String dosya;
private String klasor;
private Color ch;
private boolean textChanged=false;

public Form()
{
    initComponents();
    btnPaste.setEnabled(false);
    pasteMenu.setEnabled(false);
    pastePopup.setEnabled(false);
}

private void save(String dosya)
{
    setTitle(program+" "+path);
    try
    {
        FileWriter out=new FileWriter(dosya);
        out.write(txtArea.getText());
        out.close();
        JOptionPane.showMessageDialog(null, "Kaydedildi");
    }
    catch(Exception err)
    {
        JOptionPane.showMessageDialog(null, "Hata:"+err);
    }
    textChanged=false;
    saveMenu.setEnabled(false);
}

```

```

private void saveAs()
{
    FileDialog fd=new FileDialog(Form.this, "Farklı Kaydet...",FileDialog.SAVE);
    fd.show();
    if (fd.getFile()!=null)
    {
        dosya=fd.getFile();
        klasor=fd.getDirectory();
        path=klasor+dosya;
        setTitle(path);
        try
        {
            DataOutputStream d= new DataOutputStream(new FileOutputStream(path));
            metin=txtArea.getText();
            BufferedReader br= new BufferedReader(new StringReader(metin));
            while((metin=br.readLine())!=null)
            {
                d.writeBytes(metin+"\r\n");
            }
            d.close();
        }
        catch(Exception e)
        {
            JOptionPane.showMessageDialog(null, "Dosya Yok...");
        }
        txtArea.requestFocus();
        save(path);
    }
}

public void checkFile()
{
    BufferedReader read;
    StringBuffer sb=new StringBuffer();
    try
    {
        String line;
        read=new BufferedReader(new FileReader(path));
        while ((line=read.readLine()) !=null)
        {
            sb.append(line+"\n");
        }
        read.close();
        txtArea.setText(sb.toString());
    }
    catch(FileNotFoundException e)
    {
        JOptionPane.showMessageDialog(null, "Dosya Yok..");
    }
    catch(IOException ioe)
    {
    }
}

```

```

public void openFile()
{
    FileDialog fd=new FileDialog(Form.this, "Dosya Seç...",FileDialog.LOAD);
    fd.show();
    if (fd.getFile()!=null)
    {
        path=fd.getDirectory()+fd.getFile();
        setTitle(path);
        checkFile();
    }
    txtArea.requestFocus();
}

public void newFile()
{
    setTitle(program);
    path="";
    txtArea.setText("");
    textChanged=false;
}

private void txtAreaKeyReleased(java.awt.event.KeyEvent evt)
{
    if (KeyEvent.KEY_RELEASED!=0)
    {
        if (!textChanged) setTitle("* "+getTitle());
        textChanged=true;
        saveMenu.setEnabled(true);
    }
}

private void saveMenuActionPerformed(java.awt.event.ActionEvent evt)
{
    if (path.equals("")) saveAs(); else save(path);
}

private void saveAsMenuActionPerformed(java.awt.event.ActionEvent evt)
{
    saveAs();
}

```

```

private void formWindowClosing(java.awt.event.WindowEvent evt)
{
    if ("".equals(txtArea.getText())) System.exit(0);
    else if (!textChanged) System.exit(0);
    else
    {
        int confirm=JOptionPane.showConfirmDialog (null, " Kaydedilsinmi?", "X", 0);
        if (confirm==JOptionPane.YES_OPTION)
        {
            if ("".equals(path)) saveAs(); else save(path);
        }
        else if (confirm==JOptionPane.NO_OPTION) System.exit(0);
    }
}

private void openMenuActionPerformed(java.awt.event.ActionEvent evt)
{
    if (txtArea.getText().length()<1) openFile();
    else if (!textChanged) openFile();
    else
    {
        int confirm=JOptionPane.showConfirmDialog(null, "Kaydedilsinmi?", "X", 0);
        if (confirm==JOptionPane.YES_OPTION)
        {
            if ("".equals(path)) saveAs();
            else save(path);
            openFile();
        }
        else if (confirm==JOptionPane.NO_OPTION) openFile();
    }
}

```

```

private void newMenuActionPerformed(java.awt.event.ActionEvent evt)
{
    if (txtArea.getText().length()<1)
    {
        setTitle("Untitled-"+program);
        txtArea.setText("");
        textChanged=false;
    }
    else if(!textChanged)
    {
        setTitle("Untitled-"+program);
        txtArea.setText("");
        textChanged=false;
    }
    else
    {
        int confirm=JOptionPane.showConfirmDialog(null, "Kaydet?", "X",0);
        if (confirm==JOptionPane.YES_OPTION)
        {
            if ("".equals(path)) saveAs(); else save(path);
            newFile();
        }
        if (confirm==JOptionPane.NO_OPTION) newFile();
    }
}

private void exitMenuActionPerformed(java.awt.event.ActionEvent evt)
{
    if ("".equals(txtArea.getText())) System.exit(0);
    else if (!textChanged) System.exit(0);
    else
    {
        int confirm=JOptionPane.showConfirmDialog(null, "Kaydedilsinmi?", "X",0);
        if (confirm==JOptionPane.YES_OPTION)
        {
            if ("".equals(path)) saveAs(); else save(path);
            System.exit(0);
        }
        else if (confirm==JOptionPane.NO_OPTION) System.exit(0);
    }
}

```



```

private void cutMenuActionPerformed(java.awt.event.ActionEvent evt)
{
    txtArea.cut();
    pasteMenu.setEnabled(true);
}

private void copyMenuActionPerformed(java.awt.event.ActionEvent evt)
{
    txtArea.copy();
    pasteMenu.setEnabled(true);
}

private void pasteMenuActionPerformed(java.awt.event.ActionEvent evt)
{
    txtArea.paste();
}

private void fontDialog()
{
    JFontChooser fc=new JFontChooser();
    JOptionPane.showMessageDialog(null, fc, "Bir Font Seç", -1);
    txtArea.setFont(fc.getPreviewFont());
}

private void colorDialog()
{
    ch = JColorChooser.showDialog(null,"Bir Renk Seç",ch);
    txtArea.setForeground(ch);
}

private void fontMenuActionPerformed(java.awt.event.ActionEvent evt)
{
    fontDialog();
}

private void colorMenuActionPerformed(java.awt.event.ActionEvent evt)
{
    colorDialog();
}

```

```

private void btnCutActionPerformed(java.awt.event.ActionEvent evt)
{
    txtArea.cut();
    btnPaste.setEnabled(true);
}

private void btnCopyActionPerformed(java.awt.event.ActionEvent evt)
{
    txtArea.copy();
    btnPaste.setEnabled(true);
}

private void btnPasteActionPerformed(java.awt.event.ActionEvent evt)
{
    txtArea.paste();
}

private void btnFontActionPerformed(java.awt.event.ActionEvent evt)
{
    fontDialog();
}

private void btnColorActionPerformed(java.awt.event.ActionEvent evt)
{
    colorDialog();
}

private void cutPopupActionPerformed(java.awt.event.ActionEvent evt)
{
    txtArea.cut();
    pastePopup.setEnabled(true);
}

private void copyPopupActionPerformed(java.awt.event.ActionEvent evt)
{
    txtArea.copy();
    pastePopup.setEnabled(true);
}

private void pastePopupActionPerformed(java.awt.event.ActionEvent evt)
{
    txtArea.paste();
}

private void fontPopupActionPerformed(java.awt.event.ActionEvent evt)
{
    fontDialog();
}

private void colorPopupActionPerformed(java.awt.event.ActionEvent evt)
{
    colorDialog();
}

```

## Uygulama5:

**Bilgi Girişi...**

Öğrenci No:

Adı:

Soyadı:

Adresi:

**Programlar**

- Teknik Programlar
  - Bilgisayar
  - Elektrik
  - Elektronik
  - İnşaat
- Sosyal Programlar
  - Muhasebe
  - Pazarlama
  - Lojistik
  - Harita

**Harita**

No	Ad	Soyadı	Adres	Bölüm
1	Mustafa	Demir	Ankara	Muhasebe
2	Emine	Kara	Adana	Bilgisayar
3	Ahmet	Güneş	Hatay	İnşaat
4	Emre	Yarar	İstanbul	Lojistik
5	Fatma	Yıldırım	İzmir	Harita

```
import java.awt.event.MouseEvent;
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import javax.swing.JOptionPane;
import javax.swing.JTextField;
import javax.swing.RowFilter;
import javax.swing.table.DefaultTableModel;
import javax.swing.table.TableModel;
import javax.swing.table.TableRowSorter;
import javax.swing.tree.DefaultMutableTreeNode;
import javax.swing.tree.DefaultTreeModel;

public class Form extends javax.swing.JFrame
{
    private javax.swing.JPanel pnlGiris;
    private javax.swing.JSpinner spnNumara;
    private javax.swing.JTextField txtAd;
    private javax.swing.JTextField txtSoyad;
    private javax.swing.JComboBox<String> cmbAdres;
    private javax.swing.JButton btnNew;
    private javax.swing.JButton btnAdd;
    private javax.swing.JButton btnUpdate;
    private javax.swing.JButton btnDelete;
    private javax.swing.JButton btnDeleteAll;
    private javax.swing.JTextField txtFilter;
    private javax.swing.JTree treeBolum;
    private javax.swing.JTextField txtBolum;
    private javax.swing.JButton btnBolumEkle;
    private javax.swing.JButton btnBolumGuncelle;
    private javax.swing.JButton btnBolumSil;
    private javax.swing.JTable tblGoster;

    private DefaultTableModel dtm=new DefaultTableModel();
```

```

public Form()
{
    initComponents();
    tblGoster.setModel(dtm);
    dtm.setColumnIdentifiers (new String [] { "No","Ad","Soyad","Adres","Bölüm"});
}

private void temizle()
{
    spnNumara.setEnabled(true); spnNumara.setValue(0);
    txtAd.setText(""); txtSoyad.setText(""); cmbAdres.setSelectedIndex(0);
}

private void btnNewActionPerformed(java.awt.event.ActionEvent evt)
{
    temizle();
}

private void btnAddActionPerformed(java.awt.event.ActionEvent evt)
{
    dtm.addRow (new String [] { spnNumara.getValue().toString(), txtAd.getText(), txtSoyad.getText(),
                                cmbAdres.getSelectedItem().toString(), txtBolum.getText() } );
    temizle();
}

private void btnUpdateActionPerformed(java.awt.event.ActionEvent evt)
{
    dtm.setValueAt(txtAd.getText(), tblGoster.getSelectedRow(), 1);
    dtm.setValueAt(txtSoyad.getText(), tblGoster.getSelectedRow(), 2);
    dtm.setValueAt(cmbAdres.getSelectedItem(), tblGoster.getSelectedRow(), 3);
    dtm.setValueAt(txtBolum.getText(), tblGoster.getSelectedRow(), 4);
    temizle();
}

private void btnDeleteActionPerformed(java.awt.event.ActionEvent evt)
{
    dtm.removeRow(tblGoster.getSelectedRow());
    temizle();
}

private void btnDeleteAllActionPerformed(java.awt.event.ActionEvent evt)
{
    while (dtm.getRowCount(>0)
    {
        for(int i=0;i<dtm.getRowCount();i++)
            dtm.removeRow(i);
    }
}

```

```

private void tblGosterMouseClicked(java.awt.event.MouseEvent evt)
{
    int indis=tblGoster.getSelectedRow();
    TableModel model=tblGoster.getModel();
    spnNumara.setValue(Integer.parseInt(model.getValueAt(indis, 0).toString()));
    txtAd.setText(model.getValueAt(indis, 1).toString());
    txtSoyad.setText(model.getValueAt(indis, 2).toString());
    cmbAdres.setSelectedItem(model.getValueAt(indis, 3).toString());
    txtBolum.setText(model.getValueAt(indis, 4).toString());
    spnNumara.setEnabled(false);
}

private void filter()
{
    TableRowSorter<TableModel> model = new TableRowSorter<>(tblGoster.getModel());
    tblGoster.setRowSorter(model);
    String filtre = txtFilter.getText();
    if (filtre.trim().length() == 0) model.setRowFilter(null);
    else model.setRowFilter(RowFilter.regexFilter(filtre));
}

private void txtFilterKeyReleased(java.awt.event.KeyEvent evt)
{
    filter();
}

private void txtFilterFocusLost(java.awt.event.FocusEvent evt)
{
    txtFilter.setText("");
    filter();
}

```

```

private void formWindowClosing(java.awt.event.WindowEvent evt)
{
    txtFilter.setText(""); filter();
    File klasor = new File("C:\\Projeler");
    if (!klasor.exists()) klasor.mkdir();
    String filePath = "c:\\Projeler\\ogrenciler.txt";
    try
    {
        File dosya = new File(filePath);
        if (!dosya.exists()) dosya.createNewFile();
        FileWriter fw=new FileWriter(filePath);
        BufferedWriter bw=new BufferedWriter(fw);
        for(int i=0;i<tblGoster.getRowCount();i++)
        {
            for(int j=0;j<tblGoster.getColumnCount();j++)
                bw.write(tblGoster.getValueAt(i, j).toString()+" ");
            bw.newLine();
        }
        bw.close(); fw.close();
    }
    catch(IOException ex)
    {
        JOptionPane.showMessageDialog(null, "Dosya Yok...");
    }
}

private void formWindowOpened(java.awt.event.WindowEvent evt)
{
    String filePath = "c:\\Projeler\\ogrenciler.txt";
    File dosya=new File(filePath);
    try
    {
        FileReader fr=new FileReader(dosya);
        BufferedReader br=new BufferedReader(fr);
        DefaultTableModel model=(DefaultTableModel) tblGoster.getModel();
        Object [] lines=br.lines().toArray();
        for(int i=0;i<lines.length;i++)
        {
            String [] row=lines[i].toString().split(" ");
            model.addRow(row);
        }
    }
    catch(IOException ex)
    {
        JOptionPane.showMessageDialog(null, "Dosya Yok...");
    }
}

```

```

private DefaultMutableTreeNode secilen;

private void treeBolumMouseClicked(java.awt.event.MouseEvent evt)
{
    if (evt.getClickCount()==2)
    {
        secilen=(DefaultMutableTreeNode) treeBolum.getSelectionPath().getLastPathComponent();
        txtBolum.setText(secilen.getUserObject().toString());
    }
}

private void btnBolumEkleActionPerformed(java.awt.event.ActionEvent evt)
{
    secilen=(DefaultMutableTreeNode) treeBolum.getSelectionPath().getLastPathComponent();
    DefaultMutableTreeNode ekle= new DefaultMutableTreeNode(txtBolum.getText());
    secilen.add(ekle);
    DefaultTreeModel model= (DefaultTreeModel) treeBolum.getModel();
    model.reload();
}

private void btnBolumGuncelleActionPerformed(java.awt.event.ActionEvent evt)
{
    secilen=(DefaultMutableTreeNode) treeBolum.getSelectionPath().getLastPathComponent();
    secilen.setUserObject(txtBolum.getText());
    DefaultTreeModel model= (DefaultTreeModel) treeBolum.getModel();
    model.reload();
}

private void btnBolumSilActionPerformed(java.awt.event.ActionEvent evt)
{
    secilen=(DefaultMutableTreeNode) treeBolum.getSelectionPath().getLastPathComponent();
    if (secilen!=treeBolum.getModel().getRoot())
    {
        DefaultTreeModel model= (DefaultTreeModel) treeBolum.getModel();
        model.removeNodeFromParent(secilen);
        model.reload();
    }
}

```

## İLERİ SEVİYE (OOP-OBJECT ORIENTED PROGRAMMING) PROGRAMLAMA:

### Sınıf ve Nesne(Class-Object) Nedir?

Sınıflar nesnelerimiz için genel özellikler tanımladığımız yapılar olarak düşünülebilir. Sınıf aslında doğada yer alan bir cins olarak tanımlanabilir. Birçok farklı cins vardır fakat her cins bir nesne olarak tanımlanmaz. Yani cins birtakım özellikleri bakımından alt cinslere ayrılır.

Yaşamımızda bunun birçok örneğini görüyoruz. Örneğin Varlıklar bir sınıftır. Canlılar ve Cansızlar olarak ikiye ayrılır. Canlı sınıfında Cansızlardan ayrı belirli özellikler vardır. Canlılar içerisindeki Hayvanlar sınıfına dahil Omurgalı Hayvanlardan Balıklar da diğer omurgalı hayvanlardan kendilerini ayıran belli özelliklere sahiptir. Ancak sonuçta tüm omurgalı hayvanları ortak noktada buluşturan bir hayvan sınıfı, tüm canlı ve cansızları da ortak noktada buluşturan bir Varlık sınıfı vardır.

Örneğin **hayvanlar alemi**ni düşünelim. Hayvanlar en basit şekilde Evcil Hayvanlar ve Yabani Hayvanlar olarak iki sınıfa ayrılabilir.

#### 1. Evcil Hayvanlar Sınıfı:Kedi, Köpek, At, Eşek, Tavuk, Horoz

Evcil hayvanlarda farklı alt sınıflara ayrılabilir;

- a. **Kümes Hayvanlar** : Tavuk , horoz, hindi, kaz, ördek
- b. **Büyük Baş Hayvanlar** : Sığır, at, eşek, katır, deve

#### 2. Yabani Hayvanlar Sınıfı : Aslan, Kartal, Yılan, Ayı, Devekuşu, Maymun

Yabani hayvanlarda farklı alt sınıflara ayrılabilir;

- a. **Sürüngenler Sınıfı**: Kaplumbağa, Yılan, Timsah, Kertenkele
- b. **Kuşlar Sınıfı** : Kartal, Leylek, Ağaçkakan, Devekuşu, Penguen

Görüyoruz ki her bir cins birbirinden farklı alt cinslere de ayrılabilir. Ama burada dikkat etmek gerekiyor ki her alt sınıf kendisinden önce gelen üst sınıfların tüm özellikleri taşır.

Bu örneğimizi Java'ya uyarlamamız gerekirse aslında örneğimizdeki her bir cins java da bir sınıfa denk gelmektedir. Her sınıf aslında ortak özelliklere sahip nesnelerin bir araya gelmesiyle oluşur. Java sınıflarının nitelikleri değişkenlerle, davranışları ise metotlarla belirlenir. Oluşturulan sınıfın içerisindeki metotlarda o sınıfa ait değişkenleri kullanabileceği gibi farklı sınıflara bağlanarak onların değişken ve metotlarını da kullanabiliriz.

Sınıf kavramı kullanım açısından tamamen programlayıcıya bağlıdır. Gerek görülürse hiyerarşik bir yapıda iç içe sınıflar oluşturulabilir ve en alttaki sınıftan en üstteki sınıfa erişim sağlayabilir.



## Sınıf ve Nesne Tanımlama:

Sınıf (class) yapısının bu sınıftan üretilen nesneler için bir şablon görevi gördüğünü söyleyebiliriz. Nesne oluşturma'nın ilk adımı, o nesnenin özelliklerini ve eylemlerini belirleyen sınıf yapısını tanımlamaktır. Sınıf içinde özellik tanımlamak için değişkenler, metod tanımlamak için de çeşitli yordamlar bulunabilir.

**final:** Değişkenlere, metodlara ve hatta sınıflara uygulanabilen, kullanıldığı zaman çok faydalı olabilen bu anahtar kelime sayesinde tanımlanan değişken, parametre, metod ya da sınıflar bir çeşit sabitlenmiş olur.

**static:** Java dilinde sadece nesneye ait değil, sınıfa ait değişkenler ve de metodlar da tanımlamak mümkündür. Dolayısıyla static, sınıf içerisindeki diğer nesnelerden bağımsız olarak var olabilecek sınıf üyeleri tanımlamamızı sağlar.

Örneğin, ogrno, ad, soyad gibi değişkenlerin her birinin her öğrenci nesnesi için farklı olması gerektiğinden nesne değişkeni olarak tanımlanması gerekirken, öğrenci sayısı ise bütün öğrenciler için ortak olduğundan sınıf değişkeni olarak tanımlanması gerekir.

**Örnek:**Aşağıdaki **Daire** adlı sınıf, **yarıçap** adında bir özelliği ve **alan** ile **çevre** adlarında iki metodu bulunmaktadır.

SINIF (CLASS) → DAİRE
ÖZELLİK → Yarıçap
METOD → Alan
METOD → Çevre

```
// Daire.java
package Paket;           // Paket
public class Daire       // Sınıf
{
    public double r1;      // Nesne Özellik
    private final double p1=3.14; // Nesne Sabit

    public double alan()   // Nesne Metod
    {
        return p1 * r1 * r1;
    }

    public static double r2; // Sınıf
    private static final double p2=3.14; // Sınıf

    // Sınıf metod sadece static değişken kullanabilir.
    public static double çevre()
    {
        return 2 * p2 * r2;
    }
}
```

```
// DaireTest.java
private void btnHesaplaActionPerformed()
{
    int yariCap=Integer.parseInt(txtYariCap.getText());

    Daire d=new Daire(); // Nesne Tanımı

    d.r1=yariCap;        // Nesne özellik kullanımı
    Daire.r2=yariCap;    // Sınıf özellik kullanımı

    // Nesne Metod Kullanımı
    lblAlan.setText(String.valueOf(d.alan()));

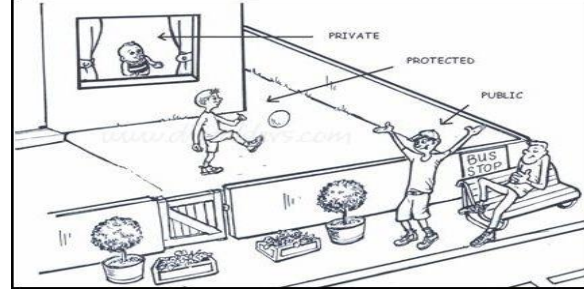
    // Sınıf Metod Kullanımı
    lblCevre.setText(String.valueOf(Daire.cevre()));
}
```

## Access Modifiers(Eriřim Belirleyiciler)

Eriřim belirleyiciler adından da anlaşılacağı üzere bir sınıfın değişkenine ya da metoduna sınıfın içinden, sınıftan oluşturulmuş nesneden ve sınıftan kalıtılmış sınıflardan ulaşımın nasıl olacağını belirleyen kelimelerdir. Java erişim belirleyicileri şunlardır;

### Eriřim Tablosu:

Eriřim Belirleyici	Sınıf içi erişim	Paket içi erişim	Paket dışından kalıtımla erişim	Paket dışından erişim
Private	EVET	HAYIR	HAYIR	HAYIR
Default	EVET	EVET	HAYIR	HAYIR
Protected	EVET	EVET	EVET	HAYIR
Public	EVET	EVET	EVET	EVET



**Public (Genel):** Public belirleyicisi bir değişkeni, bir metodu ya da bir sınıfı niteleyebilir. Nitelediği öğeler herhese açık olur. Bu tür değişken veya metodlara ulaşmak, değişkenlerin değerlerini çekmek ya da değiştirmek mümkündür. Kısaca public her yerden erişilebilir. Uygulama programlarında main() metodunun public damgalı olmasının nedeni budur.

**Private(Özel):** Sınıflar private olarak tanımlanamaz. private erişim belirleyicisi, public erişim belirleyicisinin karşıtı gibidir. private belirleyicisine sadece aynı sınıftan erişilebilir, başka sınıftan(alt sınıflar dahil) erişilemez.

**Protected(Korumalı):** Sınıflar protected olarak tanımlanamaz. Sınıf, alt sınıf ve aynı paketten ulaşılabilir. Farklı paketlerden erişim ise o sınıfı devralan sınıflardan ulaşılabilir.

**Default/Friendly:** Paketler yalnızca ön-tanımlı erişime sahiptir. Paket içindeki her sınıf pakette olan her değişken ve metoda erişilebilir. Ama başka paketlerden erişilemez. Kısaca sadece aynı paketten ulaşılabilir.

