# Kabuk Programlama (Bash) - 3

Erkan Esmer

Mart, 2013

Bu bölümde şimdiye kadar tüm işlediklerimizi kapsayan bir örnek yapıp yeni konularımıza da bu örneğimizden devam edeceğiz. Amacımız, öğrendiklermizin tümünü örneklendirebildiğimiz bir uygulama veya uygulamalar bütününü ortaya çıkartmaktır.

Şimdi bu bölümümüzde yapacağımız örneğimizi detaylandıralım. Böylece hem örneğimizi belirlemiş olacağız hem de önceki konuların üzerinden geçmiş olacağız.

Örneğimiz, kullanıcının belirlediği isimlerde klasör oluşturan, klasörler içinde dosya oluşturan ve bu dosyanın içine de kullanıcının belirlediği rakamlarla matematiksel işlem yapıp kaydeden bir uygulama olacak.

Şimdi örneğimizin haritasını çıkartalım.

- İlk önce kullanıcıdan oluşturacağımız iki adet klasör için isim girmesini isteyeceğiz.
- Echo ile isteyip read ile değişkeni okuyacağız.
- Ardından kaydettiğimiz değişken isimlerini mkdir komutuna paramatre olarak göndererek 2 klasörümüzü de oluşturacağız.
- Klasörleri oluşturduktan sonra bir klasörümüzün içinde touch komutuna kullanıcının ismini girdiği değişkenimizi parametre olarak göndererek bir dosya oluşturacağız.
- Bundan sonra kullanıcıdan 2 adet sayı isteyeceğiz ve bu sayıları matematik işlemine tabi tutarak sonucu oluşturduğumuz dosyaya yazacağız.

Yukarıdaki maddelere uygun olarak yazdığımız script dosyamızın görüntüsünü aşağıda görmekteyiz.

Şimdi satır satır uygulamamızı inceleyelim. Bu uygulamamızı daha önceki iki yazımızda işlediğimiz konular ile gerçekleştirdik. Dolayısıyla örnek uygulamamız 2 yazımızı da kapsıyor.

Gördüğünüz gibi 1. ve 2. echo komutu ile ekrana mesaj bastırıp read ile de değişkenlerimizi tutuyoruz. Böylece kullanıcının girdiği isimleri almış olduk.

mkdir $klasor1 ve mkdir $klasor2 satırları ile de mkdir komutuna değişkenlerimizi parametre olarak gönderip belirlenen isimlerde klasör oluşturuyoruz.

Yine echo komutuyla ekrana mesajımızı bastırdık.
Ardından read dosya1 satırında oluşturacağımız dosyanın ismini aldık ve touch komutuna $klasor1 ve $dosya1 değişkenlerini parametre yaptık ve kullanıcının belirlediği isimde dosyamızı da oluşturmuş olduk.
Sonrasında sayi1 ve sayi2 değişkenlerini alıyor ve sonuc değişkenine iki sayının toplamını atıyoruz.

Yaptığımız işlemin sonucu $sonuc değişkeninde. Bunu echo $sonuc > $klasor1/$klasor2 satırıyla daha önce açtığımız dosyanın içine yazıyoruz. Bu satırların üstünde yazdığımız typeset -i sonuc ibaresini, hatırlayacağınız üzere sonuc adında tamsayı değişkeni oluşturmak için kullandık.

