MİMAR SİNAN FINE ARTS UNIVERSITY

BİLGİ İŞLEM

MSGSU FİZİK BÖLÜMÜ

DERS 5

Yrd Doç Dr. Ferhat ÖZOK
MSGSU FİZİK BÖLÜMÜ

MİMAR SİNAN FINE ARTS UNIVERSITY

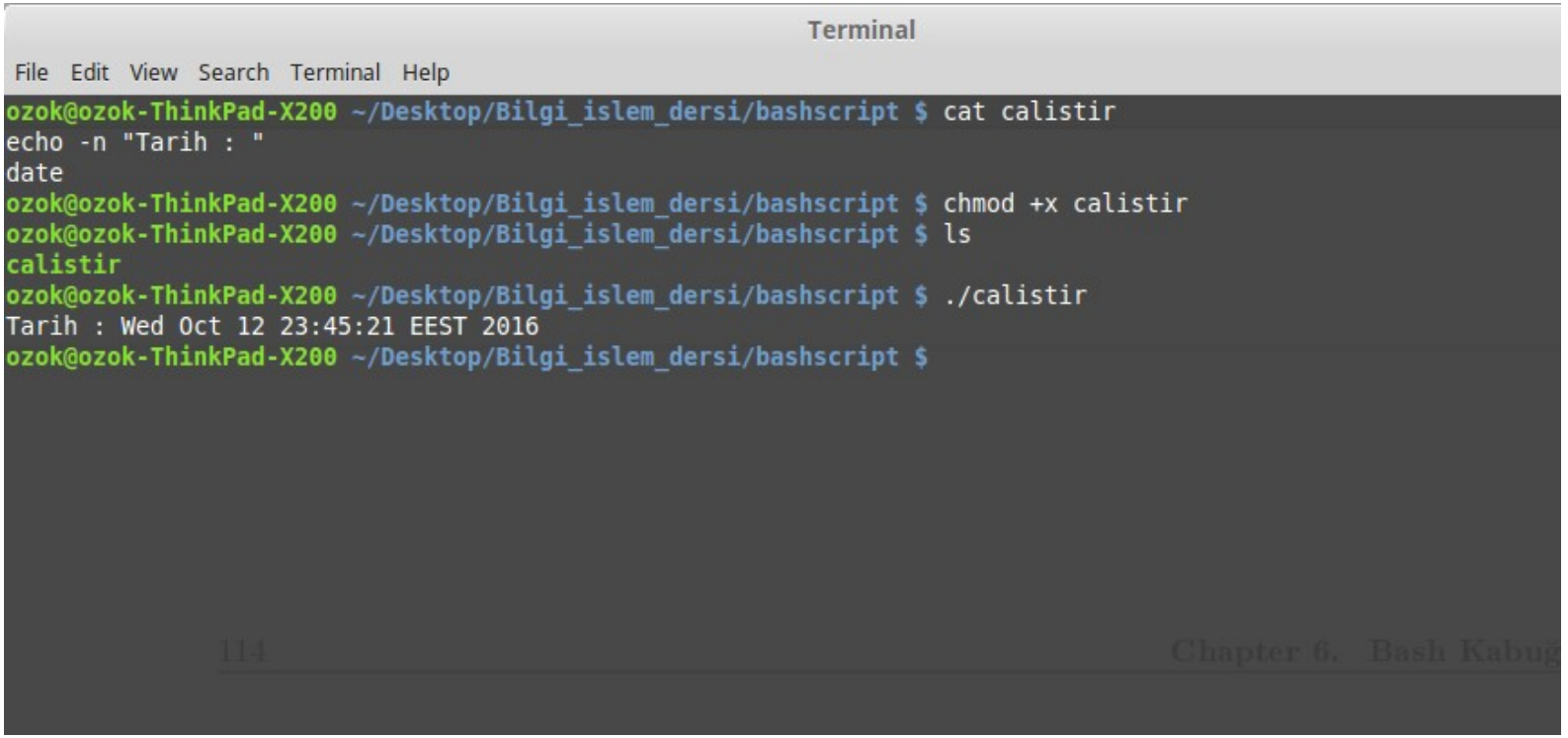
Kabuk Programlamaya Giris

Her kabugun kendine özgu programlama dili yapısı vardır. Bash kabuğu güçlü programlama özellikleriyle karmaşık programların rahatca yazılmasına izin verir. Mantıksal operatörler, döngüler, değişkenler ve modern programlama dillerinde bulunan pek çok özellik bash kabuğunda da vardır ve işleyiş tarzlarıda hemen hemen aynıdır. Genellikle, bir program olusturacak olan komutlar bir dosyaya yazılırlar ve ardından bu dosya calıştırılır. Herhangi bir editor yardımıyla yazılan program, daha sonra kabuk altında calıstırılır.
Bir kabuk programı aşka kabuk programlarınıda çalıstırabilir. Bundan sonraki derslerimizde kabuk programlama konusunda duracağız.