```
//Paket Tanımlama  
  
package paket_adi  
{  
    Paket gövdesi  
}
```

```
//Sınıf Tanımlama  
  
public class class_adi  
{  
    Class gövdesi  
}
```

```
//Metot Tanımlama  
  
protected Metot_adi  
{  
    Metot gövdesi  
}
```

```
//Değişken Tanımlama  
  
private tür değişken adı;
```

// Test.java

**package Paket1;**

public class TestClass

```
{
    public static String marka;      / Sınıf Değişkeni
    protected static String cpu;    // Sınıf Değişkeni
    private static int ram;         // Sınıf Değişkeni
    static int hdd;                 // Sınıf Değişkeni
```

```
    private void btnSecActionPerformed()
    {
```

```
        String marka=cmbMarka.getSelectedItem().toString();
        String cpu=cmbCpu.getSelectedItem().toString();
        int ram=Integer.parseInt(cmbRam.getSelectedItem().toString());
        int hdd= Integer.parseInt(cmbHdd.getSelectedItem().toString());
```

```
        Goster frm=new Goster();          // Paket2.Goster frm=new Paket2.Goster();
```

```
        frm.show();
```

```
        frm.setLocationRelativeTo(null);    // Formu Ortada Aç
```

```
        frm.setDefaultCloseOperation(frm.DISPOSE_ON_CLOSE); // Form2 kapat
```

```
    }
}
```

// Goster.java

**package Paket1;**

public class Goster

```
{
    public Goster()          //Constructor
    {
```

```
        initComponents();
        this.setLocationRelativeTo(null);
        txtAreaGoster.append("\nMarka:"+Test.marka);
        txtAreaGoster.append("\nİşlemci:"+Test.cpu);
        txtAreaGoster.append("\nRam:Kısıtlı");
        txtAreaGoster.append("\nHDD:"+Test.hdd);
    }
```

```
}
```

// Goster.java

**package Paket2;**

**import Paket1.Test;**

public class Goster

```
{
    public Goster()          //Constructor
    {
```

```
        initComponents();
        this.setLocationRelativeTo(null);
        txtAreaGoster.append("\nMarka:"+Test.marka);
        txtAreaGoster.append("\nİşlemci:Kısıtlı ");
        txtAreaGoster.append("\nRam:Kısıtlı");
        txtAreaGoster.append("\nHdd:Kısıtlı");
    }
```

```
}
```

```
}
```

## Paketler:

Paket, birbiriyle ya da aynı konuyla ilişkili olan sınıfların bir grup içinde toplanmasıdır. Başka bir deyişle, paket birbiriyle ilişkili sınıfları içeren bir dizindir. Örneğin giriş/çıkış ile ilgili olan bütün sınıflar java.io paketi içindedir.

## Örnek: Paket

// **Paket1 / Basamak.java**

```
package Paket1;

import java.util.ArrayList;

public class Basamaklar
{
    public ArrayList basamakAyir(int sayi)
    {
        int binler, yuzler, onlar, birler;

        binler=(sayi/1000)*1000;
        sayi=sayi-binler;
        yuzler=(sayi/100)*100;
        sayi=sayi-yuzler;
        onlar=(sayi/10)*10;
        birler=sayi-onlar;

        ArrayList liste=new ArrayList();

        if (binler>0) liste.add("Binler Basamağı:"+binler);
        if (yuzler>0) liste.add("Yüzler Basamağı:"+yuzler);
        if (onlar>0) liste.add("Onlar Basamağı:"+onlar);
        if (birler>0) liste.add("Birler Basamağı:"+birler);

        return(liste);
    }
}
```

**Paketler**

Bir sayı Gir:

Binler Basamağı:5000  
Yüzler Basamağı:600  
Onlar Basamağı:80  
Birler Basamağı:7

Hesapla...

// **Paket2 / BasamakTest.java**

```
package Paket2;

import Paket1.Basamak;

public class BasamakTest
{
    // 4 basamaklı bir sayıyı basamaklarına ayırır.

    private void btnHesaplaActionPerformed()
    {
        int sayi=Integer.parseInt(txtSayi.getText());
        if (sayi>9999)
        {
            JOptionPane.showMessageDialog(null,"Geçersiz");
            txtSayi.setText("");
            txtSayi.requestFocus();
            return;
        }
        Paket1.Basamak math=new Paket1.Basamak();
        ArrayList liste=math.basamakAyir(sayi);

        txtAreaGoster.setText("");
        for(Object a:liste)
            txtAreaGoster.append(a+"\n");
    }
}
```

## Örnek: Paket

```
// Paket1 / Metin.java
package Paket1;

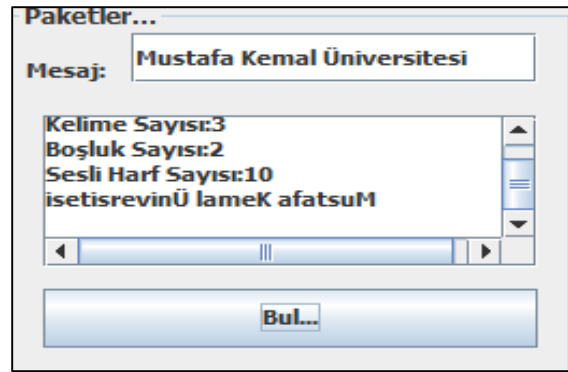
public class Metin
{
    public static int harfSay(String cumle)
    {
        int sonuc=cumle.length();
        for(int i=0; i<cumle.length(); i++)
            if(cumle.charAt(i)==' ') sonuc--;
        return sonuc;
    }

    public static int kelimeSay(String cumle)
    {
        int adet=1;
        for(int i=0; i<cumle.length(); i++)
            if(cumle.charAt(i)==' ') adet++;
        return adet;
    }

    public static int boslukSay(String cumle)
    {
        int adet=0;
        for(int i=0; i<cumle.length(); i++)
            if (cumle.charAt(i)==' ') adet++;
        return adet;
    }

    public static int sesliHarf(String cumle)
    {
        String sesli="aeiöouü";
        int adet=0;
        for(int i=0; i<cumle.length(); i++)
            if(sesli.indexOf(cumle.charAt(i))>=0)
                adet++;
        return adet;
    }

    public static String tersCevir(String cumle)
    {
        String sonuc="";
        for(int i=cumle.length()-1; i>=0; i--)
            sonuc+=cumle.charAt(i);
        return sonuc;
    }
}
```



```
// MetinTest.java
package Paket2;

import Paket1.Metin;

public class MetinTest
{
    private void btnBulActionPerformed()
    {
        String mesaj=txtMesaj.getText();

        String txt="";

        txtAreaGoster.setText("");

        txt="\nHarf Sayısı:"+Metin.harfSay(mesaj);
        txt+="\nKelime Sayısı:"+Metin.kelimeSay(mesaj);
        txt+="\nBoşluk Sayısı:"+Metin.boslukSay(mesaj);
        txt+="\nSesli Harf Sayısı:"+Metin.sesliHarf(mesaj);
        txt+="\nTers Çevir:"+Metin.tersCevir(mesaj);

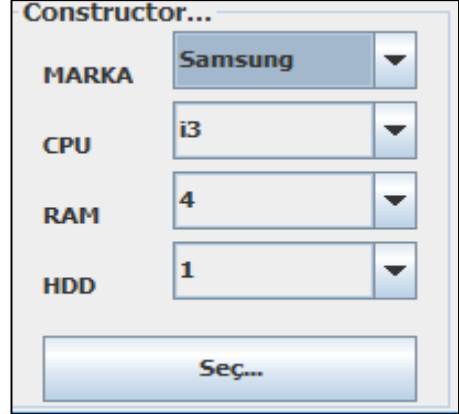
        txtAreaGoster.append(txt);
    }
}
```

## Constructor(Kurucu-Yapıcı Metot):

Constructor metotlar bir sınıftan bir nesne oluşturulduğunda ilk çalışan metotlardır. Bu metotlar aracılığı ile sınıflara ilk çalıştıklarında vermek istediğimiz değerleri verebilir, varsayılan değerlere farklı değerler atayabilir veya bir metot çalıştırabiliriz.

```
// LapTopTest.java
public class LapTopTest
{
    private void btnSecActionPerformed()
    {
        String marka=cmbMarka.getSelectedItem().toString();
        String cpu=cmbCpu.getSelectedItem().toString();
        int ram=Integer.parseInt(cmbRam.getSelectedItem().toString());
        int hdd= Integer.parseInt(cmbHdd.getSelectedItem().toString());

        LapTop lpt1 = new LapTop(); lpt1.göster();
        LapTop lpt2 = new LapTop (marka); lpt2.göster();
        LapTop lpt3 = new LapTop (marka,cpu); lpt3.göster();
        LapTop lpt4 = new LapTop (marka,cpu,ram); lpt4.göster();
        LapTop lpt5 = new LapTop (marka,cpu,ram,hdd); lpt5.göster();
    }
}
```



```
public class LapTop // LapTop.java
{
    String marka,cpu;
    int ram, hdd;
    public LapTop() // constructor sınıf ismi aynı
    {
        this.marka = "Tanımsız";
        this.cpu = "Tanımsız";
        this.ram = 0;
        this.hdd = 0;
    }
    public LapTop (String urun)
    {
        this.urun = marka;
        this.cpu = "tanımsız";
        this.ram = 0;
        this.hdd = 0;
    }
    public LapTop (String ürün, String işlemci)
    {
        this.marka = ürün;
        this.cpu = işlemci;
        this.ram = 0;
        this.hdd = 0;
    }
}
```

```
public LapTop (String ürün, String işlemci, int bellek)
{
    this.marka = ürün;
    this.cpu = işlemci;
    this.ram = bellek;
    this.hdd = 0;
}
public LapTop (String ürün,String işlemci,int bellek,int disk)
{
    this.marka = ürün;
    this.cpu = işlemci;
    this.ram = bellek;
    this.hdd = disk;
}
public void goster()
{
    System.out.println("\nMarka:"+this.marka);
    System.out.println("İşlemci:"+this.cpu);
    System.out.println("Bellek:"+this.ram);
    System.out.println("Sabit Disk:"+this.hdd);
}
}
```

## Encapsulation(Kapsülleme-Sarmalama-Paketleme):

Kapsülleme, bir nesnenin bazı özellik ve işlevlerini başka sınıflardan ve nesnelerden saklamaktır. Kapsülleme sayesinde nesneler bilinçsiz kullanımdan korunmuş olur. Kapsülleme yazılımcıya oluşturduğu sınıftaki değişkenlerin erişimlerini kontrol yetkisi verir.

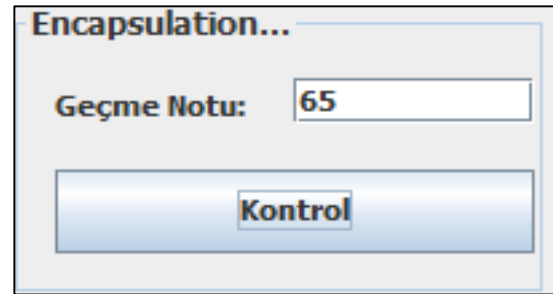
**Private** erişim belirteci sayesinde bu şekilde tanımlanan bir alanı başka sınıflardan gizleyerek kullanımını engellemiş oluruz. Fakat bazı durumlarda private alanlara erişmemiz ve özelliklerini kullanmamız gerekebilir. Bu durumda **Property** kavramı devreye girer. Property bir alanın değerini döndürmeye (**Get**) ve değerini ayarlamaya (**Set**) olanak sağlar.

### Örnek: Kapsülleme1

```
// GecmeNotu.java
public class SistemNotu
{
    private int not;

    public int getNot()
    {
        return this.not;
    }

    public void setNot(int not)
    {
        if (not==60)
            this.not = not;
        else
        {
            this.not = 60;
            System.out.println("Değiştirilemez!");
        }
    }
}
```



```
// GecmeNotuTest.java

public class GecmeNotuTest
{
    private void btnKontrolActionPerformed()
    {
        int nt=Integer.parseInt(txtNot.getText());

        GecmeNotu sistem=new GecmeNotu();

        System.out.println("\nGeçme Notu : "+sistem.getNot());

        sistem.setNot(nt);
    }
}
```

## Örnek: Kapsülleme2

// Araba.java

```
public class Araba
{
    private String tür;
    private String marka;
    private int fiyat;

    public String getTür()
    {
        return tür;
    }

    public void setTür(String tur)
    {
        this.tür = tür;
    }

    public String getMarka()
    {
        return marka;
    }

    public void setMarka(String marka)
    {
        this.marka = marka;
    }

    public int getFiyat()
    {
        return fiyat;
    }

    public void setFiyat(int fiyat)
    {
        this.fiyat = fiyat;
    }
}
```



// ArabaTest.java

```
public class ArabaTest
{
    private void btnGosterActionPerformed()
    {
        Araba oto=new Araba();

        System.out.println("\nAraba Türü:"+oto.getTür());
        System.out.println("Araba Marka:"+oto.getMarka());
        System.out.println("Araba Fiyat:"+oto.getFiyat());

        oto.setTür(txtTur.getText());
        oto.setMarka(txtMarka.getText());
        oto.setFiyat(Integer.parseInt(txtFiyat.getText()));

        System.out.println("\nAraba Türü:"+oto.getTür());
        System.out.println("Araba Marka:"+oto.getMarka());
        System.out.println("Araba Fiyat:"+oto.getFiyat());
    }
}
```



## Inheritance(Kalıtım- Miras):

Inheritance (miras alma, kalıtım), bir nesnenin özelliklerinin farklı nesneler tarafından da kullanılabilmesine olanak sağlayan özelliktir. Yazılan bir sınıf bir başka sınıf tarafından miras alınabilir. Bu işlem yapıldığı zaman temel alınan sınıfın bütün özellikleri yeni sınıfa aktarılır. Ayrıca oluşan bu yeni sınıfın kendisine ait yeni özellikleri de olabilir.

Bir metod birden fazla kez aynı veri tipi, aynı isim ve aynı parametre ile kullanılamadığından bu metod farklı bir işleve sahip gerektiği takdirde yeniden yazılması gerekir. **Override**, herhangi bir sınıftan miras alınan metodu kendimize göre uyarlamamız anlamına gelir.

Javada bütün sınıflar Object sınıfından extends(kalıtım-miras) edilmiştir.Yani her sınıf object sınıfının alt sınıfıdır ve Object sınıfındaki metotları override(değiştirilebilir) edebilir.

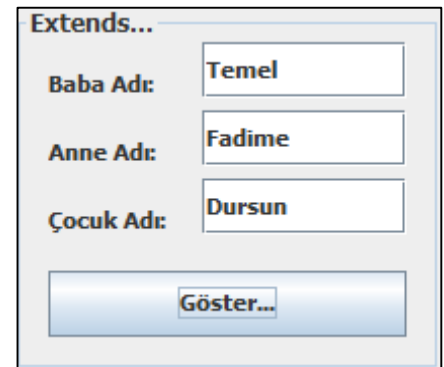
### Örnek: Kalıtım1

<pre>// Paket1 / Babam.java  public class Babam {     private String baba;      public Babam(String b)     {         this.baba = b;     }      public void göster()     {         System.out.println("Baba:"+baba);     } }</pre>	<pre>// Paket1 / Annem.java  public class Annem extends Babam {     private String anne;      public Annem(String b,String a)     {         super(b);         this.anne = a;     }      @Override     public void göster()     {         super.göster();         System.out.println("Anne:"+anne);     } }</pre>	<pre>// Paket1 / Cocugum.java  public class Cocugum extends Annem {     private String cocuk;      public Cocugum(String b,String a,String c)     {         super(b,a);         this.cocuk = c;     }      @Override     public void göster()     {         super.göster();         System.out.println("Çocuk: "+cocuk);     } }</pre>
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

```
// Paket1 / AilemTest.java

private void btnHesaplaActionPerformed()
{
    String baba=txtBaba.getText();
    String anne=txtAnne.getText();
    String cocuk=txtCocuk.getText();

    Babam babam=new Babam(baba); babam.göster();
    Annem annem=new Annem(baba,anne); annem.göster();
    Cocugum cocugum=new Cocugum(baba,anne,cocuk); cocugum.göster();
}
```



## Örnek:PaketKalıtım

// Paket1 / Spor.java

```
public class Spor
{
    private String takım, ülke;
    public Spor(String takım, String ülke)
    {
        this.takım = takım; this.ülke = ülke;
    }
    public void göster()
    {
        System.out.println("\nTakım : "+this.takım);
        System.out.println("Ülke : "+this.ülke);
    }
}
```

// Paket2 / Futbol.java

```
import Paket1.Spor;
public class Futbol extends Spor
{
    private int şampiyon;
    public Futbol(String takım,String ülke,int şampiyon)
    {
        super(takım,ülke); this.şampiyon = şampiyon;
    }
    @Override
    public void göster()
    {
        super.göster(); //spor sınıfından metod çağır
        System.out.println("Şampiyon: "+this.şampiyon);
    }
}
```

// Paket2 / Basketbol.java

```
import Paket1.Spor;
public class Basketbol extends Spor
{
    private String kuruluş,maliyet;
    public Basketbol(String takım,String ülke,
        String kuruluş,String maliyet)
    {
        super(takım,ülke);
        this.kuruluş = kuruluş; this.maliyet=maliyet;
    }
    @Override
    public void göster()
    {
        super.göster(); //spor sınıfından metod çağır
        System.out.println("Tarihi:"+this.kuruluş);
        System.out.println("Değeri:"+this.maliyet);
    }
}
```

// Paket3 / FutbolTest.java

import Paket2.Futbol;

```
private void btnHesaplaActionPerformed()
{
    String takım=cmbTakim.getSelectedItemAt().toString();
    String ulke=cmbUlke.getSelectedItemAt().toString();
    String sampiyon=spnSampiyon.getValue().toString();

    Futbol futbol = new Futbol(takim,ulke,sampiyon);
    futbol.göster();
}
```

// Paket3 / BasketbolTest.java

import Paket2.Basketbol;

```
public BasketbolTest()
{
    initComponents();
    this.setLocationRelativeTo(null);
    for(int yil=1900;yil<=2022;yil++)
        cmbKurulus.addItem(String.valueOf(yil));
}
private void btnHesaplaActionPerformed()
{
    String takım=cmbTakim.getSelectedItemAt().toString();
    String ulke=cmbUlke.getSelectedItemAt().toString();
    String kurulus=cmbKurulus.getSelectedItemAt().toString();
    String deger=txtDeger.getText();
    Basketbol basket=new Basketbol(takim,ulke,kurulus,deger);
    basket.göster();
}
```

## Composition(Kompozisyon-Kopyalama):

Kompozisyon, bir sınıf içerisinde başka bir sınıfın nesnesini oluşturma durumudur. Sınıflar kalıtımda olduğu gibi hiyerarşik bir yapıda bulunmaz. Kompozisyon, daha önceden yazılmış sınıfların özelliklerini kullanmak için temiz bir yöntemdir. Kompozisyon birleşim olarak da adlandırılır. Kompozisyon kalıtıma alternatif olarak kullanılır.

### Örnek: Kompozisyon

```
// Kafa.java
public class Kafa
{
    private String voltranKontrol;
    public Kafa(String voltranKontrol)
    {
        this.voltranKontrol = voltranKontrol;
    }
    public String getVoltranKontrol ()
    {
        return voltranKontrol;
    }
}

// SagKol.java
public class SagKol
{
    private String çevreKontrol;
    public SagKol(String çevreKontrol)
    {
        this.çevreKontrol = çevreKontrol;
    }
    public String getÇevreKontrol()
    {
        return çevreKontrol;
    }
}

// SolKol.java
public class SolKol
{
    private String güçKontrol;
    public SolKol(String güçKontrol)
    {
        this.güçKontrol = güçKontrol;
    }
    public String getGüçKontrol()
    {
        return güçKontrol;
    }
}
```

```
// SagBacak.java

public class SagBacak
{
    private String silahKontrol;

    public SagBacak(String silahKontrol)
    {
        this.silahKontrol = silahKontrol;
    }

    public String getSilahKontrol()
    {
        return silahKontrol;
    }
}

// SolBacak.java

public class SolBacak
{
    private String korumaKontrol;

    public SolBacak(String korumaKontrol)
    {
        this.korumaKontrol = korumaKontrol;
    }

    public String getKorumaKontrol()
    {
        return korumaKontrol;
    }
}
```

```
// Voltran.java
```

```
import java.util.ArrayList;
```

```
public class Voltran
```

```
{  
    private Kafa kafa;           // Composition.  
    private SagKol sagkol;       // Composition.  
    private SolKol solkol;       // Composition.  
    private SagBacak sagbacak;   // Composition.  
    private SolBacak solbacak;   // Composition.  
}
```

```
public Voltran(Kafa kf, SagKol sgk, SolKol slk, SagBacak sgb, SolBacak slb)
```

```
{  
    this.kafa = kf;  
    this.sagkol = sgk;  
    this.solkol = slk;  
    this.sagbacak = sgb;  
    this.solbacak = slb;  
}
```

```
public ArrayList göster()
```

```
{  
    ArrayList voltran=new ArrayList();  
    voltran.add(this.kafa.getVoltranKontrol());  
    voltran.add(this.sagkol.getÇevreKontrol());  
    voltran.add(this.solkol.getGüçKontrol());  
    voltran.add(this.sagbacak.getSilahKontrol());  
    voltran.add(this.solbacak.getKorumaKontrol());  
    return(voltran);  
}
```

```
// VoltranTest.java
```

```
private void btnHesaplaActionPerformed()
```

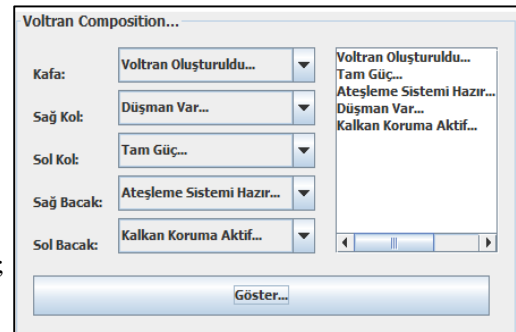
```
{  
    String kafa=cmbKafa.getSelectedItem().toString();  
    String sagKol=cmbSagKol.getSelectedItem().toString();  
    String solKol=cmbSolKol.getSelectedItem().toString();  
    String sagBacak=cmbSagBacak.getSelectedItem().toString();  
    String solBacak=cmbSolBacak.getSelectedItem().toString();  
}
```

```
Kafa kf=new Kafa(kafa);  
SagKol sgk=new SagKol(sagKol);  
SolKol slk=new SolKol(solKol);  
SagBacak sgb=new SagBacak(sagBacak);  
SolBacak slb=new SolBacak(solBacak);
```

```
Voltran voltran=new Voltran(kf , sgk , slk , sgb , slb);
```

```
txtAreaEkran.setText("");
```

```
for(Object robot:voltran.göster())  
    txtAreaEkran.append(robot+"\n");  
}
```



## Polymorphism(Çok Biçimlilik-Çok Şekillilik):

Polymorphism, kalıtım kavramı ile iç içe bir yapıdadır. Polymorphism, bir nesnenin birbirinden farklı nesneler şeklinde davranmasıdır. Yani, ana sınıf içerisindeki bir metodun alt sınıflarda değiştirilerek kullanılmasıdır. Polymorphism, kısaca bir nesnenin birden fazla sınıf için kullanılabilmesidir.

**Örnek:** Bir metod ekrana Türkler için "selam", Almanlar için "hallo", İngilizler için "hello" yazdıracak şekilde çeşitlendirilebilir.