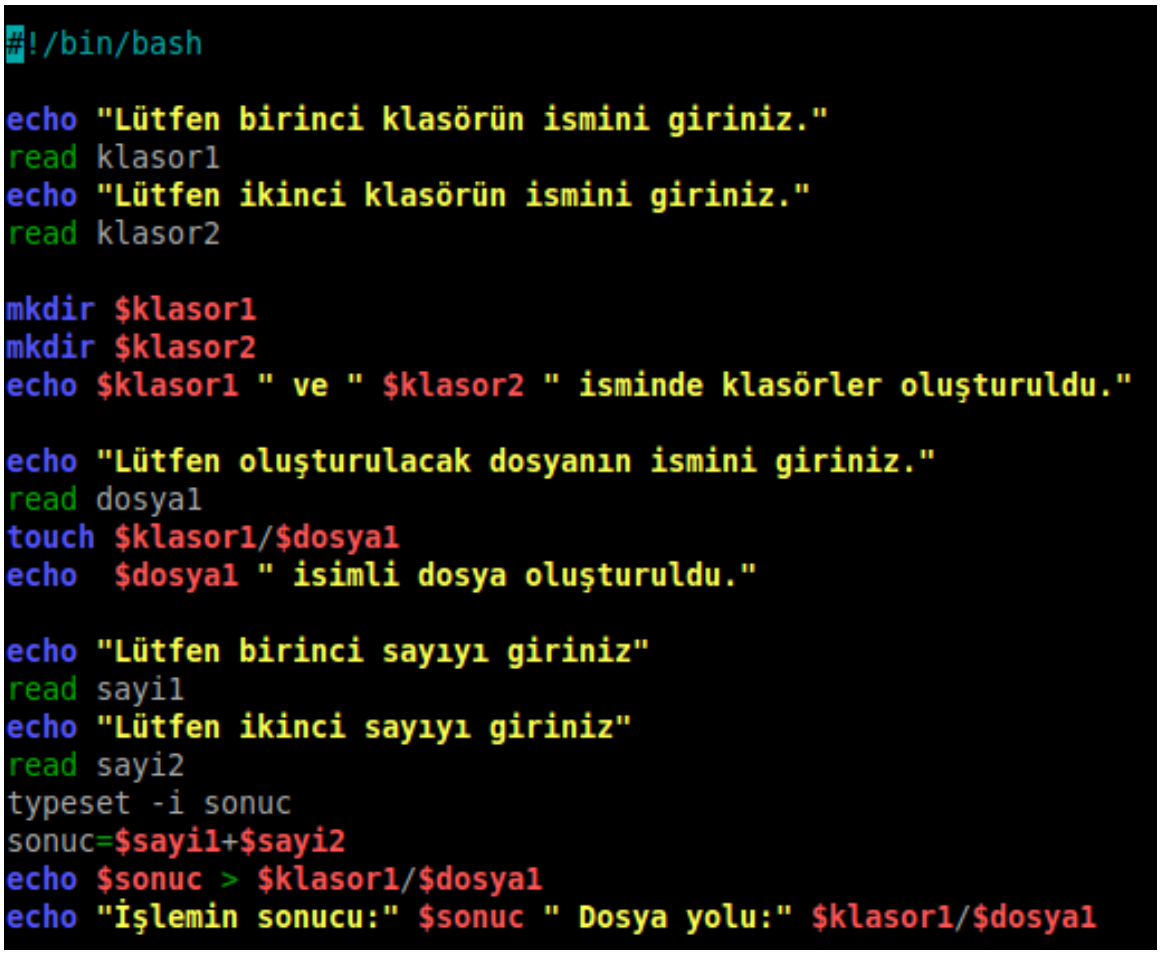
Son echo satırımız ile de sonucu ve sonucun yazdığı dosyanın yolunu ekrana bastırıyoruz.

İşte şimdiye dek öğrendiklerimizi kullanarak bir script dosyası yazmış olduk. Bunu yapabilmiş olmamız önceki konularla ilgili bir problemimizin olmadığını göstermekte.

Peki, uygulama bu hâliyle çalışmakta yalnız uygulamada birtakım hatalar meydana getirip uygulamanın çalışmasının yarıda kesilmesi şu anki hâliyle sizce mümkün müdür? Evet mümkündür. Mesela sayıları kesirli girmemiz veya sayı yerine karakter girmemiz uygulamaya hata verdirir ve çalışmaz. İsterseniz deneyebilirsiniz.

İşte geldiğimiz bu noktada kontrol deyimlerine ihtiyaç ortaya çıkıyor. Kapasitesi ne olursa olsun programlamada kontrol deyimleri hayati önem taşımaktadır.
Kontrol deyimi olarak if-then-else komutlarından bahsetmekteyiz. Şimdi son örneğimizi if-then-else ile kontrole tabi tutup hata payını sıfırlamaya çalışalım. Böylece uygulama hem daha kararlı hâle gelecek hem de biz yeni ve önemli bir konuya başlayacağız.

**Şimdi örneğimizi şu şekilde değiştirelim:**
Şimdilik kontrollere başlangıç yapmamız açısından kolay bir kontrol kurgusu kuralım. Örneğin kullanıcının gireceği ilk gireceği sayı ikinci gireceği sayıdan büyük olsun. İşlem olarak da çıkartma işlemi yapalım. İlk sayı büyük olacağı



```
#!/bin/bash

echo "Lütfen birinci klasörün ismini giriniz."
read klasor1
echo "Lütfen ikinci klasörün ismini giriniz."
read klasor2

mkdir $klasor1
mkdir $klasor2
echo $klasor1 " ve " $klasor2 " isminde klasörler oluşturuldu."

echo "Lütfen oluşturulacak dosyanın ismini giriniz."
read dosya1
touch $klasor1/$dosya1
echo  $dosya1 " isimli dosya oluşturuldu."

echo "Lütfen birinci sayıyı giriniz"
read sayi1
echo "Lütfen ikinci sayıyı giriniz"
read sayi2
typeset -i sonuc
sonuc=$sayi1+$sayi2
echo $sonuc > $klasor1/$dosya1
echo "İşlemin sonucu:" $sonuc " Dosya yolu:" $klasor1/$dosya1
```

Şekil 1:

için - sonuç çıkmayacak. Böylece uygulama hep doğru-mantıklı çalışacak yani anlamsız hatalara düşmeyecek. Bunu sağlamamız durumunda en basit ifadeyle kararlılığı sağlamış oluyoruz diyebiliriz.