MİMAR SİNAN
FINE ARTS
UNIVERSITY

Kabuk Programları , bir veya birden fazla linux komutunu tutan dosyalardır. Bu dosya Hazırlandıktan sonra calıstırılabilir. Bir kabuk programı, istenen kullanıcılar için calıştırılabilir hale getirilebilrir
$ chmod +x komut-ismi

Kabuk programlar yazarken dosyan n islevini ve her sat rdaki komutun veya komut kumesi ac klamak için açıklama satırları kullanmak işe yarar. Bir açıklama eklemek için
satır başına (veya bos satıra) # isareti eklenir ve ardından istenilen cumle girilir. # isaretinden sonraki tum satır kabuk taraf ndan gozardı edilir.



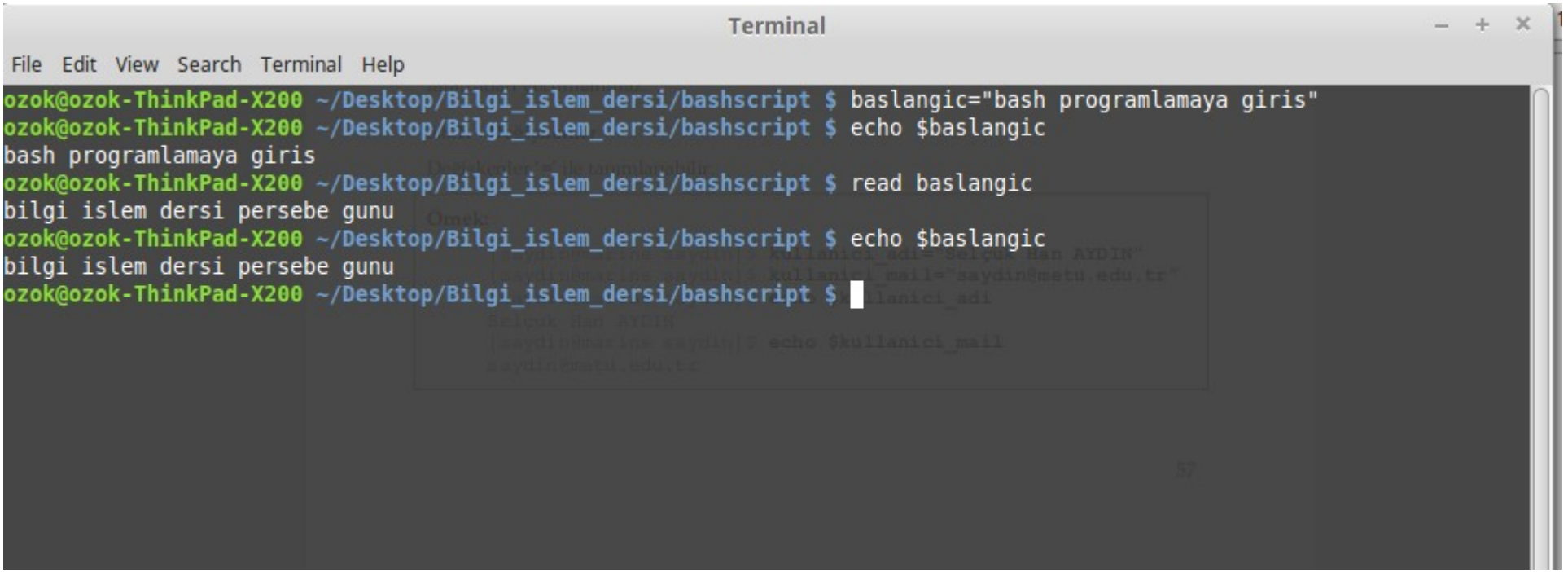
MİMAR SİNAN FINE ARTS UNIVERSITY

Değişkenlerin Kullanımı

Bir değişkene değer atandığı anda sistem tarafından tanınır. Degiskenler alfabetik veya nümerik karakterlerden olusabilirler fakat bir degisken say sal bir deger ile baslayamaz. Bunların dışında degişken isminin içinde " " karakteri de bulunabilir. Bir değiskene değer ataması "=" isareti yardımıyla yapılır.