```
// Selam.java
public class Selam
{
    private final String ad;

    public Selam (String ad)
    {
        this.ad=ad;
    }

    public String getAd()
    {
        return this.ad;
    }

    public String getMerhaba()
    {
        return "";
    }
}
```

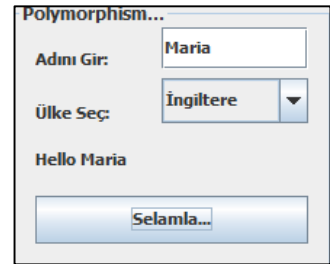
```
// SelamTest.java

private void btnSelamActionPerformed()
{
    String ad=txtAd.getText();

    Selam [] insanlar= {new Turk(ad), new Alman(ad), new Ingiliz(ad)};

    for (Selam s:insanlar)
        System.out.println(s.getMerhaba()+" " + s.getAd());

    String ulke=cmbUlke.getSelectedItem().toString();
    if (cmbUlke.getSelectedIndex()==0)
        lblGöster.setText(insanlar[0].getMerhaba() +" " + insanlar[0].getAd());
    else if (cmbUlke.getSelectedIndex()==1)
        lblGöster.setText(insanlar[1].getMerhaba() +" " + insanlar[1].getAd());
    else if (cmbUlke.getSelectedIndex()==2)
        lblGöster.setText(insanlar[2].getMerhaba() +" " + insanlar[2].getAd());
}
```



```
// Turk.java
public class Turk extends Selam
{
    public Turk(String ad)
    {
        super(ad);
    }
    @Override
    public String getMerhaba()
    {
        return "Merhaba";
    }
}
```

```
// Ingiliz.java
public class Alman extends Selam
{
    public Alman(String ad)
    {
        super(ad);
    }
    @Override
    public String getMerhaba()
    {
        return "Hallo";
    }
}
```

```
// Alman.java
public class Ingiliz extends Selam
{
    public Ingiliz(String add)
    {
        super(ad);
    }
    @Override
    public String getMerhaba()
    {
        return "Hello";
    }
}
```

## Abstraction(Soyutlama):

Abstract sınıfları genel olarak inheritance (kalıtım) uygularken kullanılır. Soyut sınıflardan nesne oluşturulamaz, ancak başka bir sınıflar türetilir. Soyut bir sınıf tanımlamak için abstract ifadesi kullanılır. Hem gövdeli hem de gövdesiz metotları içinde barındırabilir. Abstract sınıf, Interface ve Inheritance'in karışımı denebilir.

Bazen oluşturulan sınıflar direkt olarak kullanılamazlar ve kendisinden başka sınıflar türetilsin diye yazılır. Bu durumdaki sınıflara abstract(soyut) sınıflar denir. Soyut sınıf tıpkı normal bir sınıf gibi oluşturulur. Normal bir sınıftan tek farkı bir kopyasının elde edilemez olmasıdır. Ancak, soyut sınıftan başka bir sınıf türetilir. Veri alanları, metotları ve constructor metotları aynen normal bir sınıfta olduğu şekilde tanımlanır.

Soyut bir sınıfın kopyası oluşturulamadığından, bu sınıftan başka bir sınıf türetilmediği sürece pratik olarak kullanımı pek mümkün değildir. Genellikle, kendisinden türetilen çocuk sınıflarda kullanılan ortak metotları içeren ana sınıflar, bu anlamda soyut olduğundan, tek başlarına kullanılamazlar.

Örneğin, Üniversitede Vize ve Final her öğrencinin girdiğini düşünelim. Bütünleme ise sadece başarısız olanlar girer. Vize ve final herkeste ortak olduğundan somut iken, bütünleme ise değişken olduğundan soyut(abstract) olarak tanımlanmalıdır .

## Örnek:

```
// Sekil.java
public abstract class Sekil
{
    private String tür;

    public Sekil(String tür)
    {
        this.tür = tür;
    }

    public String getTür()
    {
        return tür;
    }

    public void setTür(String tür)
    {
        this.tür = tür;
    }

    //Kare ve daire alanı hesaplayacak
    abstract void alan_hesapla();
    //abstract void cevre_hesapla();
}
```

```
// Kare.java
public class Kare extends Sekil
{
    private int kenar;

    public Kare(String isim,int kenar)
    {
        super(isim);
        this.kenar = kenar;
    }

    @Override
    public double alan_hesapla()
    {
        return kenar*kenar;
    }

    public double cevre_hesapla()
    {
        return 4*kenar;
    }
}
```

```
// Daire.java
public class Daire extends Sekil
{
    private int r;

    public Daire(String isim,int r)
    {
        super(isim);
        this.r = r;
    }

    @Override
    public double alan_hesapla()
    {
        return Math.PI*r*r;
    }

    public double cevre_hesapla()
    {
        return 2*Math.PI*r;
    }
}
```

// SekilTest.java

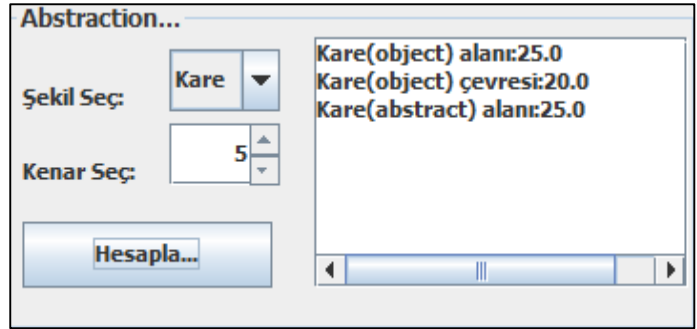
```
private void cmbSekilActionPerformed()
{
    if (cmbSekil.getSelectedIndex()==0)
        lblKenar.setText("Kenar Seç:");
    else
        lblKenar.setText("Yarıçap Seç:");
}

private void btnHesapActionPerformed()
{
    String sekil=cmbSekil.getSelectedItem().toString();
    int deger=Integer.parseInt(spnDeger.getValue().toString());
    txtAreaEkran.setText("");

    if (cmbSekil.getSelectedIndex()==0)
    {
        Kare nesne = new Kare(sekil,deger);
        txtAreaEkran.append(nesne.getTür()+"(object) alanı:"+String.valueOf(nesne.alan_hesapla())+"\n");
        txtAreaEkran.append(nesne.getTür()+"(object) çevresi:"+String.valueOf(nesne.cevre_hesapla())+"\n");

        Sekil soyut = new Kare(sekil,deger);
        txtAreaEkran.append(soyut.getTür()+"(abstract) alanı:"+String.valueOf(soyut.alan_hesapla())+"\n");
        // txtAreaEkran.append(soyut.getTür()+"(abstract) çevresi:"+String.valueOf(soyut.cevre_hesapla())+"\n");
    }
    else
    {
        Daire nesne = new Daire(sekil,deger);
        txtAreaEkran.append(nesne.getTür()+"(object) alanı:"+String.format("%.2f",nesne.alan_hesapla())+"\n");
        txtAreaEkran.append(nesne.getTür()+"(object) çevresi:"+String.format("%.2f",nesne.cevre_hesapla())+"\n");

        Sekil soyut = new Daire(sekil,deger);
        txtAreaEkran.append(soyut.getTür()+"(abstract) alanı:"+String.format("%.2f",soyut.alan_hesapla())+"\n");
        // txtAreaEkran.append(soyut.getTür()+"(abstract) çevresi:"+String.format("%.2f",soyut.cevre_hesapla())+"\n");
    }
}
```



## Interface(Arayüz):

Interface, içerisinde sadece kendisinden türeyecek olan sınıfların içini dolduracağı metod tanımlarının bulunduğu ve soyutlama yapmamıza olanak sağlayan bir yapıdır. Yani Arayüzler sınıflara kural getirmek için kullanılabileceği gibi, ortak kullanım alanları da oluşturulabilir. Interface bir tür class olmasına karşın nesne ve gövdeli metod oluşturamadığı gibi değişken tanımlaması da yapılamaz.

Bir arayüz tanımlamak için **interface** ifadesi, bir arayüzü kullanmak için ise, sınıf bildirimi ile birlikte **implements** ifadesi kullanılır. Arayüzler direk iş yapamayacaklarından, implement edildikleri sınıflarda override edilmelidirler.

### //DünyaTest.java

```
private void btnGösterActionPerformed()
{
    String ad=txtCocuk.getText();
    String kara=cmbKara.getSelectedItem().toString();
    String hava=cmbHava.getSelectedItem().toString();
    String deniz=cmbDeniz.getSelectedItem().toString();
    txtAreaEkran.setText("");
```

```
    İnsan i = new İnsan(ad); txtAreaEkran.append(i.getCocuk()+"\n");
    Kara k=new Kara(kara); txtAreaEkran.append(k.getKaraTaşıt()+"\n");
    Hava h=new Hava(hava); txtAreaEkran.append(h.getHavaTaşıt()+"\n");
    Deniz d=new Deniz(deniz); txtAreaEkran.append(d.getDenizTaşıt()+"\n");
}
```

### //IDünya.java

```
public interface IDunya
{
    public String getCocuk();
}
```

### //İnsan.java

```
public class İnsan implements IDunya
{
    private String ad;
    public İnsan(String ad)
    {
        this.ad = ad;
    }
    public String getAd()
    {
        return ad;
    }
    @Override
    public String getCocuk()
    {
        return getAd()+" dünyanın geleceğidir.";
    }
}
```



// **ITaşıtlar.java**

```
public interface ITaşıtlar
{
    public interface Tekerlek
    {
        public String getKaraTaşıtlar();
    }
    public String getHavaTaşıtlar();
    public String getDenizTaşıtlar();
}
```

// **Kara.java**

```
public class Kara implements ITaşıtlar.Tekerlek
{
    private String ad;
    public Kara(String ad)
    {
        this.ad = ad;
    }
    public String getAd()
    {
        return ad;
    }
    @Override
    public String getKaraTaşıtlar()
    {
        return getAd()+" Kara Taşıtlardır.";
    }
}
```

// **Hava.java**

```
public class Hava implements ITaşıtlar
{
    private String ad;
    public Hava(String ad)
    {
        this.ad = ad;
    }
    public String getAd()
    {
        return ad;
    }
    @Override
    public String getHavaTaşıtlar()
    {
        return getAd()+" Hava Taşıtlardır.";
    }
    @Override
    public String getDenizTaşıtlar()
    {
        return "";
    }
}
```

// **Deniz.java**

```
public class Deniz implements ITaşıtlar
{
    private String ad;
    public Deniz(String ad)
    {
        this.ad = ad;
    }
    public String getAd()
    {
        return ad;
    }
    @Override
    public String getHavaTaşıtlar()
    {
        return getAd()+" Hava Taşıtlardır.";
    }
    @Override
    public String getDenizTaşıtlar()
    {
        return "";
    }
}
```

## Örnek: Interface

```
// INotHesaplama.java //Interface
public interface INotHesaplama
{
    public double getGNT();
    public void setGNT(double gnt);
}
```

// FinalHesap.java

```
public class FinalHesap extends Hesap
{
    public FinalHesap(double vnotu, double fnotu)
    {
        super(vnotu, fnotu);
        super.setGNT(Math.round(vnotu*0.4+fnotu*0.6));
        if(getGNT()<60) new ButTest();
    }
}
```

// ButHesap.java

```
public class ButHesap extends Hesap
{
    public ButHesap(double vnotu, double bnotu)
    {
        super(vnotu, bnotu);
        super.setGNT(Math.round(vnotu*0.4+bnotu*0.6));
    }
}
```

// Hesap.java

```
public class Hesap implements INotHesaplama
```

```
{
    private double vnt;
    private double fnt;
    private double gnt;
```

```
    public Hesap(double vnt, double fnt)
    {
        this.vnt = vnt;
        this.fnt = fnt;
    }
```

```
    @Override
    public double getGNT()
    {
        return gnt;
    }
```

```
    @Override
    public void setGNT(double gnt)
    {
        this.gnt=gnt;
    }
}
```

// FinalTest.java

```
public static double vnotu;
private double fnotu;
private void btnHesapActionPerformed()
{
    vnotu=Double.parseDouble(txtVize.getText());
    fnotu=Double.parseDouble(txtFinal.getText());
    FinalHesap finalim=new FinalHesap(vnotu, fnotu);
    if (finalim.getGNT()<60)
    {
        JOptionPane.showMessageDialog(null, finalim.getGNT()+" notuyla bütünlemeye kaldın");
        ButTest frm=new ButTest();
        frm.show(); frm.setLocationRelativeTo(null); // Form2 Ortada Aç
        frm.setDefaultCloseOperation(ButTest.DISPOSE_ON_CLOSE); // Form2 kapat
    }
    else JOptionPane.showMessageDialog(null, finalim.getGNT()+" notuyla finalde geçtin...");
}
```

// ButTest.java

```
private void txtButKeyReleased()
{
    double bnotu=Double.parseDouble(txtBut.getText());
    ButHesap but=new ButHesap(FinalTest.vnotu, bnotu);
    if (but.getGNT()<60) lblBut.setText(but.getGNT()+" notuyla ders tekrarına kaldın");
    else lblBut.setText(but.getGNT()+" notuyla bütünlemede Geçtin");
}
```

## JAVA WEB PROGRAMLAMA:

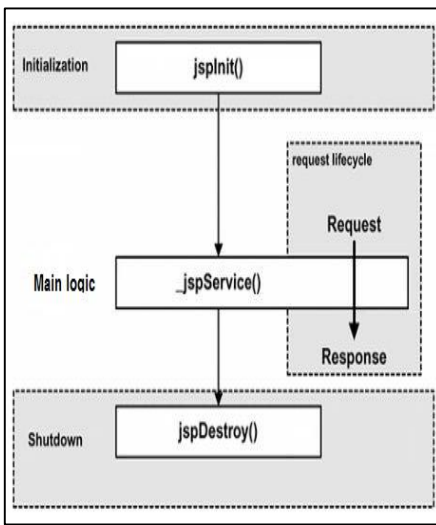
### JSP(Java Server Pages) nedir?

JSP, Web sayfalarında Java dilini kullanarak dinamik web sayfaları oluşturmamızı sağlayan bir Java teknolojisidir. Jsp html dili içine **<% %>** tagları arasında yazılır.

### JSP'nin Çalıştığı Yer Uygulama Sunucusudur.

JSP'de yazdığınız kodların çalışması için Web Sunucularına değil Uygulama Sunucu'suna atmanız gerekir. JSP bir Java uygulamasıdır. Bu yüzden JSP'nin çalışması için **Apache Tomcat** uygulama sunucusuna ihtiyacımız olacaktır.

### JSP MİMARİSİ



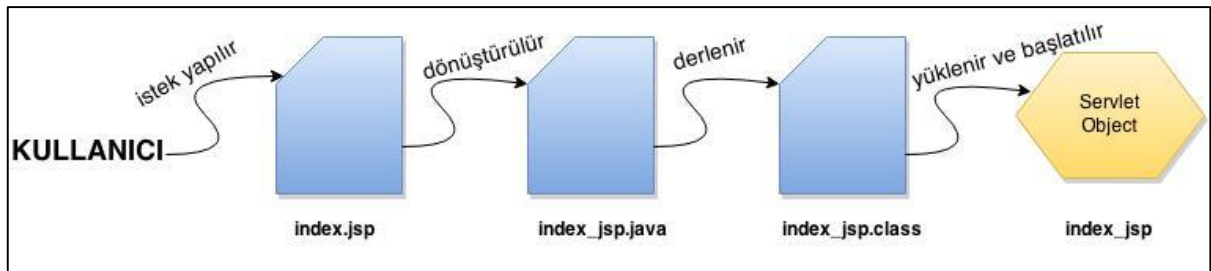
**jspInit():** Başlangıçta çalıştırılması gereken işlemlerin yapıldığı yerdir. Sadece bir kez çalışır.

**\_jspService():** Kullanıcı isteklerinin cevaplandığı yerdir. İşlem tamamlanıncaya kadar burada kalınır.

**jspDestroy():** Uygulama kapanmadan önce yapılması gereken işlemler override edilerek içerisine yazılabilir. Sadece bir kez çalışır.

### JSP Yaşam Döngüsünde Server Nasıl Davranır?

- Bir **JSP** sayfasına **request** geldiğinde sunucu bu **JSP** sayfasının derlenip derlenmediğine bakar. Derlenmemiş dosyaları derleyerek **.java** ve **.class** uzantılı dosyaların oluşturulmasını sağlar.
- Eğer **JSP** sayfasının kodları değiştirilip tekrar çalıştırılırsa **Server** bu sayfanın daha önce derlenmediğini düşünür ve tekrar derleme işlemi yapar.
- **jspDestroy()** metodu, sunucu kapatıldığında, sunucu bellek sıkıntısı olduğuna karar verdiğinde, sayfa uzun süre **request** olmadığına devreye girer.



## JSP Kodlama (Scripting) Etiketleri:

Java Server Pages kodlama stili taglar arasında olmaktadır. Her tag'ın belirli bir işlevi bulunmaktadır. Bu taglar kod okunmasını kolaylaştırır ve profesyonel projeler de JSP kullanımı taglar özelliklerine göre kodlanmaktadır.

JSP için açıklama satırları `<%-- --%>` etiketleri arasında verilmelidir.

**Scriptlet Tag:** Bu etiketler içerisinde değişken tanımlama ve java kodlama yapılabilir. Dolayısıyla bu değişkenlere sadece aynı kapsam içerisinde erişilebilir.

```
<%--
```

### Örnek1: Scriptlet (Kodlama) Tag

Java Kodlama Bloğudur. `<% %>`

```
--%>
```

```
<%
```

```
int sayı1=100, sayı2=200, toplam=sayı1+sayı2;
```

```
out.println("<p>Birinci Sayı:"+sayı1);
```

```
out.println("<p>İkinci Sayı:"+sayı2);
```

```
out.println("<p>Toplam:"+toplam);
```

```
%>
```

```
<%--
```

### Örnek2: Expression (İfade) Tag

Tek satırlık ekran çıktı bloğudur. `<%= %>`

Satır sonuna ; konulamaz .

= önce boşluk bırakılmaz.

```
--%>
```

```
<%
```

```
String ülkem1="Türkiye";
```

```
String ülkem2=" Cumhuriyeti";
```

```
String ülkem=ülkem1+ülkem2;
```

```
%>
```

```
<%=ülkem+"<p>"%> <hr>
```

```
<% =ülkem1.concat(ülkem2)%> // Hata
```

### Örnek3:

```
<h3> merhaba jsp Dünyası...</h3><hr>
```

```
<%
```

```
String program = "Java Web Öğreniyorum...";
```

```
for (int i = 1; i <= 10; i++)
```

```
{
```

```
out.print("merhaba : " + i + "<br>");
```

```
}
```

```
%>
```

```
<hr>
```

```
<%= "<h3>" + program %>
```

### Örnek4:

```
<%
```

```
for(int i=1; i<=10; i++)
```

```
{
```

```
for(int j=1; j<=i; j++)
```

```
{
```

```
out.print(" "+j);
```

```
}
```

```
out.println("<hr>");
```

```
}
```

```
%>
```

**Declaration Tag:** İçerisinde ise hem değişken hem de metot tanımı yapılabilir. Dolayısıyla bu tanımlamalara her yerden erişim sağlanabilir. Ayrıca bu tanımlamalar bir kere oluşturulacağından dolayı her istek(request) geldiğinde bellek kullanılmaz.

<!--

**Örnek1: Declaration (Bildiri) Tag**

**Değişken tanımlamaya yarar. <%! %>**

-->

```
<% int sayac1 = 0; %>
```

```
<%
```

```
sayac1++;
```

```
out.println("<p>Sayaç1: "+sayac1);
```

```
%>
```

```
<%! int sayac2 = 0; %>
```

```
<%= "<p>Sayaç2: "+sayac2++ %>
```

```
<%= "<p>Sayaç2: "+sayac2 %>
```

<!--

**Örnek2: Declaration (Bildiri) Tag**

**Metot tanımlamaya yarar. <%! %>**

-->

```
<%!
```

```
boolean kontrol(String a,String b)
```

```
{
```

```
a=a.toUpperCase().trim();
```

```
b=b.toLowerCase().trim();
```

```
if (a.equals(b)) return true;
```

```
else return false;
```

```
}
```

```
%>
```

```
<%
```

```
String kullanıcı = " admin";
```

```
String user="admin ";
```

```
out.print(kontrol(kullanıcı, user));
```

```
%>
```

**Örnek3:**

```
<% double c=500;%>
```

```
<%! double d=500;%>
```

```
<%! // Kullanılmalı
```

```
double topla(double a, double b)
```

```
{
```

```
return a+b;
```

```
//return a+b+c;
```

```
//return a+b+d;
```

```
}
```

```
%>
```

```
<% out.print("<p>Toplam : " + topla(100, 200)); %>
```

```
<%= "<p>Toplam:" + topla(100,200) %>
```

**Örnek4:**

```
<%!
```

```
int faktör(int n)
```

```
{
```

```
if (n == 1) return n;
```

```
else return n * faktör(n - 1);
```

```
}
```

```
%>
```

```
<%= "5 in faktoriyeli... " + faktör(5)%>
```

**Form Tag:** Bir sayfadan başka sayfaya veri taşımada en çok kullanılan yöntemdir.

//orneka.jsp

```
<body>
<form name="form1" method="get" action="ornekb.jsp">
<label>Bir ay numarası gir:</label>
<input type="text" name="ay">
<input type="submit" name="gonder" value="Gönder">
</form>
</body>
```

// ornekb.jsp

```
<body>
<%
String gelenBilgi=request.getParameter("ay");
String ayAd="";
int ayNo=Integer.parseInt(gelenBilgi);

switch(ayNo)
{
case 1: ayAd="Ocak"; break;
case 2: ayAd="Şubat"; break;
case 3: ayAd="Mart"; break;
case 4: ayAd="Nisan"; break;
case 5: ayAd="Mayıs"; break;
case 6: ayAd="Haziran"; break;
case 7: ayAd="Temmuz"; break;
case 8: ayAd="Ağustos"; break;
case 9: ayAd="Eylül"; break;
case 10: ayAd="Ekim"; break;
case 11: ayAd="Kasım"; break;
case 12: ayAd="Aralık"; break;
default: ayAd="Geçersiz Değer...";
}
out.println(ayNo+" . ay "+ayAd);

/*
String [] aylar={"Ocak","Şubat","Mart","Nisan","Mayıs","Haziran",
                "Temmuz","Ağustos","Eylül","Ekim","Kasım","Aralık"};
out.println(ayNo+" . ay "+aylar[ayNo-1]);
*/
%>
</body>
```

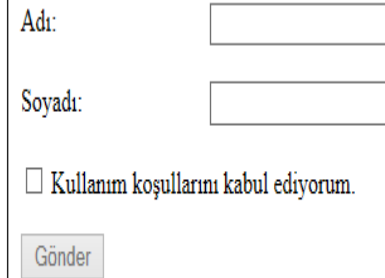
## JavaScript:

Javascript bir betiktir. Sadece o yazılan sayfa üzerinde anlık olarak çalışır ve sonlanır. Javascript'i nasıl çalıştırırız. Bunun cevabı oldukça kolay. Aynı html sayfalarda nasıl bir işlem uygulanıyorsa aynısını yapmamız yeterli olacaktır.

```
// ornek1.jsp
<style>
#uyelik_formu
{
padding: 25px;
width: 350px;
border: 1px solid black;
}
label
{
float: left; width: 150px;
}
</style>

<div align="center">
<div id="uyelik_formu">
<form action="#" method="post">
<label>Adı:</label>
<input type="text" id="ad" name="ad" onkeyup="form_kontrol()">
<br><br> <label>Soyadı:</label>
<input type="text" id="soyad" name="soyad" onkeyup="form_kontrol()">
<br><br> <input type="checkbox" id="kosullar" name="kosullar" onchange="form_kontrol()">
Kullanım koşullarını kabul ediyorum.
<p> <button type="submit" id="gonder" disabled>Gönder</button>
</form>
</div>
</div>

<script>
function form_kontrol()
{
if(document.getElementById("ad").value.length == 0)
document.getElementById("gonder").disabled = true;
else if(document.getElementById("soyad").value.length == 0)
document.getElementById("gonder").disabled = true;
else if(document.getElementById("kosullar").checked == false)
document.getElementById("gonder").disabled = true;
else
document.getElementById("gonder").disabled = false;
}
</script>
```



### //ornek2.jsp

```
<% @ page language="java" contentType="text/html; charset=UTF-8" %>
```

```
<body onload="yeni()">
<div id="sayiTahmin">
<input type="text" id="sayi" />
<button id="btn1" onclick="tahmin()">Tahmin Et</button>
<button id="btn2" onclick="yeni()">Yeni OYUN</button>
<p id="sonuc"></p>
</div>
</body>
<script>
var hakSayisi = 0;
var rastgeleSayi = 0;
var b1 = document.getElementById("btn1");
var b2 = document.getElementById("btn2");
var sonuc = document.getElementById("sonuc");
var sayi = document.getElementById("sayi");

function yeni()
{
    b1.disabled = false;
    b2.disabled = true;
    rastgeleSayi = Math.floor(Math.random()*101);
    hakSayisi = 10;
    sonuc.innerHTML = "";
    sayi.value = "";
}

function tahmin()
{
    var girilenSayi = parseInt(sayi.value);
    var mesaj = "";
    if (isNaN(girilenSayi)) return;

    if (rastgeleSayi > girilenSayi) mesaj = "Daha BÜYÜK değer deneyin";
    else if (girilenSayi > rastgeleSayi) mesaj = "Daha KÜÇÜK değer deneyin";
    else
    {
        mesaj = "Tebrikler!!!";
        btn1.disabled = true;
        btn2.disabled = false;
    }
    hakSayisi--;
    sonuc.innerHTML = mesaj;
    sonuc.innerHTML += "<br />";
    sonuc.innerHTML += "Kalan hak sayısı : " + hakSayisi;
    if (hakSayisi == 0)
    {
        btn1.disabled = true;
        btn2.disabled = false;
    }
}
</script>
```

Tebrikler!!!

Kalan hak sayısı : 4



## Include Direktif:

Mevcut JSP sayfasına başka bir sayfa eklememizi sağlar. Bu işlemi **file** özneliği ile yapar. Kullanım **<%@ include file="dahiledilecekSayfa.jsp" %>** şeklindedir.