Şimdi ilk önce matematik işlemimizde eskiden toplam alan ifadeyi çıkartma yapalım.

```
sonuc=$sayi1-$sayi2
```

Şekil 2:

Şimdi de 1.sayıyı 2.sayıdan büyük girmesine zorlayan kontrolümüzü koyacağız.

Kontrolü koda eklemeden evvel kontrol terimlerine değinelim. Kullanacağımız kontrol programlama dillerinde sık kullanılan ve önemli işlevler üstlenen if-then-else deyimi.

Biz sayı1 ve sayı2 değerlerini kullanıcıdan alıp hemen sonrasında if'i kullanarak sayıları karşılaştıracağız. Yaptığımız karşılaştırmanın sonucuna göre de eğer sorduğumuz soru olumlu dönerse yani true olursa işlemi yapacağız. Burası kontrol deyiminin then tarafını oluşturuyor. Şayet sorduğumuz soru olumsuz dönerse yani false olursa kullanıcıyı tekrar sayı girmeye zorlayacağız. Burası da kontrolün else tarafını oluşturmakta.

İf ile eğer sayı1, sayı2'den küçükse diye soruyoruz.
Sayı1, sayı2'den küçükse then (ise) kısmıyla normal işlemimizi gerçekleştiriyoruz. Eğer küçük değilse else (değilse) kısmıyla sayıları tekrar alıyoruz.

İşte bunu yaparak kullanıcıyı belli bir çerçevede sayı girmeye zorluyor ve yapacağımız matematiksel işleme uygun sonuç çıkaracak değerleri alıyoruz. Uygun değerleri aldığımız, uygun olmayan değerleri reddettiğimiz için uygulamamız yaptığı matematik işleminde anormal bir sonuç çıkartmıyor. İşte bu kontrol sayesinde uygulamamız kararlı (stabil) çalışıyor.

> Yaptığımız işlem yazdığımız koda kontrol koymaktı. Biz kontrolün içinde soruyu tekrar soran zorlayıcı bir ifadeye de yer verdik. Bu ifade için de kodun başına bir fonksiyon ekledik. Buna bu yazımızda değinmeyeceğiz. Önümüzdeki yazımızda kontrol deyimleri kullanımına daha derinlemesine bakarken kontrolle beraber kullandığımız zorlayıcı ifadeyi de açıklamaya çalışacağız. Daha detaylı inceleme yapacağımız için konu daha da pekişecek.

Şimdi yukarıda açıkladığımız kontrolümüzü kodumuza ekleyelim.

```
if [ $sayi1 -gt $sayi2 ]
   then
        typeset -i sonuc
        sonuc=$sayi1-$sayi2
        echo $sonuc > $klasor1/$dosya1
        echo "İşlemin sonucu:" $sonuc " Dosya yolu:" $klasor1/$dosya1
   else
        echo "1. sayı 2. sayıdan küçük. Lütfen tekrar giriniz."
jumpto islembasi
fi
```

Şekil 3:

İşte if ile başlayıp fi ile biten ve içinde işlemimizi barındıran kontrol deyimimiz. Kontrol deyimimizdeki detaylara önümüzdeki sayıda değineceğiz.

**Uygulamamızın son değişikliklerden sonraki hâli şöyledir:**

Kodun tümünde göreceğiniz gibi bu yazımızda değinmediğimiz fonsiyon tanımı ve if-then-else deyimini görmektesiniz.

```
#!/bin/bash
function jumpto
{
    label=$1
    cmd=$(sed -n "/$label:/{:a;n;p;ba};" $0 | grep -v ':$')
    eval "$cmd"
    exit
}
echo "Lütfen birinci klasörün ismini giriniz."
read klasor1
echo "Lütfen ikinci klasörün ismini giriniz."
read klasor2

mkdir $klasor1
mkdir $klasor2
echo $klasor1 " ve " $klasor2 " isminde klasörler oluşturuldu."

echo "Lütfen oluşturulacak dosyanın ismini giriniz."
read dosya1
touch $klasor1/$dosya1
echo  $dosya1 " isimli dosya oluşturuldu."

islembasi:
echo "Lütfen birinci sayıyı giriniz"
read sayi1
echo "Lütfen ikinci sayıyı giriniz"
read sayi2
if [ $sayi1 -gt $sayi2 ]
   then
        typeset -i sonuc
        sonuc=$sayi1-$sayi2
        echo $sonuc > $klasor1/$dosya1
        echo "İşlemin sonucu:" $sonuc " Dosya yolu:" $klasor1/$dosya1
   else
        echo "1. sayı 2. sayıdan küçük. Lütfen tekrar giriniz."
jumpto islembasi
fi
```

Şekil 4:

Sizler de gelecek sayıda değineceğimiz bu kodlara çalışabilir ve bize denemelerinizi gönderebilirsiniz. Burada yayınlamaktan ve beraber geliştirmekten zevk duyarız. Gelecek sayıda yine bu yazımızı toparlama mahiyetinde özetleyeceğiz ve kontrol deyimlerine, fonksiyonlara bakacağız.