Değer Okuma

Değişkenlere read komutu ile çalışma sırasında dışarıdan değer atanabilir



MİMAR SİNAN FINE ARTS UNIVERSITY

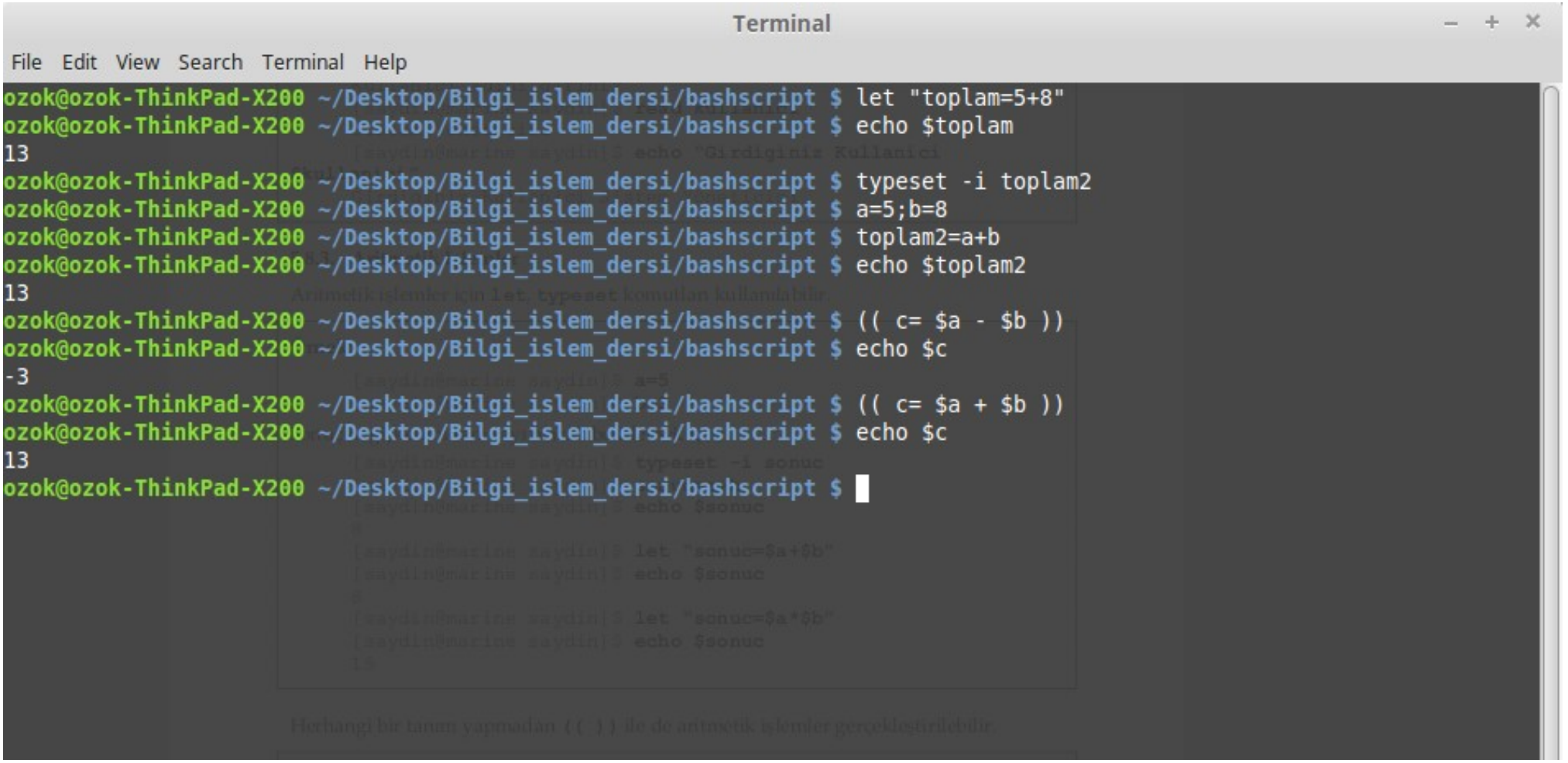
Aritmetik İşlemler

bash kabugunda matematiksel islemlere buyuk sınırlamalar getirilmiştir. Tamsayı değiskenler dışında matematiksel değişken kullanmak icin bu işlemler için geliştirilmiş ve kolaylıklar sağlayan awk veya bc gibi komutlar kullanılabilir. Aritmetik islemler icin eval komutunu veya bash kabugu altında yerlesik (builtin) komut olan let komutunu kullanabilirsiniz.