```
// main.jsp
<% @ page contentType="text/html; charset=UTF-8"%>
<% @ include file = "header.jsp" %>
<hr>
<center>
  <h2>Web Sitemize Hoş Geldiniz...</h2>
</center>
<hr>
<% @ include file = "footer.html" %>

//footer.jsp

<center>
  <h3>Copyright © 2020</h3>
</center>

//header.jsp

<% @ page contentType="text/html; charset=UTF-8"%>
<%!
int sayac = 0;
void saydir()
{
  sayac++;
}
%>
<% saydir();%>

<center>
<h1>JSP Include Uygulama</h1>
<h3>Sitemizi <%= sayac %> .kez ziyaret etmektesiniz...</h3>
</center>
```

**Page Etiket:** Sayfaya özgü özellikleri tanımlar. En önemli özellikleri şunlardır;

**a. language: JSP** ` de kullanılan programlama dilini belirler. Varsayılan değeri **java**'dır.

```
<%@page language="java" %>
```

**b. contentType:** Sayfanın türünü belirler.

```
<%@page contentType="text/html" %>
```

**c. import:** Çeşitli paketleri ve bu paketlerin altındaki java classlarını dahil etmemizi sağlar.

```
//ornek1.jsp // internet Explorer çalışıyor

<% @ page contentType="text/html; charset=UTF-8"%>
<%--
<% @page contentType="application/msword"%>
<% @page contentType="application/vnd.ms-excel"%>
--%>

<table border="2">
<tr>
<td><b>Bölüm</b></td>
<td><b>Program</b></td>
<td><b>Öğrenci Sayısı</b></td>
</tr>
<tr>
<td>Bilgisayar Teknolojileri</td>
<td>Bilgisayar Programcılığı</td>
<td>300</td>
</tr>
<tr>
<td>Yönetim ve Organizasyon</td>
<td>Lojistik</td>
<td>200</td>
</tr>
</table>
```

**Örnek2:**

```
<% @page import="java.util.Date"%>
<%
Date tarih = new Date();
out.println("<p>Tarih:" + tarih);
out.println("<p>Gün:" + tarih.getDate());
out.println("<p>Ay:" + tarih.getMonth());
out.println("<p>Yıl:" + tarih.getYear());
%>
```

**Örnek3:**

```
<% @page import = "java.util.ArrayList" %>

<%
ArrayList<String>bolumler=new ArrayList<String>();
bolumler.add("Bilgisayar");
bolumler.add("Elektrik");
bolumler.add("Elektronik");
bolumler.add("İnşaat");
%>
<%
for(int i=0;i<bolumler.size();i++)
out.println("<p>Kodu:"+(i+1)+" "+bolumler.get(i));
//out.println("<hr>" +bolumler);
%>
```

**Örnek4:**

```
<% @page import="java.util.Random"%>
<%
    Random random = new Random();
    String soz[] = {"Barış", "HoşGörü", "Sevgi", "Sağlık", "Huzur"};
    int sozIndis = random.nextInt(6);

    String resimler[] = {"resim/r1.jpg", "resim/r2.jpg", "resim/r3.jpg", "resim/r4.jpg", "resim/r5.jpg" };
    int resimIndis = random.nextInt(resimler.length);

    String renkler[] = {"red", "blue", "pink", "black", "purple"};
    int center = random.nextInt(2);
%>

<body bgcolor="<%=renkler[random.nextInt(renkler.length)]%>">
<%
if (center == 0)
{
%>
<center>
<%
    }
%>
<font color="white">
    Günün Sözü:<%= soz[sozIndis]%>
</font>
<p>

</center>
</body>
```

### **<jsp:param> Etiketi:**

Bu etiketle diğer **JSP** etiketlerine **<jsp:include>**, **<jsp:forward>** vs. parametre gönderilebilir. Bu şekilde yönlendirilen ya da dahil edilen **JSP** sayfaları bu parametrelere ulaşabilir.

**<jsp:param name="Parametrenin adı" value="Parametrenin değeri" />**

### **<jsp:forward> Etiketi**

JSP sayfasına yapılan bir isteği başka bir sayfaya yönlendirmek için kullanılır. Bu etiketin bir özelliği de yönlendirme yapıldığında adresin değişmemesidir.

**<jsp:forward page="adres" />**

### **<jsp:include> Etiketi**

Bir **JSP** sayfası içerisinde başka bir **JSP** sayfasını çağdırmamızı sağlar. **<%@include...%>** etiketi ile aynı işlevi görür. Farklı olarak **<jsp:include>** etiketi ile dahil etme işlemi **request**(istek) geldiğinde gerçekleşirken, **<%@ include ...%>** etiketiyle dahil etme işlemi ise **JSP** sayfaları **Servlet**'e çevrilirken gerçekleşmektedir.

**<jsp:include page="Dahil edilecek sayfa adresi" />**

```
//orneka.jsp

<%@ page contentType="text/html; charset=UTF-8"%>
<body>
<jsp:forward page="ornekb.jsp">                                // <jsp:include page="ornekb.jsp">
<jsp:param name="ogrno" value="197102258" />
<jsp:param name="ad" value="Kenan" />
<jsp:param name="soyad" value="Yaramaz" />
<jsp:param name="bolum" value="Elektrik" />
</jsp:forward>                                                //<jsp:include>
</body>

//ornekb.jsp

<%@ page contentType="text/html; charset=UTF-8"%>
<body>
<h2>Öğrenci Bilgileri</h2><hr>
No   : <%=request.getParameter("ogrno") %><br>
Adı  : <%=request.getParameter("ad") %><br>
Soyadı: <%=request.getParameter("soyad") %><br>
Bölümü: <%=request.getParameter("bolum") %>
</body>
```

## <jsp:useBean> Etiketi

**JSP** sayfalarında **Java Bean**'leri çağırmak için kullanılır. Java Bean aslında Java'dan alışık olduğumuz sınıflardan başka bir şey değildir.

```
<jsp:useBean id="xxx" class="paketAdi.classAdi" scope="Session"/>
<jsp:setProperty name="xxx" property="Sınıftaki değişken adı" />
<jsp:getProperty name="xxx" property="Sınıftaki değişken adı" />
```

Eğer birden fazla değişkene aynı anda değer ataması yapmak istiyorsak '\*' kullanabiliriz.

```
<jsp:setProperty name="xxx" property="*" />
```

<pre>// orneka.jsp  &lt;form name="Form" method="post" action="ornekb.jsp"&gt;  &lt;p&gt; Adı.....: &lt;input type="text" name="ad"&gt;  &lt;p&gt; Vize Notu..: &lt;input type="text" name="vizem"&gt;  &lt;p&gt; Final Notu..: &lt;input type="text" name="finalim"&gt;  &lt;p&gt; &lt;input type="submit" value="Hesapla"&gt;  &lt;/form&gt;  //ornekb.jsp  &lt;%@ page contentType="text/html; charset=UTF-8"%&gt; &lt;body&gt; &lt;h3&gt;Sonuç Ekranı....&lt;/h3&gt; &lt;hr&gt;  &lt;jsp:useBean id="xxx" class="veri.Hesapla" scope="request"/&gt;  &lt;jsp:setProperty name="xxx" property="*" /&gt;  Sayın: &lt;jsp:getProperty name="xxx" property="ad"/&gt;  &lt;p&gt; Vize:&lt;jsp:getProperty name="xxx" property="vizem"/&gt;  &lt;p&gt; Final:&lt;jsp:getProperty name="xxx" property="finalim"/&gt;  &lt;p&gt; Sonuç:&lt;jsp:getProperty name="xxx" property="sonucum"/&gt;  &lt;/body&gt;</pre>	<pre>//Hesapla.java //class  package veri;  public class Hesapla {     private String ad="";     private int vizem=0, finalim=0, sonucum=0;      public String getAd()     {         return (ad);     }      public void setAd(String ad)     {         this.ad=ad;     }      public int getVizem()     {         return (vizem);     }      public void setVizem(int vizem)     {         this.vizem=vizem;     }      public int getFinalim()     {         return (finalim);     }      public void setFinalim(int finalim)     {         this.finalim=finalim;     }      public double getSonucum()     {         sonucum=(int) (vizem*0.4+finalim*0.6);         return sonucum;     } }</pre>
----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

## GÖMÜLÜ (IMPLICIT OBJECTS) NESNELER:

Gömülü nesneler, JSP sayfalarının **Servlet**'e çevrilmesi anında Sunucu tarafından oluşturulurlar. Gömülü olarak adlandırılmalarının sebebi budur. Sunucu bu nesneleri **\_jspService()** metodunun içerisinde oluşturur. Dolayısıyla Scriptlet tag içerisinde tekrar oluşturmaya gerek kalmadan olduğu gibi kullanabiliriz.

1. **Out**
2. **Exception**
3. **Response**
4. **Request**
5. **Session**
6. **Application**
7. **Page / PageContext**

**1. Out:** Ekranı içerik bastırmak için kullanılır. İçerik ve buffer ile ilgili metotları bulunur.

**2. Exception:** Hataları yakalayıp gerekli mesajı göstermek için kullanılır. Sadece **isErrorPage** özelliği **true** olan JSP sayfalarından erişilebilir.

**3. Response:** Yapılan bir istek sonrası sunucu tarafından kullanıcıya gönderilen cevaplar ile ilgili işlemlerle görevlidir.

**4. Request:** Bu nesnenin en temel amacı veri almaktır. Forma girilen veriler, yönlendirilen sayfada request nesnesi ile alınır ve üzerinde çeşitli işlemler yapılır.

**5. Session:** Oturum yönetimi için kullanılır. Oturumun açık kaldığı süre boyunca, kullanıcının bilgilerine bu nesne sayesinde her yerden erişilebilir. En çok kullanılan nesnelerden birisidir.

**6. Application:** Projemizin parametrelerine ulaşmada kullanılır. Örneğin; versiyonlar, url'ler, sunucu bilgileri.

**7. Page:** Yaygın kullanımı yoktur. **"this"** anahtar kelimesiyle eş anlamlıdır.

**8. PageContext:** Page, Request, Application ve Session seviyelerine ulaşmak için kullanılır.

## 1. OUT NESNESİ:

Ekrana veri basmak için kullanılır. Başlıca metotları şunlardır;

**void print()/println()** : Parametre olarak gönderilen veriyi ekrana basar.

```
<% out.print("JSP Programlama"); %>
```

```
<% out.println("JSP Programlama"); %>
```

**void newLine()** : Yeni bir satır açar. println() metodu ile aynı işlevi görür.

Diğer metotları açıklamadan önce buffer kavramına değinmek istiyorum. Buffer'ın kelime anlamı tampon'dur. Jsp sayfalarında ekrana basılan verinin boyutunu sınırlar. Varsayılan olarak 8kb'dir. Yine varsayılan olarak 8kb yetmediğinde otomatik olarak buffer tazelenir. Buna da flush denir.

**void clear()** : Ekrana o ana kadar ne yazdırılmışsa temizler. Yani buffer'ı sıfırlar.

**void clearBuffer()** : clear() metoduyla aynı işlevi görür. Bu metodun farkı eğer önceden flush() metodu çağrıldıysa exception(hata) fırlatır.

**flush()** : Bufferı tazeler. Sınırın aşılmasını sağlar. Eğer flush() metodu çağrılmaz ve sınır aşılsa "java.io.IOException: Error: JSP Buffer overflow" hatası fırlatılır.

**getBufferSize()** : Varsayılan buffer değerini (8kb=8192 byte) döndürür.

**getRemaining()** : Buffer'da ne kadar boş alan kaldığını anlamamızı sağlar.

**boolean isAutoFlush()** : Buffer dolduğunda otomatik olarak arttırılıp arttırılmayacağını "true" ya da "false" olarak tutar. Varsayılan olarak "true" dir.

//out1.jsp	//out2.jsp
<% @page contentType="text/html; charset=UTF-8"%>	<% @page contentType="text/html; charset=UTF-8"%>
<% out.print("Bu mesaj görünmeyecek..."); %>	<% out.print("Sizce bu yıl şampiyon olacak?..."); %>
<% out.clearBuffer(); %>	<% out.flush(); %>
<%="<p>Buffer Miktarı : " + out.getBufferSize() %>	<% out.print("<p><b>Galatasaray"); %>
<%="<p>Mustafa Kemal Üniversitesi") %>	<% out.clearBuffer(); %>
<%="<p>Kalan Buffer:" + out.getRemaining() %>	<% out.print("<p>Fenerbahçe"); %>
<% out.flush(); %>	<% out.flush(); %>
<%="<p>Flush sonrası Buffer:" + out.getRemaining() %>	<% out.clear(); %>
<% out.clear(); // Hata Buffer Boş %>	<% out.print("<p>Beşiktaş"); %>

## 2. EXCEPTION NESNESİ:

Bu nesneye sadece hata sayfası(Error Page) olarak belirlenmiş sayfalardan erişilebilir. Eğer bir **JSP** sayfasında hata oluşması bekleniyorsa, hata oluştuğunda hangi sayfaya gidileceğini **errorPage** belirlerken, **isErrorPage** ise gidilecek sayfayı hata sayfası olarak tanımlayarak başka bir **JSP** sayfasından hata oluştuğunda çağrılmasına izin vermektedir. Varsayılan değeri **false**'dir.

### Örnek:

```
// index.jsp      Harici web tarayıcılarında çalıştırılmalıdır
<% @ page contentType="text/html; charset=UTF-8"%>
<form action="display.jsp">
<p>Birinci Sayı: <input type="text" name="s1" />
<p>İkinci Sayı : <input type="text" name="s2" />
<p><input type="submit" value="Hesapla..." />
</form>
```

Birinci Sayı: 10

İkinci Sayı : 5

Hesapla...

```
// display.jsp

<% @ page contentType="text/html; charset=UTF-8"%>
<% @ page errorPage="error.jsp" %>
<%
int a=Integer.parseInt(request.getParameter("s1"));
int b=Integer.parseInt(request.getParameter("s2"));
int c=a/b;
out.print("<p>Birinci Sayı: "+a);
out.print("<p>İkinci Sayı : "+b);
out.print("<p>Bölme İşlemi: "+c);
%>
```

Birinci Sayı: 10

İkinci Sayı : 5

Bölme İşlemi: 2

```
// error.jsp

<% @ page contentType="text/html; charset=UTF-8"%>
<% @ page isErrorPage="true" %>
<h3>Üzgünüm, Hata Meydana Geldi!</h3>
Hata Mesajı: <%= exception %>
```

**Üzgünüm, Hata Meydana Geldi!**

Hata Mesajı: java.lang.NumberFormatException: For input string: "a"



### 3. RESPONSE NESNESİ:

Sunucunun istemciye cevap vermesini sağlar. Başlıca metotlar;

**setContentTypes:** Sayfa içeriğinin tipini belirler.

```
<%  
    response.setContentType("text/html");  
    response.setContentType("image/gif");  
    response.setContentType("image/png");  
    response.setContentType("application/pdf");  
%>
```

**setHeader:** Sayfanın cache/sunucu yüklenme durumunu belirler.

Sayfanın cache bellekten milisaniye cinsinden çağırmak istiyorsak;

```
<% response.setHeader ("Cache-Control", "max-age = 5000"); %>  
<% response.setHeader ("Cache-Control", "max-age = 0"); %>
```

Sayfamızı sürekli sunucudan çağırmak istiyorsak;

```
<%  
    response.setHeader("Cache-Control","no-cache");  
    response.setHeader("Pragma","no-cache");  
    response.setDateHeader ("Expires", 0);  
%>
```

**setIntHeader("Refresh", saniye):** Otomatik yenileme sağlar.

```
<%response.setIntHeader("Refresh", 1); %>
```

**sendError:** Sayfaya hata mesajı yazdırır.

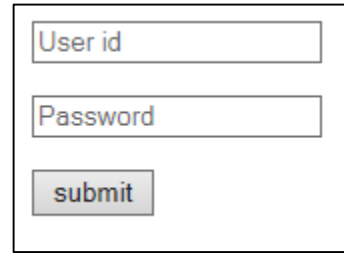
```
<% response.sendError(404, "Page not found error");%>
```

**sendRedirect:** Parametre olarak aldığı sayfaya yönlendirir.

```
<% response.sendRedirect("gidilecekSayfa.jsp"); %>
```

**// index.jsp**

```
<body>
<form action="control.jsp">
  <p><input placeholder="Kullanıcı Adı..." name="userid">
  <p><input placeholder="Parola..." name="password">
  <p><input type="submit" value="submit">
</form>
</body>
```

A login form with two input fields labeled 'User id' and 'Password', and a 'submit' button.

**// control.jsp**

```
<%
String userid=request.getParameter("userid");
String password=request.getParameter("password");
if (userid.equals("admin") && password.equals("admin"))
    response.sendRedirect("display.jsp");
else
    response.sendRedirect("error.jsp");
%>
```

**// display.jsp**

```
<% @ page contentType="text/html; charset=UTF-8"%>
<%
response.setHeader("Cache-Control", "no-cache");
response.setDateHeader("Expires", 0);
response.setIntHeader("Refresh", 5);

out.println("<h1>Web Sitemize Hoş Geldiniz...</h1>");

// response.setStatus(301); // Siteyi bulamazsan aşağıdaki siteye yönlendir
// response.setHeader( "Location", "http://www.google.com.tr");
// response.setHeader( "Connection", "close" );
%>
```

**// error.jsp**

```
<% @ page contentType="text/html; charset=UTF-8"%>
<body>
  <% response.sendError(404, "Giriş Yetkiniz Bulunmuyor...");%>
</body>
```

#### 4. REQUEST NESNESİ:

Request demek istek, bilgi demektir. Request nesnesinin görevi bir HTTP isteği anında JSP sayfasındaki verileri Sunucuya iletmektir. Bu veriler; çerezler, başlıklar ya da kullanıcı tarafından girilmiş parametreler olabilir. İster sunucudan, ister kullanıcıdan bilgi alımı konusunda Request nesnesinin metodlarını kullanacağız.

```
// request.jsp
<% @ page contentType="text/html; charset=UTF-8"%>
<%= "<p>Sunucu Adı : " + request.getServerName() %>
<%= "<p>Sunucu Çıkış Portu : " + request.getServerPort() %>
<%= "<p>Ziyaretçi IP Adresi : " + request.getRemoteAddr() %>
<%= "<p>Sunucu IP Adresi : " + request.getRemoteHost() %>
<%= "<p>Paket Gönderim Metodu : " + request.getMethod() %>
<%= "<p>Protokol Sürümü : " + request.getProtocol() %>
<%= "<p>Bölge Kodu : " + request.getLocale() %>
<%= "<p>Domain Adı : " + request.getContextPath() %>
<%= "<p>Aktif Dizin Yolu : " + request.getServletPath() %>
<%= "<p>Domain Dizin Yolu : " + request.getRequestURI() %>
<%= "<p>Sunucu Dizin Yolu : " + request.getRequestURL() %>
```

#### QueryString:

Adından anlaşılacağı gibi veri talep etmek için kullanılır. Gerek formlardan, gerek sorgulardan, gerekse çerezlerden veri almak için kullanılır. Bu yöntem kullanım kolaylığı ve sunucuya herhangi bir yük getirmedikten sıkça tercih edilmektedir. Ancak parametre ile taşınan veriler diğer kullanıcılar tarafından rahatlıkla erişebildiğinden güvenlik açığı oluşturmaktadır. Dolayısıyla bu yönetime çok fazla güvenip çok önemli veriler taşımamak gerekir. Diğer dezavantajı ise tarayıcıların URL uzunluğunu 2083 karakter olarak sınırlar.

**getParameter/ getParameterValues:** getParameter formdan veya url satırından gelen tek parametrelili değeri karşılarken, getParameterValues ise birden çok parametrelili (**checkbox** gibi) değerleri karşılayabilmektedir.

```
// index.jsp
<% @ page contentType="text/html; charset=UTF-8"%>
<% @ page import="java.net.URLEncoder"%>
<p><a href="display.jsp?adsoyad=Ali ŞimşekÇakan&meslek=Öğretmen">Gönder1...</a>
<p><a href="display.jsp?adsoyad=Ali ŞimşekÇakan&meslek=Öğretmen&Şoför">Gönder2...</a>
<p><a href="display.jsp?adsoyad=Ali ŞimşekÇakan&meslek=
<%=URLEncoder.encode("Öğretmen&Şoför","UTF-8")%>">Gönder3...</a>
<p><a href="http://www.mku.edu.tr/default.php?ad=Ayşe&soyad=Gül">Gönder4...</a>
<p><a href="http://www.mku.edu.tr/default.aspx?ad=Ayşe&soyad=Gül">Gönder5...</a>

//display.jsp
<% @ page contentType="text/html; charset=UTF-8"%>
<% @ page import="java.net.URLDecoder"%>
<% request.setCharacterEncoding("utf-8"); //Türkçe karakter sorunu %>

<%= "<p>Query String : " + request.getQueryString() %>
<%= "<p>Ad Soyad : " + request.getParameter("adsoyad") %>
<%= "<p>Meslek : " + URLDecoder.decode(request.getParameter("meslek")) %>
```

## Örnek:

```
// index.jsp
<a href="kontrol.jsp?kod=1">Otomotiv</a>
<a href="kontrol.jsp?kod=2">Mobilya</a>
<a href="kontrol.jsp?kod=3">Elektronik</a>

//kontrol.jsp
<% @ page contentType="text/html; charset=UTF-8"%>
<%
int kod=Integer.parseInt(request.getParameter("kod"));
if (kod==1) response.sendRedirect("otomotiv.jsp");
else if (kod==2) response.sendRedirect("mobilya.jsp");
else response.sendRedirect("elektronik.jsp");
%>

// otomotiv.jsp

<h1>Otomotiv Ürünler Sayfası...</h1>

// mobilya.jsp

<h1>Mobilya Ürünler Sayfası...</h1>

// elektronik.jsp

<h1>Elektronik Ürünler Sayfası...</h1>
```

[Otomotiv](#) [Mobilya](#) [Elektronik](#)

## Örnek:

```
// index1.jsp
<center>
<%
int sayfa;
if (request.getParameter("sayfa") == null) sayfa = 20;
else sayfa = Integer.parseInt(request.getParameter("sayfa"));
%>
<h1 style="font-size:200px">
<%=sayfa%>
</h1>
<a href="index1.jsp?sayfa=<%=sayfa-1 %>">
<<<<
</a>
---|---
<a href="index1.jsp?sayfa=<%= sayfa+1 %>">
>>>>
</a>
</center>
```

20

<<<< ---|--- >>>>

## Örnek:

```
// index2.jsp
<center>
<%
int sayfa,ilk,son;
if (request.getParameter("sayfa") == null) sayfa = 20;
    else sayfa = Integer.parseInt(request.getParameter("sayfa"));

if (sayfa<1) sayfa=1;
if (sayfa>=20) sayfa=20;
ilk=sayfa-5;
if (ilk<=1) ilk=1;
son=sayfa+5;
if (son > 20) son = 20;
%>

<a href="index2.jsp?sayfa=<%=sayfa-1 %>">
<<<< </a>

<%
for (int i=ilk;i<=sayfa-1;i++)
{
%>

<a href=" index2.jsp?sayfa=<%=i %>">
<%=i %> </a>

<%
}
%>

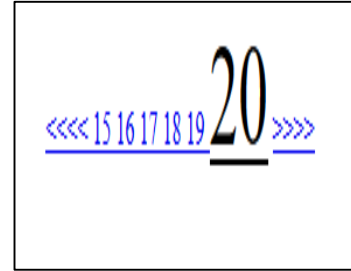
<a href="#" style="color:blue; font-size:50px">
<%=sayfa%></a>

<%
for (int i=sayfa+1;i<=son; i++)
{
%>

<a href=" index2.jsp?sayfa=<%=i%>">
<%=i %> </a>

<%
}
%>

<a href=" index2.jsp?sayfa=<%= sayfa+1 %>">
>>>> </a>
</center>
```



## FORM KULLANMAK:

Form vasıtasıyla bilgi göndermek için GET veya POST metodlarından birisi kullanılır. Ancak güvenlik gerektiren şifre gibi gizli bilgiler GET metodu yerine POST metoduyla gönderilmelidir. Ayrıca GET metodu ile veri gönderilirken max 256 karakter sınırlaması varken post metodunda sınırlama söz konusu değildir. Bilgiler GET metodunda url satırı üzerinde gönderilirken, POST metodunda http protokolü içine gömülür.

### Örnek:

//index.jsp

```
<% @ page contentType="text/html; charset=UTF-8"%>
```

```
<%
```

```
String kullanıcı = request.getParameter("user");
```

```
String şifre = request.getParameter("password");
```

```
if (kullanıcı!=null && şifre!=null)
```

```
{
```

```
if (kullanıcı.equals("btp") && şifre.equals("123"))
```

```
response.sendRedirect("display.jsp?user=" + kullanıcı + "&uyemi=evet");
```

```
else out.print("Giriş Yetkiniz Bulunmuyor...");
```

```
}
```

```
%>
```

```
<Form action="index.jsp" method="post">
```

```
<p>User.....:<input name="user">
```

```
<p>Password.....:<input type="password" name="password">
```

```
<p><input type="submit">
```

```
</Form>
```

//display.jsp

```
<% @ page contentType="text/html; charset=UTF-8"%>
```

```
<%
```

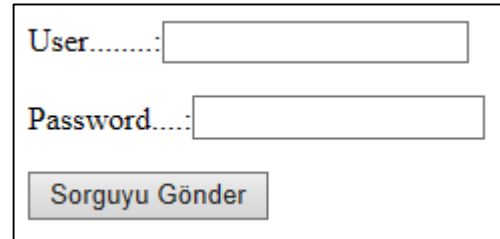
```
String kullanıcı = request.getParameter("user");
```

```
String uye = request.getParameter("uyemi");
```

```
if (uye == null) out.print("Giriş Yetkiniz Yok");
```

```
else if (uye.equals("evet")) out.print("Hoşgeldin..." + kullanıcı);
```

```
%>
```



The screenshot shows a web form with two input fields. The first field is labeled "User.....:" and the second field is labeled "Password.....:". Below these fields is a button labeled "Sorguyu Gönder".

**getParameterNames():** request ile birlikte gönderilen tüm parametrelerin değişken isimlerini **Enumeration** tipinde döndürür.

### Örnek:

// index.html

```
<Form action="display.jsp" method="post">
<p>Adı : <input type="text" name="Email"></input><br>
<p>Soyadı : <input type="text" name="Pass"></input><br>

<p>Cinsiyet
<input name="radioButton" type="radio" value="Erkek" checked="checked"/>Erkek
<input name="radioButton" type="radio" value="Kadın" />Kadın

<p>Hobiler
<input type="checkbox" name="hobi" value="Spor">Spor
<input type="checkbox" name="hobi" value="Muzik">Muzik
<input type="checkbox" name="hobi" value="Sinema">Sinema

<p><input type="submit" value="Gönder...">
</Form>
```

Adı :

Soyadı :

Cinsiyet ☒ Erkek ☐ Kadın

Hobiler ☐ Spor ☐ Muzik ☐ Sinema

// display.jsp

```
<% @ page contentType="text/html; charset=UTF-8"%>
<% @page import="java.util.*"%>

<%
Enumeration e = request.getParameterNames();
while (e.hasMoreElements())
{
String element = e.nextElement().toString();
out.println("<p>" + element);
}
%>
```

Email

Pass

radioButton

hobi

## Örnek:

// index.jsp

```
<% @page contentType="text/html" pageEncoding="UTF-8"%>
```

```
<style>
```

```
#uyeForm
```

```
{
```

```
padding: 25px; width: 500px;
```

```
border: 1px solid black;
```

```
margin-left: 400px; margin-top: 50px
```

```
}
```

```
</style>
```

```
<div id="uyeForm">
```

```
<Form action="display.jsp" method="post" target="_blank">
```

```
<p><input name="ad" Placeholder="Adınız..." id="ad" onkeyup="form_kontrol()">
```

```
<p><input name="email" placeholder="E-Mail" size="45" id="email" onkeyup="form_kontrol()">
```

```
<p>Cinsiyetiniz:
```

```
<input type="radio" name="cinsiyet" value="Kadın" id="cins1" onchange="form_kontrol()">
```

```
<label for="cins1" >Kadın</label>
```

```
<input type="radio" name="cinsiyet" value="Erkek" id="cins2" onchange="form_kontrol()">
```

```
<label for="cins2" >Erkek</label>
```

```
<p>Doğum Tarihiniz:
```

```
<select name="gun" id="gun" onchange="form_kontrol()">
```

```
<option value="">
```

```
<%
```

```
for(int i=1;i<=31;i++) { %>
```

```
<option value="<%= i %>"><% out.println(i); %>
```

```
<% } %>
```

```
</select>
```

```
<select name="ay" id="ay" onchange="form_kontrol()">
```

```
<option value="">
```

```
<% for(int i=1;i<=12;i++) { %>
```

```
<option value="<%= i %>"><% out.println(i); %>
```

```
<% } %>
```

```
</select>
```

```
<select name="yil" id="yil" onchange="form_kontrol()">
```

```
<option value="">
```

```
<% for(int i=2022;i>=1900;i--) { %>
```

```
<option value="<%= i %>"><% out.println(i); %>
```

```
<% } %>
```

```
</select>
```

The screenshot shows a web form with the following elements:

- Text input for name: "Kemal"
- Text input for email: "Yıldırım"
- Gender selection: "Cinsiyetiniz: ☐ Kadın ☒ Erkek"
- Birth date selection: "Doğum Tarihiniz: 14 6 1977" (using dropdown menus)
- Programming language selection: "Mesleki Birikim" with checkboxes for PHP (checked), ASP, JAVA (checked), and ANDROID.
- Goals section: "Hedefleriniz :" with a text area containing "Programcı Olmak".
- Buttons: "Gönder..." and "Temizle..."



```
// index.jsp
```

```
<p>Hedefleriniz :<p>
```

```
<textarea name="hedef" rows="5" cols="30" id="hedef" onkeyup="form_kontrol()"></textarea>
```

```
<p>Mesleki Birikiminiz:<p>
```

```
<input type="checkbox" name="dil" value="PHP" id="dil" onkeyup="form_kontrol()">PHP
```

```
<input type="checkbox" name="dil" value="ASP" id="dil" onkeyup="form_kontrol()">ASP
```

```
<input type="checkbox" name="dil" value="JAVA" id="dil" onkeyup="form_kontrol()">JAVA
```

```
<input type="checkbox" name="dil" value="C++" id="dil" onkeyup="form_kontrol()">C++
```

```
<p><input type="checkbox" name="kosullar" id="kosullar" onchange="form_kontrol()">
```

```
Kullanım koşullarını kabul ediyorum.
```

```
<p> <button type="submit" id="gonder" disabled>Gönder</button>
```

```
<input type="reset" value="Temizle...">
```

```
</Form>
```

```
</div>
```

```
<script>
```

```
function form_kontrol()
```

```
{
```

```
if(document.getElementById("ad").value.length == 0)
```

```
document.getElementById("gonder").disabled = true;
```

```
else if(document.getElementById("email").value.length == 0)
```

```
document.getElementById("gonder").disabled = true;
```

```
else if(document.getElementById("cins1").checked==false && document.getElementById("cins2").checked==false)
```

```
document.getElementById("gonder").disabled = true;
```

```
else if(document.getElementById("gun").value.length == 0)
```

```
document.getElementById("gonder").disabled = true;
```

```
else if(document.getElementById("ay").value.length == 0)
```

```
document.getElementById("gonder").disabled = true;
```

```
else if(document.getElementById("yil").value.length == 0)
```

```
document.getElementById("gonder").disabled = true;
```

```
else if(document.getElementById("hedef").value.length == 0)
```

```
document.getElementById("gonder").disabled = true;
```

```
else if(document.getElementById("dil").checked==false)
```

```
document.getElementById("gonder").disabled = true;
```

```
else if(document.getElementById("kosullar").checked == false)
```

```
document.getElementById("gonder").disabled = true;
```

```
else
```

```
document.getElementById("gonder").disabled = false;
```

```
}
```

```
</script>
```

```
// display.jsp
<% @page contentType="text/html" pageEncoding="UTF-8"%>

<%!
boolean bilgiKontrol(String veri)
{
    if (veri.trim().length() != 0) return true;
    else return false;
}
%>

<%
request.setCharacterEncoding("utf-8");//Türkçe karakter
String ad = request.getParameter("ad");
String email = request.getParameter("email");
String cinsiyet = request.getParameter("cinsiyet");
String gun = request.getParameter("gun");
String ay = request.getParameter("ay");
String yıl = request.getParameter("yıl");
String hedef = request.getParameter("hedef");
String[] programlar = request.getParameterValues("dil");

if (bilgiKontrol(ad)) out.print("<p>Adınız.....:" + ad);
else out.print("<p>Adınız Eksik...");

if (bilgiKontrol(email))
{
    if (emailKontrol(email)) out.print("<p>E-Mail Adresiniz.....:" + email);
    else out.print("<p>Geçersiz E-mail adresi...");
}
else out.print("<p>E-Mail Adresiniz Eksik...");

out.print("<p>Cinsiyetiniz.....:" + cinsiyet);
out.print("<p>Doğum Tarihiniz.....:" + gun + " / " + ay + " / " + yıl);

if (bilgiKontrol(hedef)) out.print("<p>Hedefleriniz.....:" + hedef);
else out.print("<p>Hedefleriniz Eksik...");

out.println("<p>Mesleki Birikiminiz.....:" );
for (String program : programlar) out.println(program);
%>

<%!
public boolean emailKontrol(String email)
{
    int at, nokta;
    boolean at_var = true, nokta_var = true;
    at = email.indexOf("@");
    if (at == -1) at_var = false;
    nokta = email.indexOf(".");
    if (nokta == -1) nokta_var = false;
    if (at_var & nokta_var) return true;
    else return false;
}
%>
```

## **5. VERİ TABANI NEDİR?**

Veri tabanı, birbiriyle ilişkili verilerin bütünleşik olarak tutulduğu alandır. Daha detaylı bir tanımla, veriyi yönetmek ve sunmak için kullanılan tablolar, formlar, sorgular ve raporlardan oluşan nesneler topluluğu olarak ifade edilebilir. Veritabanı, aslında veri sorgulama sistemiyle donatılmış arşivden başka bir şey değildir.

### **CREATE DATABASE dersler**

```
CREATE TABLE notlar (id INT PRIMARY KEY IDENTITY(1,1),  
    ad VARCHAR(50) NOT NULL, soyad VARCHAR(50) NOT NULL,  
    dersad VARCHAR(50) NULL, vize INT NULL, final INT NULL)
```

```
INSERT INTO notlar (ad, soyad, dersad, vize, final)  
    VALUES ('Murat', 'Toker', 'JAVA', 70, 85)
```

```
INSERT INTO notlar VALUES ('Veysel', 'Demir', 'PHP', 60, 60)
```

```
INSERT INTO notlar (ad,soyad,dersad) VALUES ('Sema', 'Kara', 'ASP')
```

```
UPDATE notlar SET final=60
```

```
UPDATE notlar SET final=60 WHERE final<50
```

```
DELETE from notlar
```

```
DELETE * from notlar WHERE dersad IS NULL
```

```
SELECT * FROM notlar
```

```
SELECT id, ad, soyad, dersad, vize, final FROM notlar
```

```
SELECT id as 'Öğrenci Numarası', dersad as 'Dersin Adı', vize, final FROM notlar
```

```
SELECT id, ad, soyad FROM notlar WHERE dersad='JAVA'
```

```
SELECT * FROM notlar WHERE final=40 OR final=85
```

```
SELECT * FROM notlar WHERE Final IN (40, 85)
```

```
SELECT * FROM notlar WHERE dersad IN ('PHP', 'JAVA')
```

```
SELECT * FROM notlar WHERE vize>=60 AND vize<=80
```

```
SELECT * FROM notlar WHERE vize BETWEEN 60 AND 80
```

**SELECT \* FROM notlar WHERE vize IS NULL**

**SELECT \* FROM notlar WHERE dersad='JAVA' AND final IS NOT NULL**

**SELECT id, ad as 'Adı', soyad as 'Soyadı', ISNULL(dersad,'') as 'Ders Adı',  
ISNULL(vize,0) as 'Vize', ISNULL(final,0) as 'Final' FROM notlar**

**SELECT \*FROM notlar WHERE ad LIKE 'A%'**

**SELECT \*FROM notlar WHERE ad LIKE '%A'**

**SELECT \*FROM notlar WHERE ad LIKE '%A%'**

**SELECT \* FROM notlar ORDER BY ad ASC**

**SELECT \* FROM notlar ORDER BY ad, soyad DESC**

**SELECT \* FROM notlar WHERE dersad='JAVA' ORDER By id**

**SELECT dersad as 'Verilen Dersler' FROM notlar**

**SELECT DISTINCT dersad as 'Verilen Dersler' FROM notlar**

**SELECT COUNT(id) as 'Toplam Öğrenci Sayısı' FROM notlar**

**SELECT COUNT(\*) as 'Java Öğrenci Sayısı' FROM notlar Where dersad='JAVA'**

**SELECT COUNT(dersad) as 'Ders Sayısı' FROM notlar**

**SELECT COUNT(DISTINCT dersad) as 'Ders Sayısı' FROM notlar**

**SELECT MIN(vize) as 'En Düşük Vize Notu' FROM notlar**

**SELECT MAX(final) as 'JAVA Final Notu' FROM notlar where dersad='JAVA'**

**SELECT SUM (final) as 'Java Final Toplamı' FROM notlar Where dersad='JAVA'**

**SELECT AVG (final) as 'PYTHON Ortalama' FROM notlar Where dersad='PYTHON'**

**Select id,ad,soyad,vize,final, vize\*0.4+final\*0.6 as 'Ortalama' from notlar**

**SELECT dersad, count (\*) as 'Öğrenci Sayısı' FROM notlar GROUP BY dersad**

**SELECT dersad, AVG( vize ) as 'Vize Ortalaması' FROM notlar GROUP BY dersad**

## JDBC(Java DataBase Connectivity):

JDBC ile herhangi bir veritabanına bağlanarak SQL komutları ile verilere erişebildiğimiz bir yapıdır. JDBC programdan bağımsız bir şekilde yazılarak farklı programlarda kullanılabilir. Veri tabanına bağlanmak için genellikle şu adımlar takip edilmelidir;

1. Veritabanı kütüphanesi eklenir. **import java.sql.\*;**
2. Bağlantı için gerekli JDBC sürücüleri WEB-INF/lib klasörüne eklenmelidir.

**Class.forName("com.mysql.jdbc.Driver");**

3. Veritabanı yolu olarak MYSQL 'in kullandığı port numarası 3306 varsayılan olarak gelir.

**String url = "jdbc:mysql://localhost:3306/veritabanıadi";**

4. Veri tabanına bağlanılır.

**Connection con = DriverManager.getConnection(url,"kullaniciadi","sifre");**

### Örnek: mySQL Bağlanmak

```
// baglan.jsp
<% @ page import="java.sql.ResultSet" %>
<% @ page import="java.sql.SQLException" %>
<% @ page import="java.sql.Statement" %>
<% @ page import="java.sql.Connection" %>
<% @ page import="java.sql.DriverManager" %>
<%
Connection connect = null;
Statement s = null;
try
{
    Class.forName("com.mysql.jdbc.Driver");
    connect = DriverManager.getConnection("jdbc:mysql://localhost/okul" + "?user=admin&password=admin");
    if (connect != null) out.println("Bağlantı Sağlandı...");
    else out.println("Bağlantı Sağlanamadı...");
}
catch (Exception e)
{
    out.println(e.getMessage());
    e.printStackTrace();
}
%>
```

## 5. Statement / PreparedStatement sorguları oluşturulur.

### Statement Nesnesi:

Temel SQL ifadelerini çalıştırır. Bu temel ifadelerden en önemlileri Select, Insert, Update, Delete gibi ifadelerdir. Statement Nesnesi sayesinde SQL ifadelerinin bizlere değer döndürme özelliği bulunmaktadır. Statement nesnemizin içine Sql ifadesi yazadıktan sonra bu ifadeyi executeUpdate parametresi ile göndereceğiz.

```
Statement state;  
String sorgu = "insert into ogrenci (id , ad , soyad) " + "VALUES (1 , 'Emine ' , 'Kahraman ' )";  
state = conn.createStatement();  
state.executeUpdate(sorgu);  
state.close();  
conn.close();
```

### PreparedStatement Nesnesi:

Statement nesnesinde programımız üzerinde ifade derlenip veritabanımız sadece sorgulama işlemini gerçekleştirmektedir. PreparedStatement ise, verileri önceden derlenmiş olan preparedstatment verilerimizi tekrar tekrar göndererek işlemleri daha rahat yapmamız mümkün olmaktadır. PreparedStatement nesnesine ifadeler parametre ile yollanırken tablomuzdaki veri tipi ile uyum sağlaması açısından **setInt**, **setString** veya **setDouble** gibi veri tipleri kullanılmalıdır. Önceden derlenmiş bu verilere bu yöntemle erişmek hız ve düşük kaynak bakımından avantaj sağlamaktadır.

```
String sorgu = "insert into ogrenci(id, ad, soyad) VALUES" + "(? , ? , ?)";  
PreparedStatement preState = connection.prepareStatement(sorgu);  
preState.setInt(1, 1);  
preState.setString(2, "Emine");  
preState.setString(3, "Kahraman");  
preState.executeUpdate();  
preState.close();  
conn.close();
```

**6. ResultSet:** Statements nesnelerini kullanarak SQL sorgusunu çalıştırdıktan sonra veritabanından alınan verileri tutmak için bu nesneler kullanılır.

```
Statement state = conn.createStatement();  
ResultSet res = state.executeQuery("Select * From ogrenci");  
while (res.next())  
{  
    out.print(resultSet.getInt("id"));  
    out.print(resultSet.getString("ad"));  
    out.print(resultSet.getString("soyad"));  
}  
state.close(); res.close();
```

**7. Son olarak oluşturulan bağlantı / statement / resultset nesneleri kapatılır.**

**con.close();**

## Transaction Kavramı:

Transaction, en küçük işlem yığınıdır. Transaction, bütünlük, tutarlılık, bağımsızlık ve dayanıklılık özelliği sağlar. Bir grup işlemin arka arkaya yapılırken işlemlerden bir tanesi bile başarısız olsa bu grupta yapılan tüm işlemler geri alınır(rollback), bu grupta yer alan tüm işlemler başarı ile gerçekleşmesi (commit) gerekmektedir. Tüm işlemler başarılı ise transaction içindeki tüm veri değişiklikleri sisteme yansır. Ya hep, ya hiç mantığı vardır.

Transaction özellikle bankacılık işlemlerinde sıkça kullanılır. Örneğin bir öğrenci ev sahibinin hesabına 600 TL kira ödeyecektir. Öğrenci para yatırma işlemini onayladığında önce 600 TL bakiyesinden düşürülür daha sonra ev sahibinin hesabına yatırılır. Eğer bu işlemler gerçekleşirken herhangi bir nedenden dolayı kira yatırılmasaydı ev sahibinin hesabına para yatmaz ve öğrencinin hesabından bakiye düşmez. İşte bu gibi kritik işlemlerde yani bir bütünlük olması gereken işlemlerde Transaction kullanılır.

### **setAutoCommit():**

JDBC'nin veritabanı bağlantısını otomatik olarak ya da bizim istediğimiz durumlarda bağlanmasını sağlamak için kullanırız.

**conn.setAutoCommit(false);** // Otomatik bağlantı kapatılır.

**conn.setAutoCommit(true);** // Otomatik bağlantı açılır.

**commit():** İşlemleri toplu şekilde veritabanına yansıtılmasını sağlar. **conn.commit();**

**rollback():** commit sırasında herhangi bir hata olduğunda tüm işlemleri sırası ile geri alarak, işlemin yarısında oluşan sorunlardan etkilenmemeyi sağlar. **conn.rollback();**

```
try
{
    conn.setAutoCommit(false);
    connection.commit();
}
catch (SQLException e)
{
    try
    {
        connection.rollback();
        System.out.println("Rollback");
    }
    catch (SQLException e1)
    {
        e1.printStackTrace(System.err);
    }
}
```

## Örnek: mySQL Mimarisi

```
<% @ page contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
```

```
<% @ page import="java.sql.*"%>
```

```
<%!
```

```
public static final String user = "root";
```

```
public static final String password = "";
```

```
public static final String driver = "com.mysql.jdbc.Driver";
```

```
public static final String database = "jdbc:mysql://localhost:3306/okulumuz";
```

```
public Connection baglan;
```

```
%>
```

```
<%
```

```
try
```

```
{
```

```
Class.forName(driver);
```

```
out.println("<p>JDBC Driver Bulundu...");
```

```
}
```

```
catch(ClassNotFoundException e)
```

```
{
```

```
out.println("<p>JDBC Driver Bulunamadı...");
```

```
e.printStackTrace(); //Hata Göster
```

```
}
```

```
try
```

```
{
```

```
baglan= DriverManager.getConnection(database, user, password);
```

```
out.println("<p>Bağlantı Başarılı...");
```

```
}
```

```
catch(SQLException e)
```

```
{
```

```
out.println("<p>Bağlantı Başarısız...");
```

```
e.printStackTrace(); //Hata Göster
```

```
}
```

```
%>
```

```
<%
```

```
out.print(ogrenciGetir());
```

```
// out.print(ogrenciGetir("Ali"));
```

```
// out.print(ogrenciGetir(2));
```

```
// out.print(ogrenciGetir(2,"A%"));
```

```
// out.print(ogrenciEkle());
```

```
// out.print(ogrenciEkle("Elif", "Demir", "Muhasebe"));
```

```
// out.print(ogrenciCuncelle());
```

```
// out.print(ogrenciCuncelle("Ali","Hukuk"));
```

```
// out.print(ogrenciSil());
```

```
// out.print(ogrenciSil("Ali"));
```

```
// out.print(ogrenciSil(3,8));
```

```
%>
```

Database: **okulumuz**

Table: **ogrenciler**

Field:

ID	int (3)	PrimaryKey	AutoIncrement
Ad / Soyad / bolum			Text



```

<%!
// Parametresiz Kayıt Listeleme İşlemleri
public String ogrenciGetir()
{
String liste="<p>ogrno adı soyadı bölümü<br>-----<br>";
try
{
String query="SELECT * FROM ogrenciler";
// String query="SELECT * FROM ogrenciler WHERE ad LIKE 'A%'";
Statement st=baglan.createStatement();
ResultSet rs=st.executeQuery(query);
while (rs.next()) //Kayıtları listele
{
liste+=rs.getInt("ogrno")+ " "+rs.getString("ad")+ " "+rs.getString("soyad")+ " "+rs.getString("bolum")+ "<p>";
}
}
catch (SQLException e)
{
e.printStackTrace();
}
return(liste);
}
%>

<%!
// Parametrelili Kayıt Listeleme İşlemleri
public String ogrenciGetir(String isim)           //(int id)           //(int id , String isim)
{
String liste="<p>ogrno adı soyadı bölümü<br>-----<br>";
try
{
String query1="SELECT * FROM ogrenciler WHERE ad = " + isim + " ";
Statement st=baglan.createStatement();
ResultSet rs=st.executeQuery(query1);

/* String query2="SELECT * FROM ogrenciler WHERE ogrno=?";
PreparedStatement prst = baglan.prepareStatement(query2);
prst.setInt(1, id); ResultSet rs=prst.executeQuery(); */

/* String query3="SELECT * FROM ogrenciler WHERE ogrno=? AND ad LIKE ?";
PreparedStatement prst = baglan.prepareStatement(query3);
prst.setInt(1, id); prst.setString(2, isim);
ResultSet rs=prst.executeQuery(); */

while (rs.next()) //Kayıtları listele
{
liste+=rs.getInt("ogrno")+ " "+rs.getString("ad")+ " "+rs.getString("soyad")+ " "+rs.getString("bolum")+ "<p>";
}
}
catch (SQLException e)
{
e.printStackTrace();
}
return(liste);
}
%>

```

```

<%!
// Parametresiz Kayıt Ekleme İşlemleri
public String ogrenciEkle()
{
    String ekle="";
    try
    {
        String query="INSERT INTO ogrenciler (ad,soyad,bolum) VALUES('aaa','bbb','ccc')";
        Statement st= baglan.createStatement();
        ekle=st.executeUpdate(query1)+" adet kayıt eklendi...<p>";
    }
    catch (SQLException e)
    {
        e.printStackTrace();
    }
    return(ekle);
}
%>

<%!
// Parametrelili Kayıt Ekleme İşlemleri
public String ogrenciEkle(String isim, String soyisim,String bolumu)
{
    String ekle="";
    try
    {

        // Birinci Yol
        String query1="INSERT INTO ogrenciler (ad,soyad,bolum) VALUES( '"+isim+"', '"+soyisim+"', '"+bolumu+"' )";
        Statement st= baglan.createStatement();
        ekle=st.executeUpdate(query1)+" adet kayıt eklendi...<p>";

        /*
        // İkinci Yol
        String query2 = "INSERT INTO ogrenciler(ad,soyad,bolum) VALUES (?,?,?)";
        PreparedStatement prst = baglan.prepareStatement(query2);
        prst.setString(1, isim);
        prst.setString(2, soyisim);
        prst.setString(3, bolumu);
        ekle=prst .executeUpdate()+" adet kayıt eklendi...<p>";
        */
    }
    catch (SQLException e)
    {
        e.printStackTrace();
    }
    return(ekle);
}
%>

```

```

<%!
// Parametresiz Kayıt Güncelleme İşlemleri
public String ogrenciCuncelle()
{
    String guncelle="";
    try
    {
        // Birinci Yol
        String query = "UPDATE ogrenciler SET bolum='Matematik' WHERE ad='Ayşe'";
        Statement st= baglan.createStatement();
        guncelle=st.executeUpdate(query)+" adet kayıt güncellendi...<p>";
    }
    catch (SQLException e)
    {
        e.printStackTrace();
    }
    return(guncelle);
}
%>

<%!
// Parametrelili Kayıt Güncelleme İşlemleri
public String ogrenciCuncelle(String isim, String bolumu)
{
    String guncelle="";
    try
    {

        // Birinci Yol
        String query1 = "UPDATE ogrenciler SET bolum = " + bolumu + " WHERE ad=" + isim + " ";
        Statement st= baglan.createStatement();
        guncelle=st.executeUpdate(query1)+" adet kayıt güncellendi...<p>";

        /*
        // İkinci Yol
        String query2 = "UPDATE ogrenciler SET bolum = ? WHERE ad = ?";
        PreparedStatement prst = baglan.prepareStatement(query2);
        prst.setString(1, bolumu); prst.setString(2, isim);
        guncelle=prst.executeUpdate()+" adet kayıt güncellendi...<p>";
        */
    }
    catch (SQLException e)
    {
        e.printStackTrace();
    }
    return(guncelle);
}
%>

```

```

<%!
// Parametresiz Kayıt Silme İşlemleri
public String ogrenciSil()
{
    String sil="";
    try
    {
        String query="DELETE FROM ogrenciler WHERE ad='aaa' ";
        Statement st= baglan.createStatement();
        sil=st.executeUpdate(query)+" adet kayıt silindi...";
    }
    catch (SQLException e)
    {
        e.printStackTrace();
    }
    return(sil);
}
%>

<%!
// Parametrelili Kayıt Silme İşlemleri
public String ogrenciSil(String bolumu)           // (int ilk , int son)
{
    String sil="";
    try
    {
        // Birinci Yol
        String query1="DELETE FROM ogrenciler WHERE bolum = '" + bolumu + "'";
        Statement st= baglan.createStatement();
        sil=st.executeUpdate(query1)+" adet kayıt silindi...";

        /*
        // İkinci Yol
        String query2="DELETE FROM ogrenciler WHERE ogrno>=? AND ogrno<=?";
        PreparedStatement prst = baglan.prepareStatement(query2);
        prst.setInt(1, ilk);
        prst.setInt(2, son);
        sil=prst.executeUpdate()+" adet kayıt silindi...";
        */
    }
    catch (SQLException e)
    {
        e.printStackTrace();
    }
    return(sil);
}
%>

```

## Örnek: Katmansız Veri Tabanı Mimarisi:

Database: **sirketim**

Table: **admin**

username : varchar(10) PrimaryKey

password : varchar(10)

Table: **birimler**

birimID : int (3) PrimaryKey Auto\_Increment

birimAD : varchar(50)

Table: **meslekler**

meslekID : int (3) PrimaryKey Auto\_Increment

meslekAD : varchar(50)

Table: **personel**

sicilNo : int (3) PrimaryKey Auto\_Increment

ad / soyad : varchar(50)

birimKod / gorevKod : int(3) / maas: int(6)

// **database.jsp**

```
<% @page contentType="text/html" pageEncoding="UTF-8"%>
```

```
<% @page import="java.sql.*"%>
```

```
<%!
```

```
public Connection baglan=null;
```

```
public Statement st=null;
```

```
public PreparedStatement prst=null;
```

```
public ResultSet rs=null;
```

```
public String query=null;
```

```
%>
```

```
<%
```

```
try
```

```
{
```

```
Class.forName("com.mysql.jdbc.Driver"); // out.println("<p>JDBC Bulundu");
```

```
}
```

```
catch(ClassNotFoundException e)
```

```
{
```

```
e.printStackTrace(); // out.println("<p>JDBC Bulunamadı");
```

```
}
```

```
try
```

```
{
```

```
baglan=DriverManager.getConnection("jdbc:mysql://localhost:3306/sirketim" , "root" , "");
```

```
// out.println("<p>Bağlandı");
```

```
}
```

```
catch(SQLException e)
```

```
{
```

```
e.printStackTrace(); // out.println("<p>Bağlanamadı");
```

```
}
```

```
%>
```

```
// index.jsp
```

```
<% @page contentType="text/html" pageEncoding="UTF-8"%>
```

```
<% @include file="database.jsp"%>
```

```
<form method="post" action="index.jsp?action=login">
```

```
<table border="1" align="center" width="300">
```

```
<tr>
```

```
<th colspan="2">Yönetici Giriş Ekranı</th>
```

```
</tr>
```

```
<tr>
```

```
<th>Kullanıcı Adı</th>
```

```
<th><input name="kullanici"></th>
```

```
</tr>
```

```
<tr>
```

```
<th>Parola</th>
```

```
<th><input name="parola"></th>
```

```
</tr>
```

```
<tr>
```

```
<th colspan="2">
```

```
<button type="submit"><b>Login...</b></button>
```

```
</th>
```

```
</tr>
```

```
</table>
```

```
</form>
```

```
<%
```

```
if (request.getParameter("action")!=null)
```

```
{
```

```
try
```

```
{
```

```
String kimlik=request.getParameter("kullanici");
```

```
String sifre=request.getParameter("parola");
```

```
query="SELECT * FROM admin WHERE username='"+kimlik+"' AND password='"+sifre+"'";
```

```
st=baglan.createStatement();
```

```
rs=st.executeQuery(query);
```

```
if (!rs.next())
```

```
out.println("<font color=red>Geçersiz Bilgiler...!</font>");
```

```
else
```

```
{
```

```
session.setAttribute("kimlik",rs.getString("username"));
```

```
response.sendRedirect("list.jsp");
```

```
}
```

```
}
```

```
catch(SQLException e)
```

```
{
```

```
out.println("Bağlantı Gerçekleşmedi...");
```

```
}
```

```
}
```

```
%>
```

Yönetici Giriş Ekranı...	
Kullanıcı Adı	<input type="text"/>
Parola	<input type="password"/>
<input type="button" value="Login..."/>	

```
// list.jsp
```

```
<% @page contentType="text/html" pageEncoding="UTF-8"%>
```

```
<% @include file="database.jsp"%>
```

```
<%
```

```
Object oturum=session.getAttribute("kimlik");
```

```
if (oturum==null)
```

```
response.sendRedirect("index.jsp");
```

```
%>
```

```
<form method="post" action="list.jsp">
```

```
<table border="1" cellpadding="0" align="center" width="700" height="50">
```

```
<tr>
```

```
<th>
```

```
<input name="filtre" placeholder="Adını veya Soyadını Gir...">
```

```
<button type="submit"><b>Filtrele</b></button>
```

```
</th>
```

```
</tr>
```

```
</table>
```

```
</form>
```

```
<%
```

```
request.setCharacterEncoding("utf-8");
```

```
try
```

```
{
```

```
String sorgu=request.getParameter("filtre");
```

```
if (sorgu!=null)
```

```
query="SELECT * FROM personel WHERE ad LIKE '%" +sorgu+"%'OR soyad LIKE '%" +sorgu+"%'";
```

```
else
```

```
query="SELECT * FROM personel";
```

```
st=baglan.createStatement();
```

```
rs=st.executeQuery(query);
```

```
%>
```

Sicil No	Adı	Soyadı	İşlemler...		
1	Mustafa	Demir	<input type="button" value="Detail"/>	<input type="button" value="Update"/>	<input type="button" value="Delete"/>
2	Fatma	Korkmaz	<input type="button" value="Detail"/>	<input type="button" value="Update"/>	<input type="button" value="Delete"/>
3	Ali	Güneş	<input type="button" value="Detail"/>	<input type="button" value="Update"/>	<input type="button" value="Delete"/>
4	Orhan	Yarar	<input type="button" value="Detail"/>	<input type="button" value="Update"/>	<input type="button" value="Delete"/>
5	Hatice	Uzun	<input type="button" value="Detail"/>	<input type="button" value="Update"/>	<input type="button" value="Delete"/>
6	Mustafa	Karahan	<input type="button" value="Detail"/>	<input type="button" value="Update"/>	<input type="button" value="Delete"/>
9	Pamuk	Karahan	<input type="button" value="Detail"/>	<input type="button" value="Update"/>	<input type="button" value="Delete"/>
<input type="button" value="Insert"/>					

```
// list.jsp
<table border="1" cellspacing="0" align="center" width="700">
  <tr>
    <th>Sicil No</th>
    <th>Adı</th>
    <th>Soyadı</th>
    <th colspan="4">İşlemler...</th>
  </tr>

  <% while (rs!=null && rs.next()) { %>

    <tr>
      <th width="150"><%=rs.getInt("sicilNo")%></th>
      <th width="150"><%=rs.getString("ad")%></th>
      <th width="150"><%=rs.getString("soyad")%></th>

      <th>
        <a href="detail.jsp?ID=<%=rs.getInt("sicilNo")%>">
          <button><b>Detail</b></button>
        </a>
      </th>

      <th>
        <a href="delete.jsp?ID=<%=rs.getInt("sicilNo")%>">
          <button><b>Delete</b></button>
        </a>
      </th>

      <th>
        <a href="updateform.jsp?ID=<%=rs.getInt("sicilNo")%>">
          <button><b>Update</b></button>
        </a>
      </th>
    </tr>

    <% } %>

    <tr>
      <th colspan="8">
        <a href="insertform.jsp">
          <button><b>Insert</b></button>
        </a>
      </th>
    </tr>
  </table>

  <%
  }
  catch(Exception e)
  {
    out.print("Kayıt Listelenemedi...");
  }
  %>
```



// detail.jsp

```
<% @page contentType="text/html" pageEncoding="UTF-8"%>

<% @include file="database.jsp"%>

<%

Object oturum=session.getAttribute("kimlik");

if (oturum==null) response.sendRedirect("index.jsp");

%>

<%

try

{

int gelenID=Integer.parseInt(request.getParameter("ID"));

query="SELECT sicilNo,ad,soyad,birimAd,meslekAd,maas FROM personel "

+ "INNER JOIN birimler ON (personel.birimKod=birimler.birimId) "

+ "INNER JOIN meslekler ON (personel.gorevKod=meslekler.meslekId) "

+ "WHERE sicilNo="+gelenID;

st=baglan.createStatement();

rs=st.executeQuery(query);

if (rs!=null) rs.next();

%>
```

```
// detail.jsp
```

```
<table border="1" cellspacing="0" align="center" width="350" height="250">
```

```
  <caption><b>Personel Detay Ekranı...</b></caption>
```

```
  <tr>
```

```
    <th align="left">Personel Sicil No</th>
```

```
    <th align="left"><%=rs.getInt("sicilNo")%></th>
```

```
  </tr>
```

```
  <tr>
```

```
    <th align="left">Personel Adı</th>
```

```
    <th align="left"><%=rs.getString("ad")%></th>
```

```
  </tr>
```

```
  <tr>
```

```
    <th align="left">Personel Soyadı</th>
```

```
    <th align="left"><%=rs.getString("soyad")%></th>
```

```
  </tr>
```

```
  <tr>
```

```
    <th align="left">Personel Birimi</th>
```

```
    <th align="left"><%=rs.getString("birimad")%></th>
```

```
  </tr>
```

```
  <tr>
```

```
    <th align="left">Personel Mesleği</th>
```

```
    <th align="left"><%=rs.getString("meslekad")%></th>
```

```
  </tr>
```

```
  <tr>
```

```
    <th align="left">Personel Maaşı</th>
```

```
    <th align="left"><%=rs.getInt("maas")%></th>
```

```
  </tr>
```

```
  <tr>
```

```
    <th colspan="2">
```

```
    <a href="list.jsp"> <button><b>Kayıt Listele...</b></button> </a>
```

```
  </th>
```

```
  </tr>
```

```
</table>
```

```
<%
```

```
  }
```

```
  catch(SQLException e)
```

```
  {
```

```
    out.print("Kayıt Görüntülenemedi...");
```

```
  }
```

```
%>
```

Personel Detay Ekranı...	
Sicil No	3
Adı	Ali
Soyadı	Güneş
Birim Adı	Pazarlama
Meslek Adı	Mutemet
Maaşı	5500
<a href="#">Kayıt Listele...</a>	

```
// insertform.jsp
```

```
<% @page contentType="text/html" pageEncoding="UTF-8"%>
```

```
<% @include file="database.jsp"%>
```

```
<%
```

```
Object oturum=session.getAttribute("kimlik");
```

```
if (oturum==null) response.sendRedirect("index.jsp");
```

```
try
```

```
{
```

```
%>
```

Personel Kayıt Formunu Doldurunuz...	
Personel Adı	<input type="text"/>
Personel Soyadı	<input type="text"/>
Personel Birimi	<input type="text" value="Finans"/>
Personel Mesleği	<input type="text" value="Avukat"/>
Personel Maaşı	<input type="text"/>
<input type="button" value="Insert..."/>	

```
<form method="post" action="insert.jsp">
```

```
<table border="1" cellpadding="0" align="center" width="350" height="250">
```

```
<caption><b>Personel Kayıt Formunu Doldurunuz...</b></caption>
```

```
<tr>
```

```
<th align="left">Personel Adı</th>
```

```
<th align="left"><input name="ad"></th>
```

```
</tr>
```

```
<tr>
```

```
<th align="left">Personel Soyadı</th>
```

```
<th align="left"><input name="soyad"></th>
```

```
</tr>
```

```
<tr>
```

```
<th align="left">Personel Birimi</th>
```

```
<th align="left">
```

```
<select name="birimAd" style="width:125px">
```

```
<%
```

```
query="select * from birimler";
```

```
st=baglan.createStatement();
```

```
rs=st.executeQuery(query);
```

```
while (rs!=null && rs.next())
```

```
{
```

```
%>
```

```
<option value="<%=rs.getInt("birimId")%>"><%=rs.getString("birimAd")%></option>
```

```
<% } %>
```

```
</select>
```

```
</th>
```

```
</tr>
```

```
// insertform.jsp
```

```
<tr>
  <th align="left">Personel Mesleği</th>
  <th align="left">
    <select name="meslekAd" style="width:125px">
      <%
        query="select * from meslekler";
        st=baglan.createStatement();
        rs=st.executeQuery(query);
        while (rs!=null && rs.next())
        {
          <option value="<%=rs.getInt("meslekId")%>"><%=rs.getString("meslekAd")%></option>
        } %>
      </select>
    </th>
</tr>

<tr>
  <th align="left">Personel Maaşı</th>
  <th align="left"><input name="maas"></th>
</tr>

<tr>
  <th colspan="2">
    <button type="submit"><b>Insert...</b></button>
  </th>
</tr>
</table>
</form>

<%
}
catch(SQLException e)
{
  out.print("Kayıt Görüntülenemedi...");
}
%>
```

```
// insert.jsp

<% @page contentType="text/html" pageEncoding="UTF-8"%>
<% @include file="database.jsp"%>

<%
Object oturum=session.getAttribute("kimlik");
if (oturum==null) response.sendRedirect("index.jsp");

request.setCharacterEncoding("utf-8");

String isim=request.getParameter("ad");
String soyisim=request.getParameter("soyad");
int birim=Integer.parseInt(request.getParameter("birimAd"));
int meslek=Integer.parseInt(request.getParameter("meslekAd"));
int maas=Integer.parseInt(request.getParameter("maas"));

try
{
query="INSERT INTO personel (ad,soyad,birimkod,gorevKod,maas) VALUES(?,?,?,?,?)";
prst=baglan.prepareStatement(query);
prst.setString(1,isim);
prst.setString(2,soyisim);
prst.setInt(3,birim);
prst.setInt(4,meslek);
prst.setInt(5,maas);
prst.executeUpdate();
response.sendRedirect("list.jsp");
}
catch(SQLException e)
{
out.println("Kayıt Eklenemedi...");
}
%>
```

```
// delete.jsp

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<%@include file="database.jsp"%>

<%

Object oturum = session.getAttribute("kimlik");

if (oturum == null) response.sendRedirect("index.jsp");

try

{

int gelenID=Integer.parseInt(request.getParameter("ID"));

query = "DELETE FROM personel WHERE sicilNo=" + gelenID;

st = baglan.createStatement();

st.executeUpdate(query);

response.sendRedirect("list.jsp");

}

catch (Exception e)

{

out.println("Kayıt Silinemedi...");

}

%>
```

```
// updateform.jsp
```

```
<% @page contentType="text/html" pageEncoding="UTF-8"%>
```

```
<% @include file="database.jsp"%>
```

```
<%
```

```
Object oturum = session.getAttribute("kimlik");
```

```
if (oturum == null) response.sendRedirect("list.jsp");
```

```
try
```

```
{
```

```
String gelenID = request.getParameter("ID");
```

```
query = "SELECT sicilNo,ad,soyad,birimKod,birimAd,gorevKod,meslekAd,maas " +
```

```
"FROM personel INNER JOIN birimler ON (personel.birimKod=birimler.birimId) " +
```

```
"INNER JOIN meslekler ON (personel.gorevKod=meslekler.meslekId) " +
```

```
"WHERE sicilNo =" +gelenID;
```

```
st = baglan.createStatement();
```

```
rs = st.executeQuery(query);
```

```
if (rs != null) rs.next();
```

```
int sicilNo=rs.getInt("sicilNo");
```

```
String ad=rs.getString("ad");
```

```
String soyad=rs.getString("soyad");
```

```
int birimKod=rs.getInt("birimKod");
```

```
String birimAd=rs.getString("birimAd");
```

```
int gorevKod=rs.getInt("gorevKod");
```

```
String meslekAd=rs.getString("meslekAd");
```

```
int maas=rs.getInt("maas");
```

```
%>
```

```
// updateform.jsp
```

```
<form method="post" action="update.jsp?ID=<%=sicilNo%>">
```

```
<table border="1" cellspacing="0" align="center" width="350" height="250">
```

```
<caption><b>Kayıt Güncelleme Ekranı...</b></caption>
```

```
<tr>
```

```
<th align="left">Personel Sicil No</th>
```

```
<th align="left"><%=sicilNo%></th>
```

```
</tr>
```

```
<tr>
```

```
<th align="left">Personel Adı</th>
```

```
<th align="left"><input name="ad" value="<%=ad%>"></th>
```

```
</tr>
```

```
<tr>
```

```
<th align="left">Personel Soyadı</th>
```

```
<th align="left"><input name="soyad" value="<%=soyad%>"></th>
```

```
</tr>
```

```
<tr>
```

```
<th align="left">Personel Birimi</th>
```

```
<th align="left"><select name="birimAd" style="width:150px;">
```

```
<option value="<%=birimKod%>"><%=birimAd%></option>
```

```
<%=
```

```
query = "SELECT * FROM birimler";
```

```
rs = st.executeQuery(query);
```

```
while((rs!=null) && (rs.next()))
```

```
{
```

```
%>
```

```
<option value="<%=rs.getString("birimId")%>"><%=rs.getString("birimAd")%></option>
```

```
<% } %>
```

```
</select>
```

```
</th>
```

```
</tr>
```

Kayıt Güncelleme Ekranı...	
<input type="button" value="Update..."/>	
Personel Sicil No	6
Personel Adı	<input type="text" value="Mustafa"/>
Personel Soyadı	<input type="text" value="Karahana"/>
Personel Birimi	<input type="text" value="Üretim"/>
Personel Mesleği	<input type="text" value="Memur"/>
Personel Maaşı	<input type="text" value="7000"/>



```
// updateform.jsp
```

```
<tr>
  <th align="left">Personel Mesleği</th>
  <th align="left">
    <select name="meslekAd" style="width:150px;">
      <option value="<%=gorevKod%>"><%=meslekAd%></option>
    <%
      query = "SELECT * FROM meslekler";
      rs = st.executeQuery(query);
      while ((rs!=null) && (rs.next()))
      {
    %>
      <option value="<%=rs.getString("meslekId")%>"><%=rs.getString("meslekAd")%></option>
      <% } %>
    </select>
  </th>
</tr>

<tr>
  <th align="left">Personel Maaşı</th>
  <th align="left"><input type="text" name="maas" value="<%=maas%>"></th>
</tr>

<tr>
  <th colspan="2">
    <button type="submit"> <b>Update...</b> </button>
  </th>
</tr>

</table>
</form>

<%
}
catch (Exception e)
{
  out.println("Kayıt Görüntülenemedi...");
}
%>
```

```

// update.jsp
<% @page contentType="text/html" pageEncoding="UTF-8"% >
<% @include file="database.jsp"% >

<%
Object oturum=session.getAttribute("kimlik");
if (oturum==null) response.sendRedirect("index.jsp");

request.setCharacterEncoding("utf-8");

int gelenID=Integer.parseInt(request.getParameter("ID"));
String isim=request.getParameter("ad");
String soyisim=request.getParameter("soyad");
int birim=Integer.parseInt(request.getParameter("birimAd"));
int meslek=Integer.parseInt(request.getParameter("meslekAd"));
int maas=Integer.parseInt(request.getParameter("maas"));

try
{
    query="UPDATE personel SET ad=?,soyad=?,birimKod=?,gorevKod=?,maas=? WHERE sicilNo=?";
    prst=baglan.prepareStatement(query);
    prst.setString(1,isim);
    prst.setString(2,soyisim);
    prst.setInt(3,birim);
    prst.setInt(4,meslek);
    prst.setInt(5,maas);
    prst.setInt(6,gelenID);
    prst.executeUpdate();
    response.sendRedirect("list.jsp");
}

catch(SQLException e)
{
    out.println("Kayıt Güncellenemedi...");
}
%>

```

## Örnek : Katmanlı Veri Tabanı Mimarisi:

// Model / adminModel.java

```
package Model;
```

```
public class adminModel
```

```
{
```

```
    private String kullanıcıAdi;
```

```
    private String sifre;
```

```
    public adminModel(String kullanıcıAdi)
```

```
    {
```

```
        this.kullanıcıAdi = kullanıcıAdi;
```

```
    }
```

```
    public adminModel(String kullanıcıAdi,String sifre)
```

```
    {
```

```
        this.kullanıcıAdi = kullanıcıAdi;
```

```
        this.sifre = sifre;
```

```
    }
```

```
    public String getKullanıcıAdi()
```

```
    {
```

```
        return kullanıcıAdi;
```

```
    }
```

```
    public void setKullanıcıAdi(String kullanıcıAdi)
```

```
    {
```

```
        this.kullanıcıAdi = kullanıcıAdi;
```

```
    }
```

```
    public String getSifre()
```

```
    {
```

```
        return sifre;
```

```
    }
```

```
    public void setSifre(String sifre)
```

```
    {
```

```
        this.sifre = sifre;
```

```
    }
```

```
}
```

Database: **magazam**

Table: **yoneticiler**

username : varchar(10)

PrimaryKey

password : varchar(10)

Table: **uyeler**

kullanıcıAd : varchar(10)

Primary Key

ad / soyad / il / ilçe : varchar(50)

adres : varchar(255)

// **Model / uyeModel.java**

package Model;

public class uyeModel

{

private String kullaniciAdi;

private String ad;

private String soyad;

private String il;

private String ilce;

private String adres;

public uyeModel(String kullaniciAdi)

{

    this.kullaniciAdi = kullaniciAdi;

}

public uyeModel(String kullaniciAdi,String ad,  
                    String soyad, String il,String ilce,String adres)

{

    this.kullaniciAdi = kullaniciAdi;

    this.ad = ad;

    this.soyad = soyad;

    this.il = il;

    this.ilce = ilce;

    this.adres = adres;

}

public String getKullaniciAdi()

{

    return kullaniciAdi;

}

public void setKullaniciAdi(String kullaniciAdi)

{

    this.kullaniciAdi = kullaniciAdi;

}

public String getAd()

{

    return ad;

}

public void setAd(String ad)

{

    this.ad = ad;

}

public String getSoyad()

{

    return soyad;

}

public void setSoyad(String soyad)

{

    this.soyad = soyad;

}

public String getIl()

{

    return il;

}

public void setIl(String il)

{

    this.il = il;

}

public String getIlce()

{

    return ilce;

}

public void setIlce(String ilce)

{

    this.ilce = ilce;

}

public String getAdres()

{

    return adres;

}

public void setAdres(String adres)

{

    this.adres = adres;

}

}

```
// Control / dbControl.java
```

```
package Control;
```

```
import java . sql . *;
```

```
public class dbControl
```

```
{
```

```
    public PreparedStatement ps=null;
```

```
    public ResultSet rs=null;
```

```
    public String sql=null;
```

```
    public Connection baglan()
```

```
    {
```

```
        Connection magazam=null;
```

```
        try
```

```
        {
```

```
            Class.forName("com.mysql.jdbc.Driver");
```

```
            magazam = DriverManager.getConnection("jdbc:mysql://localhost:3306/magazam","root","");
```

```
        }
```

```
        catch (Exception e)
```

```
        {
```

```
            e.printStackTrace();
```

```
        }
```

```
        return magazam;
```

```
    }
```

```
}
```

**// Control / adminControl.java**

```
package Control;
```

```
import java . util . *;
```

```
import java . sql . *;
```

```
import Model . adminModel;
```

```
public class adminControl extends dbControl
```

```
{
```

```
    dbConnect db=new dbConnect();
```

```
    Connection conn=db.baglan();
```

```
    public boolean yoneticiciKontrol(adminModel yoneticici)
```

```
    {
```

```
        boolean sonuc=false;
```

```
        try
```

```
        {
```

```
            sql="SELECT * FROM yoneticiler";
```

```
            ps=conn.prepareStatement(sql);
```

```
            rs=ps.executeQuery();
```

```
            while(rs.next())
```

```
            {
```

```
                if(yoneticici.getKullaniciAdi().equals(rs.getString("username"))
```

```
                    && yoneticici.getSifre().equals(rs.getString("password")))sonuc=true;
```

```
            }
```

```
            return sonuc;
```

```
        }
```

```
        catch (Exception e)
```

```
        {
```

```
            return sonuc;
```

```
        }
```

```
    }
```

```
}
```

**// Control / uyeControl.java**

```
package Control;

import java . util . *;
import java . sql . *;
import Model . uyeModel;

public class uyeControl extends dbControl
{
    dbConnect db=new dbConnect();
    Connection conn=db.baglan();

    public boolean uyeKontrol(uyeModel uye)
    {
        boolean sonuc=false;
        try
        {
            sql="SELECT * FROM uyeler";
            ps=conn.prepareStatement(sql);
            rs=ps.executeQuery();

            while(rs.next())
            {
                if(uye.getKullaniciAdi().equals(rs.getString("kullaniciAdi"))) sonuc=true;
            }
            return sonuc;
        }
        catch (Exception e)
        {
            return sonuc;
        }
    }

    public List<uyeModel> uyeGetir()
    {
        List<uyeModel> liste=new ArrayList<>();
        try
        {
            sql="SELECT * FROM uyeler";
            ps=conn.prepareStatement(sql);
            rs=ps.executeQuery();
            while(rs.next())
            {
                uyeModel uye=new uyeModel(rs.getString("kullaniciAdi") , rs.getString("ad"),
                    rs.getString("soyad") , rs.getString("il") , rs.getString("ilce") , rs.getString("adres"));

                liste.add(uye);
            }
            return liste;
        }
        catch (Exception e)
        {
            return null;
        }
    }
}
```

// Control / uyeControl.java

```
public void uyeEkle(uyeModel uye)
{
    try
    {
        sql="INSERT INTO uyeler(kullaniciAdi , ad , soyad , il , ilce , adres) VALUES(?, ?, ?, ?, ?, ?)";
        ps=conn.prepareStatement(sql);
        ps.setString(1 , uye.getKullaniciAdi());
        ps.setString(2 , uye.getAd());
        ps.setString(3 , uye.getSoyad());
        ps.setString(4 , uye.getIl());
        ps.setString(5 , uye.getIlce());
        ps.setString(6 , uye.getAdres());
        ps.executeUpdate();
    }
    catch (Exception e)
    {
        e.printStackTrace();
    }
}

public void uyeGuncelle(uyeModel uye)
{
    try
    {
        sql="UPDATE uyeler SET ad=?, soyad=?, il=?, ilce=?, adres=? WHERE kullaniciAdi=?";
        ps=conn.prepareStatement(sql);
        ps.setString(1 , uye.getAd());
        ps.setString(2 , uye.getSoyad());
        ps.setString(3 , uye.getIl());
        ps.setString(4 , uye.getIlce());
        ps.setString(5 , uye.getAdres());
        ps.setString(6 , uye.getKullaniciAdi());
        ps.executeUpdate();
    }
    catch (Exception e)
    {
        e.printStackTrace();
    }
}

public void uyeSil(uyeModel uye)
{
    try
    {
        sql="DELETE FROM uyeler WHERE kullaniciAdi=?";
        ps=conn.prepareStatement(sql);
        ps.setString(1 , uye.getKullaniciAdi());
        ps.executeUpdate();
    }
    catch (Exception e)
    {
        e.printStackTrace();
    }
}
```



#### // adminLogin.jsp

```
<% @page contentType="text/html" pageEncoding="UTF-8" %>

<form action="adminKontrol.jsp" method="post">
<table border="1" align="center" width="300">
<tr>
<th colspan="2">
Yönetici Giriş Ekranı...
</th>
</tr>
<tr>
<th>Kullanıcı Adı</th>
<th><input name="kullaniciAdi"></th>
</tr>
<tr>
<th>Parola</th>
<th><input name="sifre"></th>
</tr>
<tr>
<th align="center" colspan="2">
<button><b>Login...</b></button>
</th>
</tr>
</table>
</form>
```

Yönetici Giriş Ekranı...	
Kullanıcı Adı	<input type="text"/>
Parola	<input type="password"/>
<input type="button" value="Login..."/>	

#### // adminKontrol.jsp

```
<% @page contentType="text/html" pageEncoding="UTF-8" %>
<% @page import="Model . adminModel , Control . adminControl" %>

<%
String kullaniciAdi=request.getParameter("kullaniciAdi");
String sifre=request.getParameter("sifre");

adminControl db=new adminControl();
adminModel yonetici=new adminModel(kullaniciAdi,sifre);
boolean kontrol=db.yoneticiKontrol(yonetici);

if (kontrol)
response.sendRedirect("adminPanel.jsp");
else
response.sendRedirect("adminLogin.jsp");
%>
```

// adminPanel.jsp

```
<% @ page contentType="text/html; charset=UTF-8" pageEncoding="UTF-8" %>
<% @page import ="Control . uyeControl , Model . uyeModel , java . util . *" %>
```

```
<form action="uyeEkle.jsp" method=""post">
```

```
<table border="1" align="center" cellpadding="0" width="350">
```

```
<Caption><h2>Üye Takip Ekranı...</h2></caption>
```

```
<tr>
```

```
<td>Üye ID</td>
```

```
<td><input name="kullaniciAdi"></td>
```

```
</tr>
```

```
<tr>
```

```
<td>Üye Adı</td>
```

```
<td><input name="ad"></td>
```

```
</tr>
```

```
<tr>
```

```
<td>Üye Soyadı</td>
```

```
<td><input name="soyad"></td>
```

```
</tr>
```

```
<tr>
```

```
<td>İl Adı</td>
```

```
<td>
```

```
<select name="il" id="il" required style="width:175px">
```

```
<option value="">İl</option>
```

```
</select>
```

```
</td>
```

```
</tr>
```

```
<tr>
```

```
<td>İlçe Adı</td>
```

```
<td>
```

```
<select name="ilce" id="ilce" style="width:175px" disabled required>
```

```
<option value="">İlçe</option>
```

```
</select>
```

```
</td>
```

```
</tr>
```

```
<tr>
```

```
<td>İletişim Adresi</td>
```

```
<td><textarea name="adres" rows="5" cols="20"></textarea> </td>
```

```
</tr>
```

```
<tr>
```

```
<th colspan="2">
```

```
<button><b>Insert...</b></button>
```

```
</th>
```

```
</tr>
```

```
</table>
```

```
</form>
```

Üye Takip Ekranı...			
Üye ID	<input type="text"/>		
Üye Adı	<input type="text"/>		
Üye Soyadı	<input type="text"/>		
İl Adı	<input type="text"/>		
İlçe Adı	<input type="text"/>		
İletişim Adresi	<input type="text"/>		
<input type="button" value="Insert..."/>			

Üye ID	Üye Adı	Üye Soyadı	İşlemler...
aaa	aaa	aaa	<input type="button" value="Detail..."/> <input type="button" value="Update..."/> <input type="button" value="Delete..."/>
akara	Aykut	Kara	<input type="button" value="Detail..."/> <input type="button" value="Update..."/> <input type="button" value="Delete..."/>
fyuksel	Fatma	Yüksel	<input type="button" value="Detail..."/> <input type="button" value="Update..."/> <input type="button" value="Delete..."/>

```
// adminPanel.jsp
```

```
<hr>
```

```
<table border="1" align="center" cellspacing="0" width="500" height="200" >
```

```
<tr>
```

```
<th>Üye ID</th>
```

```
<th>Üye Adı</th>
```

```
<th>Üye Soyadı</th>
```

```
<th>İşlemler</th>
```

```
</tr>
```

```
<%
```

```
uyeControl db=new uyeControl();
```

```
List<uyeModel> uyeler=db.uyeGetir();
```

```
for(uyeModel uye:uyeler)
```

```
{
```

```
%>
```

```
<tr>
```

```
<th><%=uye.getKullaniciAdi() %></th>
```

```
<td><%=uye.getAd() %></td>
```

```
<td><%=uye.getSoyad() %></td>
```

```
<a href="uyeGuncelle.jsp?kullaniciAdi=<%=uye.getKullaniciAdi()%>">
```

```
<button> <b>Update...</b> </button>
```

```
</a>
```

```
<td align="center">
```

```
<a href="uyeDetay.jsp?kullaniciAdi=<%=uye.getKullaniciAdi()%>">
```

```
<button> <b>Detail...</b> </button>
```

```
</a>
```

```
<a href="uyeSil.jsp?kullaniciAdi=<%=uye.getKullaniciAdi()%>">
```

```
<button> <b>Delete...</b> </button>
```

```
</a>
```

```
</td>
```

```
</tr>
```

```
<% } %>
```

```
</table>
```

```
<% @include file="adresler.jsp"%>
```

#### // adresler.jsp

```
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script>
$.getJSON("iller.json", function(sonuc){
    $.each(sonuc, function(index, value){
        var row="";
        row += '<option value="'+value.il+'">'+value.il+'</option>';
        $("#il").append(row);
    })
});
$("#il").on("change", function(){
    var il=$(this).val();
    $("#ilce").attr("disabled", false).html("<option value=''>Seçin..</option>");
    $.getJSON("ilceler.json", function(sonuc){
        $.each(sonuc, function(index, value){
            var row="";
            if(value.il==il)
            {
                row += '<option value="'+value.ilce+'">'+value.ilce+'</option>';
                $("#ilce").append(row);
            }
        });
    });
});
</script>
```

#### // uyeEkle.jsp

```
<% @page import = "Control . uyeControl , Model . uyeModel , java . util . *" %>
<%
request.setCharacterEncoding("utf-8");
String kullaniciAdi=request.getParameter("kullaniciAdi");
String ad=request.getParameter("ad");
String soyad=request.getParameter("soyad");
String il=request.getParameter("il");
String ilce=request.getParameter("ilce");
String adres=request.getParameter("adres");

uyeControl db=new uyeControl();
uyeModel uye=new uyeModel(kullaniciAdi , ad , soyad , il , ilce , adres);
boolean kontrol=db.uyeKontrol(uye);

if (kontrol) db.uyeGuncelle(uye);
else db.uyeEkle(uye);
response.sendRedirect("uyeDetay.jsp?kullaniciAdi="+kullaniciAdi);
%>
```

## // uyeDetay.jsp

```
<% @ page contentType="text/html; charset=UTF-8" pageEncoding="UTF-8" %>
<% @page import ="Control . uyeControl , Model . uyeModel , java . util . *" %>

<%
String kullanıcıAdi=request.getParameter("kullaniciAdi");
if (kullanıcıAdi!=null)
{
    uyeControl db=new uyeControl();
    List<uyeModel> uyeler= db.uyeGetir();

    for (uyeModel uye:uyeler)
        if (uye.getKullaniciAdi().equals(kullanıcıAdi))
        {
%>
```

### Üye Detay Ekranı...

Üye ID	akara
Üye Adı	Aykut
Üye Soyadı	Kara
İl Adı	ANKARA
İlçe Adı	ÇANKAYA
İletişim Adresi	Güneş Sok. No:6
<input type="button" value="View..."/> <input type="button" value="Update..."/> <input type="button" value="Delete..."/>	

```
<table border="1" align="center" cellspacing="0" width="350" height="250">
<Caption><h2>Üye Detay Ekranı...</h2></caption>
<tr>
<td>Üye ID</td>
<td><%=uye.getKullaniciAdi()%></td>
</tr>
<tr>
<td>Üye Adı</td>
<td><%=uye.getAd()%></td>
</tr>
<tr>
<td>Üye Soyadı</td>
<td><%=uye.getSoyad()%></td>
</tr>
<tr>
<td>İl Adı</td>
<td><%=uye.getIl()%></td>
</tr>
<tr>
<td>İlçe Adı</td>
<td><%=uye.getIlce()%></td>
</tr>
<tr>
<td>İletişim Adresi</td>
<td><%=uye.getAdres()%></td>
</tr>
<tr>
<th colspan="2">
<a href="adminPanel.jsp"><button><b>View...</b></button></a>

<a href="uyeGuncelle.jsp?kullaniciAdi=<%=uye.getKullaniciAdi()%>">
<button><b>Update...</b></button>
</a>

<a href="uyeSil.jsp?kullaniciAdi=<%=uye.getKullaniciAdi()%>">
<button><b>Delete...</b></button>
</a>
</th>
</tr>
</table>

<% } } %>
```

```
// uyeGuncelle.jsp
```

```
<% @ page contentType="text/html; charset=UTF-8" pageEncoding="UTF-8" %>
```

```
<% @page import ="Control . uyeControl , Model . uyeModel , java . util . *" %>
```

```
<%
```

```
String kullanıcıAdi=request.getParameter("kullanıcıAdi");
```

```
if (kullanıcıAdi!=null)
```

```
{
```

```
    uyeControl db=new uyeControl();
```

```
    List<uyeModel> uyeler= db.uyeGetir();
```

```
    for (uyeModel uye:uyeler)
```

```
        if (uye.getKullanıcıAdi().equals(kullanıcıAdi))
```

```
        {
```

```
%>
```

```
<form action="uyeEkle.jsp" method="post">
```

```
<table border="1" align="center" cellspacing="0" width="350" height="250">
```

```
<Caption><h2>Üye Güncelle Ekranı...</h2></caption>
```

```
<tr>
```

```
<td>Üye ID</td>
```

```
<td><input name="kullanıcıAdi" value="<%=uye.getKullanıcıAdi()%>" readonly></td>
```

```
</tr>
```

```
<tr>
```

```
<td>Üye Adı</td>
```

```
<td><input name="ad" value="<%=uye.getAd()%>"></td>
```

```
</tr>
```

```
<tr>
```

```
<td>Üye Soyadı</td>
```

```
<td><input name="soyad" value="<%=uye.getSoyad()%>"></td>
```

```
</tr>
```

```
<tr>
```

```
<td>İl Adı</td>
```

```
<td>
```

```
<select name="il" id="il" required style="width:175px">
```

```
<option value="<%=uye.getIl()%>"><%=uye.getIl()%></option>
```

```
</select>
```

```
</td>
```

```
</tr>
```

Üye ID	akara
Üye Adı	Aykut
Üye Soyadı	Kara
İl Adı	ANKARA
İlçe Adı	ÇANKAYA
İletişim Adresi	Güneş Sok. No:6
<input type="button" value="Update..."/>	

```
// uyeGuncelle.jsp
```

```
<tr>
  <td>İlçe Adı</td>
  <td>
    <select name="ilce" id="ilce" style="width:175px" readonly required>
      <option value="<%=uye.getIlce()%>"><%=uye.getIlce()%></option>
    </select>
  </td>
</tr>

<tr>
  <td>İletişim Adresi</td>
  <td>
    <textarea name="adres" rows="5" cols="20">
      <%=uye.getAdres()%>
    </textarea>
  </td>
</tr>

<tr>
  <th colspan="2">
    <button><b>Update...</b></button>
  </th>
</tr>
</table>
</form>

<% @include file="adresler.jsp"%>
```

```
// uyeSil.jsp
```

```
<% @page import = "Control . uyeControl , Model . uyeModel" %>

<%
String kullanıcıAdi=request.getParameter("kullaniciAdi");

uyeControl db=new uyeControl();
uyeModel uye=new uyeModel(kullaniciAdi);
db.uyeSil(uye);

response.sendRedirect("adminPanel.jsp");
%>
```

## 6. COOKIES (ÇEREZ) KULLANMAK:

İstemci ile sunucu arasındaki etkileşim her zaman için istemciden sunucuya doğru olduğu bilinmektedir. Yani sunucu istemciden gelen bilgileri rahatlıkla saklayabilirken, istemci sunucudan gelen bilgileri saklayamamaktaydı. Çerezler istemci ile sunucu arasındaki bu açığı gidermek için geliştirilmiştir.

Bazı kaynaklarda, HTTP Cookie, Web Cookie, Browser Cookie, Session Cookie olarak da karşımıza çıkan COOKIE'ler (çerezler), internet sitelerinin bilgisayarımıza kaydettiği küçük metin dosyalarıdır.

Genel kullanım amacı, siteye gelen ziyaretçileri, bir sonraki ziyaretlerinde hatırlamaktır. Pratikte çerezler, web server tarafından tarayıcıya gönderilen ve istemci bilgisayarında kaydedilen ufak metin dosyalarıdır. Çerezler, kullanıcı kimlik doğrulama ve tanımlama, kullanıcı tercihleri, alışveriş sepeti, vb. bilgileri istemci tarafında tutmak için kullanılır. Ayrıca, bilgilerin bir sayfadan başka bir sayfaya taşınması için de kullanılabilir.

### Çerezlerin Özellikleri:

**1. Veri Devamlılığı:** İstemcilerde veri bütünlüğünü en yüksek oranda garanti edebilen yapılardır. İstemci taraflı olduğundan sunucu kaynaklarını yormaz. Rahatlıkla erişip okunabilen çerezler sadece text tabanlı veriler tutabilir.

**2. Boyut Limiti:** Birçok web tarayıcısı çerez boyutu olarak 4 KB sınırı getirmiştir. Bir istemci bütün web sitelerinden aynı anda en fazla 300 çerezi barındırabilir. Bir web sunucusu da istemcide 20'den fazla çerez dosyası oluşturamaz.

**3. Güvenlik Riski:** Çerezler aynı bilgisayarı kullanan diğer kullanıcılar tarafından rahatlıkla görülebilir. Bu nedenle güvenliği düşünüp kullanıcı bilgileri şifreleme algoritmaları yardımıyla korunması gerekir.

**4. Çerez Ömrü:** Çerezler, kullanıcının silmesiyle, tarayıcı geçmişini temizlemesiyle, bilgisayarını formatlamasıyla ya da çerezin zaman aşımına uğramasıyla ömrünü tamamlar.

**5. Tarayıcı Faktörü:** İstemcideki her web tarayıcı için farklı çerez klasörü kullanılır. Yani IE'deki çerezi, MF veya GC okuyamaz. Ayrıca bazı istemciler çerez alımını tarayıcıdan kapatabilir, bu da çerez kullanımını olanaksız hale getirir. Ayrıca çerezler yüksek güvenlik seviyesinde çalışmazlar.



## Örnek:Cookie1

// index.jsp

```
<% @ page contentType="text/html; charset=UTF-8"%>
<form action="cerezOlustur.jsp" method="POST">
  <p>User.....: <input name="kullanici">
  <p>Password..: <input name="parola" />
  <p><input type="submit" value="Submit" />
</form>
```

User.....:	<input type="text" value="Türkiyem"/>
Password..:	<input type="password" value="Cennetim"/>
<input type="submit" value="Submit"/>	

// cerezOlustur.jsp

```
<% @ page contentType="text/html; charset=UTF-8"%>

<%
String user=request.getParameter("kullanici");
String password=request.getParameter("parola");

// Sabit Çerez
Cookie kullanıcı1 = new Cookie("user", user);
Cookie parola1 = new Cookie("password", password);
kullanıcı1.setMaxAge(3600*1*24);      // 1 gün
parola1.setMaxAge(3600*1*24);        // 1 gün
response.addCookie(kullanıcı1);
response.addCookie(parola1);

//Dinamik Çerez

Cookie kullanıcı2 = new Cookie(user, user);
Cookie parola2 = new Cookie(password, password);
kullanıcı2.setMaxAge(3600*1*24);      // 1 gün
parola2.setMaxAge(3600*1*24);        // 1 gün
response.addCookie(kullanıcı2);
response.addCookie(parola2);
%>

Oluşturulan Çerez Bilgileri<hr>
<p>Kullanıcı.....:<b><%=user%></b>
<p>Parola.....:<b><%=password%></b>

<%out.println("<p>Çerez Oluşturuldu...");%>
<p><a href="cerezListele.jsp">Çerezleri Listele...</a>
```

### Oluşturulan Çerez Bilgileri

Kullanıcı.....: **Türkiyem**

Parola.....: **Cennetim**

Çerez Oluşturuldu...

[Çerezleri Listele...](#)

//cerezListele.jsp

```
<% @ page contentType="text/html; charset=UTF-8"%>
<h2>Localhost Sunucusunun Çerez Listesi </h2>
<%
    Cookie[] cerezler= request.getCookies();
    if(cerezler !=null)
    {
        for(Cookie cerez : cerezler)
            out.println("<p>" + cerez.getName()+":"+cerez.getValue() );
    }
%>
<p><a href="cererSil.jsp">Çerezleri Sil...</a>
<a href="index.jsp">Çerez Tanımla...</a>
```

### Localhost Sunucusunun Çerez Listesi

Turkiyem:Turkiyem  
Cennetim:Cennetim  
user:Hatay  
password:Antakya  
Hatay:Hatay  
Antakya:Antakya  
JSESSIONID:6D2A39E085BB2C4E1253573700D72902  
[Çerezleri Sil...](#) [Çerez Tanımla...](#)

//cererSil.jsp

```
<% @ page contentType="text/html; charset=UTF-8"%>
<%
    Cookie [] cerezler = request.getCookies();
    if( cerezler != null )
    {
        out.println("<h2>Bilgisayarınızdan Silinen Çerezler...</h2><hr>");
        for (Cookie cerez:cerezler)
        {
            // if(cerez.getName().equals("user") || cerez.getName().equals("password"))
            {
                cerez.setMaxAge(0);
                response.addCookie(cerez);
                out.print("<p>Çerez Adı: " +cerez.getName( ) + "<br/>");
            }
        }
    }
%>
<p><a href="cererListele.jsp">Çerezleri Listele...</a>
```

### Silinen Çerezler...

Çerez Adı: Turkiyem  
Çerez Adı: Cennetim  
Çerez Adı: user  
Çerez Adı: password  
Çerez Adı: Hatay  
Çerez Adı: Antakya  
Çerez Adı: JSESSIONID  
[Çerezleri Listele...](#)

## Örnek:Cookie2

// index.jsp

```
<% @ page language="java" contentType="text/html; charset=UTF-8"%>
<form method="get" action="ziyaret.jsp">
  <p><b>User Your Name: </b>
  <p><input name="username">
  <p><input type="submit" value="Giriş...">
</form>
```

**User Your Name:**

//cerezListele.jsp

```
<% @ page contentType="text/html; charset=UTF-8"%>
```

```
<h2>Localhost Sunucusunun Çerez Listesi </h2>
```

```
<h2>Localhost Sunucusunun Çerez Listesi </h2>
```

```
<%
```

```
    Cookie[] cerezler= request.getCookies();
```

```
    if(cerezler !=null)
```

```
    {
```

```
        for(Cookie cerez : cerezler)
```

```
            out.println("<p>" + cerez.getName()+":"+cerez.getValue() );
```

```
    }
```

```
%>
```

```
<p><a href="index.jsp">Site Giriş...</a>
```

**Localhost Sunucusunun Çerez Listesi**

eee:4

hhh:2

aaa:3

bbb:5

hasan:3

JSESSIONID:386D689C5322335D5BC993CE118E5E89

[Site Giriş...](#)

//ziyaret.jsp

```
<% @ page language="java" contentType="text/html; charset=UTF-8"%>
<%
    request.setCharacterEncoding("utf-8");
    String ziyaretci=request.getParameter("username").trim();
    if (ziyaretci.length()==0) ziyaretci="misafir";
    Cookie cerezler[]=request.getCookies();
    Cookie cerezim=null;
    if (cerezler != null)
    for (int i = 0; i < cerezler.length; i++)
    if (cerezler[i].getName().equals (ziyaretci))
    {
        cerezim = cerezler[i]; break;
    }
    if (cerezim == null)
    {
        out.println("<h2>"+ziyaretci+" isimli çerez bulunamadı...");
        out.println("<p>Sitemizi ilk kez ziyaret ediyorsun...</h2>");
        cerezim=new Cookie(ziyaretci,"1");
        cerezim.setMaxAge(3600*10*24);// 10 gün
        response.addCookie(cerezim);
    }
    else
    {
        int saydir=0;
        for(Cookie cerez:cerezler)
        {
            if(cerez.getName().equals(ziyaretci))
            {
                saydir = Integer.parseInt(cerez.getValue());
                saydir++;
                cerez.setValue(String.valueOf(saydir));
                response.addCookie(cerez); break;
            }
        }
        out.println("<p><h2>Hoşgeldin....:"+cerezim.getName());
        out.println("</b><p>Sitemizi: " + saydir + ". kez ziyaret ediyorsun...</h2>");
    }
    out.print("<a href='cereziListele.jsp'>cerezi Listele...</a>");
%>
```

**Hoşgeldin....:hasan**

**Sitemizi: 3. kez ziyaret ediyorsun...**

[cerezi Listele...](#)

## Örnek:Cookie3

// index.jsp

```
<% @ page contentType="text/html; charset=UTF-8"%>
```

```
<%
request.setCharacterEncoding("utf-8");
Cookie[] cerezler = request.getCookies();
String username="", password="";
if(cerezler!=null)
{
for(Cookie cerez: cerezler)
{
if(cerez.getName().equals("kullanici")) username= cerez.getValue();
else if(cerez.getName().equals("parola")) password= cerez.getValue();
}
}
}%>
```

```
<form action="hatirlaUnut.jsp" method="POST">
<%
if (username.trim().length()!=0)
{
%>
```

```
<p> Kullanıcı: <input name="user" value ="<%=username %>" readonly/>
<p> Parola: <input name="password" value="<%=password%>" readonly/>
<p> <input type="checkbox" name="unut"/>Beni Unut
<p> <input type="hidden" name="cerez" value="<%=username%>"/>
<p> <input type="submit" value="Submit"/> <%
}
else
{
%>
<p> Kullanıcı: <input name="user" id="user" onkeyup="kontrol()"/>
<p> Parola: <input name="password" id="password" onkeyup="kontrol()"/>
<p> <input type="checkbox" name="hatirla"/>Beni Hatırla
<p> <input type="hidden" name="cerez" value="<%=username%>">
<p> <input type="submit" value="Submit" id="gonder" disabled/>
<% }%>
</form>
```

```
<script>
function kontrol()
{
if(document.getElementById("user").value.length == 0)
document.getElementById("gonder").disabled = true;
else if(document.getElementById("password").value.length == 0)
document.getElementById("gonder").disabled = true;
else
document.getElementById("gonder").disabled = false;
}
</script>
```

Username:

Password:

☒ Beni Hatırla

Username:

Password:

☒ Beni Unut

// hatirlaUnut.jsp

```
<% @ page contentType="text/html; charset=UTF-8"%>
<%
    request.setCharacterEncoding("utf-8");
    String username=request.getParameter("user");
    String password=request.getParameter("password");
    String hatirla= request.getParameter("hatirla");
    String unut= request.getParameter("unut");

    String message="User : "+ username + "<p> Password :"+ password ;

    if (hatirla==null && request.getParameter("cerez").length()==0)
        out.println("<h2>Sen Kimsin...</h2>");
    else if(hatirla!=null && request.getParameter("cerez").length()==0)
    {
        Cookie cerez1 = new Cookie("kullanici", username);
        Cookie cerez2 = new Cookie("parola", password);
        cerez1.setMaxAge(3600*7*24);           // 1 Hafta
        cerez2.setMaxAge(3600*7*24);           // 1 Hafta
        response.addCookie(cerez1);
        response.addCookie(cerez2);
        out.println(message);
        out.println("<p><h2>Seni Hatırlıyorum...</h2>");
    }
    else if (unut==null && request.getParameter("cerez").length()!=0)
    {
        out.println(message);
        out.println("<p><h2>Seni Hatırlıyorum...</h2>");
    }
    else if(unut!=null)
    {
        Cookie[] cerezler = request.getCookies();
        for (Cookie cerez:cerezler)
        {
            if(cerez.getName().equals("kullanici") || cerez.getName().equals("parola"))
            {
                cerez.setMaxAge(0);
                response.addCookie(cerez);
            }
        }
        out.println("<h2>Seni Unuttum...</h2>");
    }
%>
```

User : admin  
Password :admin

**Seni Hatırlıyorum...**

## 7. SESSION (OTURUM) NESNESİ:

Session kelime anlamı oturum demektir. Oturum ise web tarayıcısının açıldığı andan kapatıldığı ana kadar geçen süredir. Sunucu, her ziyaretçi için otomatikman oturum başlatmaktadır. Oturum başlatılan her ziyaretçi için sunucu tarafından oturum boyunca geçerli olacak SessionID üretilir. Session nesnesi JSP ile yazılım geliştirirken en çok kullanılan nesnelerden birisidir.

Bir web sayfasından diğerine değişken aktarmak neredeyse imkânsızdır. Çünkü değişkenlerin ömrü sadece tanımlandıkları sayfayla sınırlı olduğu unutulmamalıdır. Bu sorunu çözenin en kolay yolu değişkenleri session olarak saklamaktır. Session nesnesinin en önemli işlevi, kullanıcı bilgilerine oturum aktif olduğu süre boyunca uygulama içerisinde her yerden erişim sağlamaktır.

Session değişkenlerini web tarayıcısıyla ilişkilendirmekle ziyaretçiyle ilişkilendirmek aynı şey değildir. Çünkü sunucu açısından bakıldığında, IP adresleri aynı olsa bile, her web tarayıcısı ayrı bir oturum olarak kabul edilmektedir. En çok kullanılan metodları şunlardır;

**setMaxInactiveInterval()** : Her ziyaretçi için sunucu üzerinde açılan ideal pasif kalma süresi 1800 saniye (30 dk) 'dır. Ancak ziyaretçi sitede 1 dk kalsa bile geri kalan 29 dk boyunca bu oturumun sunucuda yer işgal etmeye devam edeceği unutulmamalıdır.

**invalidate()** : Oturumu sonlandırmaya yarar. Bu metod çağırıldıktan sonra tekrar session nesnesine erişilmeye çalışılırsa hata alınır.

**setAttribute(String, object)** : Bu metod session nesnesinde veri saklamamızı sağlar. Veri Object tipinde olduğundan her tipte veri saklayabilir.

**getAttribute(String name)** : Parametre olarak aldığı değere bağlı bir veri saklanıyorsa onu getirir. Eğer saklanan bir veri yoksa null döndürür.

**removeAttribute(String name)** : Session nesnesinde saklanan bir veriyi silmeyi sağlar. Parametre olarak veriye atanmış String tipindeki değer verilmelidir.

**isNew()** : Oturumun yeni olup olmadığını kontrol eder. True ya da false döndürür. Aynı zamanda tarayıcının cookie(çerez) özelliğinin açık olup olmadığını kontrol etmek için de kullanılır. Eğer cookie özelliği kapalıysa isNew() metodu her zaman true dönecektir.

**getCreationTime()** : Session nesnesinin ilk oluşturulduğu tarihi verir.

**getLastAccessedTime()** : Oturumun açık olduğu süre boyunca uygulamaya en son erişilen tarihi verir.

### Session oturumunu Kapatmak:

```
<%@page language="java" session="false"%>
```

## 8. APPLICATION (UYGULAMA) NESNESİ:

Application, sunucu taraflı bir veri saklama yöntemidir. Bu nesne bir anlamda sitenize giren her ziyaretçi için yeni bir program açılıyor gibi düşünülebilir. Application nesnesi, özellikle hit saymak için başvurulmuş bir durum yönetimidir.

Session nesnesi ziyaretçi sunucuya bağlı kaldığı sürece bütün sayfalar için geçerli iken, Application nesnesi ise uygulama açık kaldığı sürece bütün ziyaretçiler için geçerli değişkenleri tanımlayan nesnedir.

Dolayısıyla her oturumda değişmeyen ve bütün ziyaretçiler için geçerli evrensel tanımlara veya değerlere ihtiyaç duyulduğunda Application nesnesi kullanılmalıdır. Application nesnesinde tanımlanan bir değişkeni ve değerini tüm ziyaretçiler görür. Bu değişkenlerin ömrü web sunucunun çalışma süresi kadardır.

**void setAttribute(String name, Object object):** Her yerden erişilebilecek bir değişken oluşturmaya yarar. Değişken Object tipinde olduğundan her türde veri saklayabilir.

```
<% application.setAttribute("web","mku"); %>
```

**Object getAttribute(String name):** setAttribute() ile oluşturulan değişkene erişebilir.

```
<% application.getAttribute("web"); %>
```

**void removeAttribute(String name):** setAttribute() ile oluşturulan değişkenin sahip olduğu değeri siler. Değeri silinen bir değişkene ulaşmaya çalışırsanız null dönecektir.

```
<% application.removeAttribute("Site"); %>
```

**Enumeration getAttributeNames():** Eğer birden fazla değişken tanımladıysak ve bu değişkenlerin isimlerine toplu olarak ulaşmak istiyorsak bu metodu kullanırız.

```
<% Enumeration attributeNames = application.getAttributeNames(); %>
```

```
<% Enumeration parameterNames = application.getInitParameterNames(); %>
```

**String getRealPath(String value):** Parametre olarak verilen değer dizin yolunu verir.

```
<%
```

```
String path = application.getRealPath("/ApplicationObject.jsp");
```

```
out.print(path);
```

```
%>
```

**void log(String message):** Parametre olarak verilen değeri, hangi JSP sunucusu kullanılıyorsa onun varsayılan log dosyasına yazar.

```
<% application.log("File not found"); %>
```

**String getServerInfo():** Kullanılan sunucunun ismini ve sürüm numarasını döndürür.

```
<%
```

```
String serverInfo = application.getServerInfo();
```

```
out.print(serverInfo);
```

```
%>
```



### Örnek:

//sessionCounter.jsp

```
<%@ page contentType="text/html; charset=UTF-8"%>
<%
Integer sayac=(Integer)session.getAttribute("oturumSayac");
if( sayac ==null || sayac == 0 )
{
    out.println("İlk Oturumunuz...!");
    sayac = 1;
}
else
{
    out.println("Tekrar Oturum Açıyorsunuz...!");
    sayac += 1;
}
session.setAttribute("oturumSayac", sayac);
%>
<p>Toplam Oturum Sayısı: <%= sayac%></p>
```

İlk Ziyaretiniz...!

Toplam Ziyaret Sayısı: 1

### Örnek:

//appCounter.jsp

```
<%@ page contentType="text/html; charset=UTF-8"%>
<%
Integer sayac = (Integer)application.getAttribute("hitSayac");
if ( sayac ==null || sayac == 0 )
{
    out.println("İlk Ziyaretiniz...!");
    sayac = 1;
}
else
{
    out.println("Tekrar Ziyaret Ediyorsunuz...!");
    sayac += 1;
}

application.setAttribute("hitSayac", sayac);
%>
<p>Toplam Ziyaret Sayısı: <%= sayac%></p>
```

İlk Ziyaretiniz...!

Toplam Ziyaret Sayısı: 1

## Örnek:Session

//index.jsp

```
<% @page contentType="text/html" pageEncoding="UTF-8"%>
<% @ page session="false"%>

<form method="post" action="session.jsp?action=login">
  <p>User.....: <input name="kullanici">
  <p>Password..: <input name="parola">
  <p> <input type="submit" value="login..."><br/>
</form>
```

User.....:	<input type="text" value="admin"/>
Parola.....:	<input type="text" value="admin"/>
<input type="button" value="login..."/>	

//session.jsp

```
<% @page contentType="text/html" pageEncoding="UTF-8"%>
<% @ page import = "java.io.*, java.util.*" %>

<% ! Integer sayac=0; %>

<%
  if (request.getParameter("action") != null)
  {
    String kullanici=request.getParameter("kullanici").trim();
    String parola=request.getParameter("parola").trim();

    if (kullanici.equals("admin") && parola.equals("admin"))
    {
      HttpSession oturum = request.getSession(true); //oturum aç
      oturum.putValue("login", "true");

      session.setAttribute("user", kullanici);
      session.setAttribute("ziyaret", sayac);
      session.setMaxInactiveInterval(30); // 30 saniye
      String user=(String)session.getAttribute("user");

      out.println("<h4>" + user + " isimli Oturum Başlatıldı...<hr>");
      out.println("<a href='detail.jsp'>Oturum Detay...</a>");
    }
    else response.sendRedirect("./error.jsp");
  }
  else response.sendRedirect("./index.jsp");
%>
```

admin isimli Oturum Başlatıldı...
<a href="#">Oturum Bilgileri</a>

//detail.jsp

```
<% @ page contentType="text/html;charset=UTF-8" %>
<% @ page import = "java.io.*, java.util.*" %>

<% @include file="pencere.jsp"%>
<body onload="toplamSure()">

<%
    HttpSession oturum = request.getSession(true);
    if (oturum.getValue("login")=="true")
    {
        Date ilkErisim = new Date(session.getCreationTime());
        Date sonErisim = new Date(session.getLastAccessedTime());

        int sayac = (Integer)session.getAttribute("ziyaret");
        sayac++;
        session.setAttribute("ziyaret", sayac);

        out.println("<h4>Oturum Adı..."+session.getAttribute("user"));
        out.println("<p>Oturum ID..."+session.getId());
        out.println("<p>Oturum Süresi:"+session.getMaxInactiveInterval()/60+" dakika");
        out.println("<p>İlk Erişim Tarihi..."+ilkErisim);
        out.println("<p>Son Erişim Tarihi..."+sonErisim);
        out.println("<p>İstek Sayısı..."+sayac);
        out.println("<hr><a href='index.jsp' >Yeni Oturum...</a>");
        out.println("<a href='error.jsp' >Oturum Kapat...</a>");
    }
    else response.sendRedirect("/error.jsp");
%>
</body>
```

Oturum Adı...:admin

Oturum ID...:B14B78AE7D6A2693196A3875949416DA

Oturum Süresi...:0 dakika

İlk Erişim Tarihi...:Sat Jul 18 15:01:47 EET 2020

Son Erişim Tarihi...:Sat Jul 18 15:01:56 EET 2020

İstek Sayısı...:1

[Yeni Oturum...](#) [Oturum Sonlandırmak...](#)

//logout.jsp

```
<% @ page contentType="text/html;charset=UTF-8" %>
<%
    session.invalidate();
    HttpSession oturum = request.getSession(false);
    out.print("<h2>Oturum Bulunamadı...</h2>");
%>
<a href="index.jsp">
    <h3 style="color: red">Oturum Açmak İstermisin...!</h3>
</a>
```

**Oturum Bulunamadı...**

**Oturum Açmak İstermisin...!**

// timeout.jsp

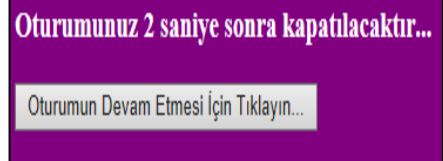
```
<script type="text/javascript">
    zamanGoster();
    function zamanGoster()
    {
        var sayac = document.getElementById("kalan").innerHTML;
        var sure = setTimeout("zamanGoster()", 1000);
        if (sayac > 0)
            document.getElementById("kalan").innerHTML = sayac - 1;
        else
        {
            document.getElementById("btnDevam").style.visibility="hidden";
            document.getElementById("timeOut").style.visibility = "visible";
            clearTimeout(sure);
        }
    }

    function oturumDevam()
    {
        window.close();
        if (!window.opener.closed) window.opener.location.reload();
    }
</script>
```

// pencere.jsp

```
<% @page contentType="text/html" pageEncoding="UTF-8"%>
<script type="text/javascript">

    //20 saniye kala pencere açılacak
    var pencere = <%=session.getMaxInactiveInterval()*1000%> -10000;
    function toplamSure()
    {
        var timer = setTimeout("toplamSure()", 1000);
        if (pencere > 0) pencere -= 1000;
        else
        {
            clearTimeout(timer); window.open("./timeout.jsp","X", "width=400,height=200");
        }
    }
</script>
```



## Örnek: Application

// index.jsp

```
<% @page contentType="text/html" pageEncoding="UTF-8"%>

<%
request.setCharacterEncoding("utf-8");
Cookie[] cerezler = request.getCookies();
String username="",password="";
if(cerezler!=null)
{
for(Cookie cerez:cerezler)
{
if(cerez.getName().equals("username")) username= cerez.getValue();
else if(cerez.getName().equals("password")) password= cerez.getValue();
}
}
%>

<body>
<center>
<h1>Login</h1>
<form action="cookie.jsp" method="POST">
<p> Username: <input name="username" value ="<%= username %>" />
<p> Password:<input name="password" value="<%= password %>" />
<p> <input type="checkbox" name="hatirla" value ="true"/>Beni Hatırla
<input type="checkbox" name="unut" value ="true"/>Beni Unut
<p> <input type="submit" value="Submit"/>
</form>
</center>
</body>
```

A screenshot of a web browser displaying a login page. The page has a title "Login" in a large, bold, black serif font. Below the title, there are two input fields: "Username:" with the text "karahan" entered, and "Password:" with seven dots. Below these fields are two checkboxes: "Beni Hatırla" (checked) and "Beni Unut" (unchecked). At the bottom is a "Submit" button.

//login.jsp

```
<% @page contentType="text/html" pageEncoding="UTF-8"%>
<%
if (!session.isNew())
{
String kullanıcı=(String)session.getAttribute("kullanici");
String parola=(String)session.getAttribute("parola");

if ((kullanıcı.length()==0) || (parola.length()==0)) response.sendRedirect("./index.jsp");
else
{
out.print("<h2>Hoşgeldin...:" +kullanıcı+"</h2><hr>");
out.print("<a href='chat.jsp'>Chat Odası Girişi...</a>");
}
}
else response.sendRedirect("./index.jsp");
%>
```

A screenshot of a web browser displaying a message "Hoşgeldin...:karahan" in a large, bold, black serif font. Below this message is a link "Chat Odası Girişi..." in a purple, underlined serif font.

```
// cookie.jsp
<%@page contentType="text/html" pageEncoding="UTF-8"%>

<%
if (!session.isNew())
{
    request.setCharacterEncoding("utf-8");
    String username=request.getParameter("username").trim();
    String password=request.getParameter("password").trim();
    String hatirla= request.getParameter("hatirla");
    String unut= request.getParameter("unut");

    if (hatirla!=null)
    {
        Cookie cerez1=new Cookie("username", username);
        Cookie cerez2=new Cookie("password", password);
        cerez1.setMaxAge(24*60*60); //1 gün
        cerez2.setMaxAge(24*60*60); //1 gün
        response.addCookie(cerez1);
        response.addCookie(cerez2);
    }
    else if (unut!=null)
    {
        Cookie[] cerezler = request.getCookies();
        for (Cookie cerez:cerezler)
            if (cerez.getName().equals("username") || cerez.getName().equals("password"))
            {
                cerez.setMaxAge(0);
                response.addCookie(cerez);
            }
    }

    session.setAttribute("kullanici",username);
    session.setAttribute("parola",password);
    response.sendRedirect("login.jsp");
}
else response.sendRedirect("index.jsp");
%>
```

## // chat.jsp

```
<% @page contentType="text/html" pageEncoding="UTF-8"%>
<% @ page import="java.util.*" %>

<center>
<%
    request.setCharacterEncoding("utf-8");

    response.setDateHeader("Expires", -1); //Cache temizle
    response.setHeader("Refresh", "10");// 10 saniye refresh

    String userName = (String)session.getAttribute("kullanici");
    if (userName == null) userName = "Misafir";

    String mesaj = request.getParameter("mesaj");
    String chatKayit = (String)application.getAttribute("chatKayit");

    if (mesaj!= null)
    {
        Date d = new Date();
        mesaj = userName + "(" + d.toLocaleString() + "):" + mesaj;
        if (chatKayit == null) chatKayit = mesaj;
        else chatKayit = chatKayit + "<br>" + mesaj;
    }

    if (chatKayit!=null)
    {
        application.setAttribute("chatKayit", chatKayit);
    }
%>

<table border="1" width="400">
<tr>
<th>Sohbet Odası...</th>
</tr>
<tr>
<td><%=application.getAttribute("chatKayit")%></td>
</tr>
</table>
<% } %>

<Form action="chat.jsp" method="post">
<p>
<input name="mesaj" value="">
<input type="submit" value="Gönder...">
</p>
</Form>
</center>
```

Sohbet Odası...
karahan(18.Tem.2020 15:16:05):selam chat dünyası admin(18.Tem.2020 15:16:37):selam karahan
<input type="text"/> <input type="button" value="Gönder..."/>