let "degisken=aritmetik islem"

Aritmetik değişken tanımlamanın diğer bir yolu da typeset komutudur. Herhangi bir tanım yapmadan (( )) ile de aritmetik işlemler gerçekleştirilebilir.

MİMAR SİNAN
FINE ARTS
UNIVERSITY



MİMAR SİNAN FINE ARTS UNIVERSITY

If-Else Kalıbı ve Kontrol İşlemleri

Bash kabuk programında $? ile en son çalışan komutun sonucu öğrenilebilir. Bu özellikten yararlanarak komutların sonuçları ile ilgili bilgi edinilebilir. Eğer komutlar başarılı bir şekilde sonuçlanmışsa 0 sonucunu, diğer durumlarda 0'dan farklı bir değer üretirler.

```
ozok@ozok-ThinkPad-X200 ~ $ test 5 -gt 3
ozok@ozok-ThinkPad-X200 ~ $ echo $?
0
ozok@ozok-ThinkPad-X200 ~ $ test 3 -gt 3
ozok@ozok-ThinkPad-X200 ~ $ echo $?
1
ozok@ozok-ThinkPad-X200 ~ $
```

If-Else Kalıbı ve Kontrol İşlemleri

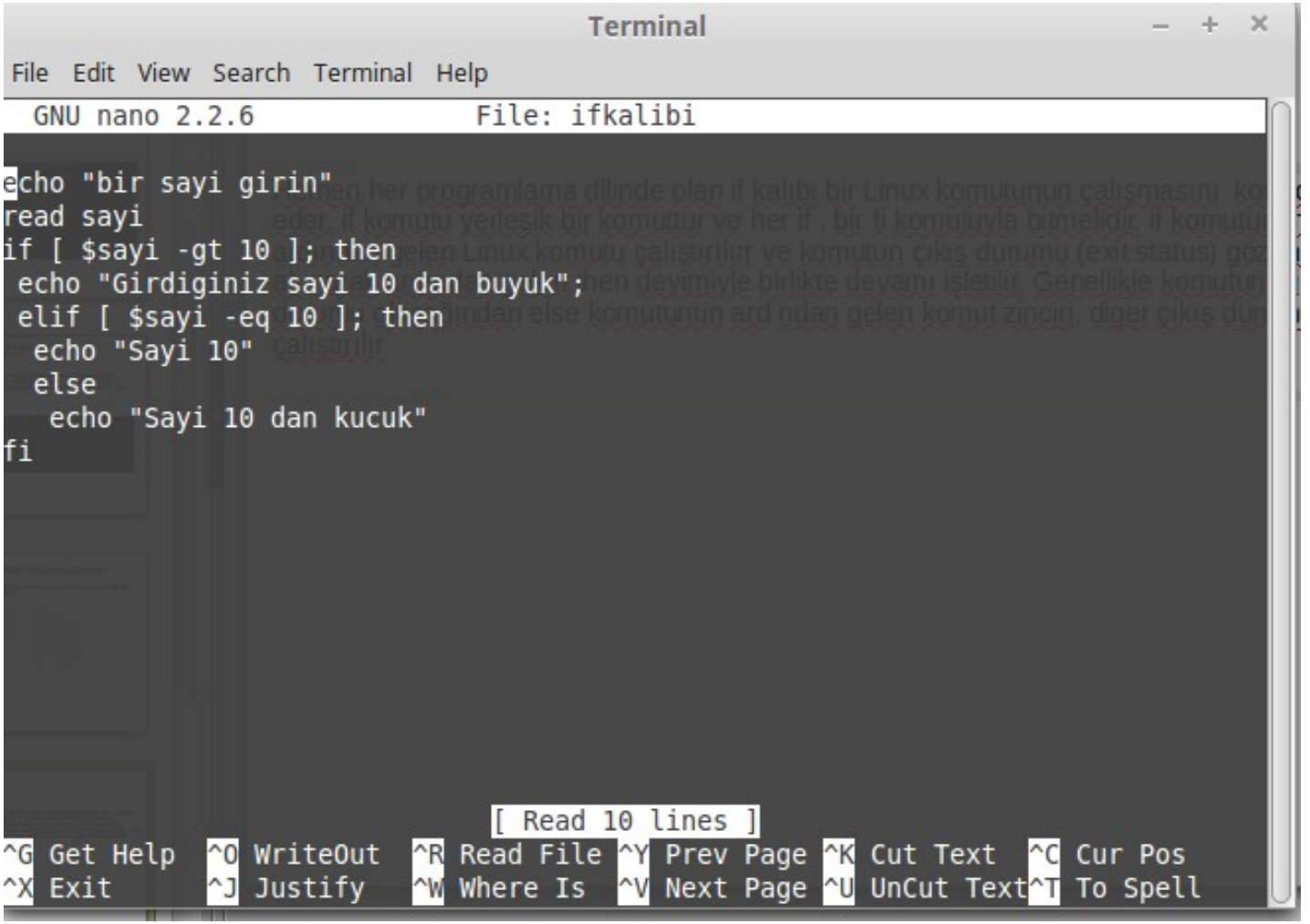
Aritmetik karşılaştırmalarda, karakter dizileri (string) ve dosya karşılaştırmalarında aşağıdaki seçenekler kullanılabilir:

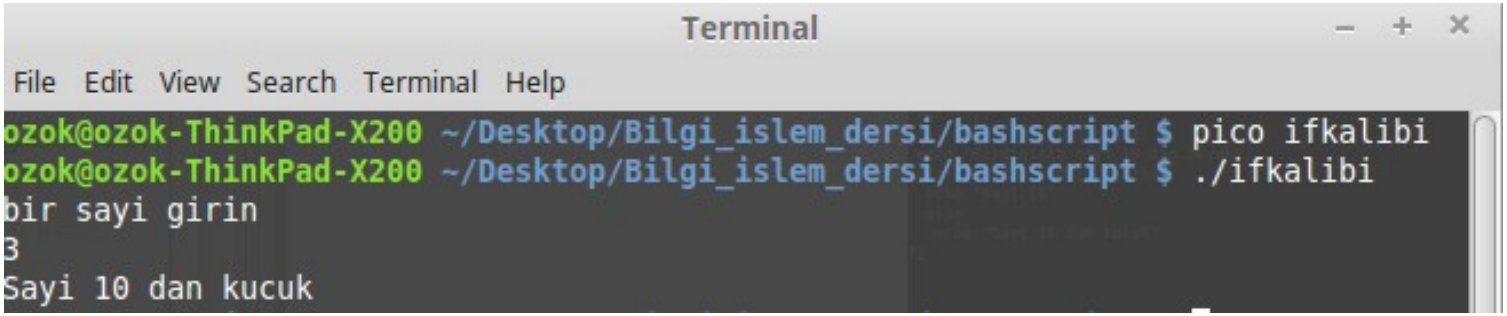
Aritmetik		Karakter dizileri		Dosya	
`eq`	eşit	`-z`	Boş	`-f`	Dosya var
`gt`	büyük	`-n`	Tanımlı	`-s`	Dosya boş değil
`lt`	küçük	`=`	Eşit	`-r`	Dosya okunabilir
`ge`	büyük eşit	`!=`	Farklı	`-w`	Dosya yazılabilir
`le`	küçük eşit			`-x`	Dosya çalıştırılabilir
				`-h`	Sembolik bağlantı
				`-c`	Karakter dosyası
				`-b`	Blok dosyası

Hemen her programlama dilinde olan if kalıbı bir Linux komutunun çalışmasını kontrol (test) eder. if komutu yerlesik bir komuttur ve her if , bir fi komutuyla bitmelidir. if komutunun ardından gelen Linux komutu çalıştırılırr ve komutun çıkış durumu (exit status) gözönüne alı narak ardından gelen then deyimiyle birlikte devamı işletilir. Genellikle komutun iki türlü çıkış durumu olacağından else komutunun ardından gelen komut zinciri, diger çıkış durumunda çalıştırılır
İki mantıksal sonuç && (ve), || (ya da) ile karşılaştırılabilir

```
if linux komutu
then
komut1
komut2
...
else
komut1
komut2
...
fi
```

MİMAR SİNAN
FINE ARTS
UNIVERSITY





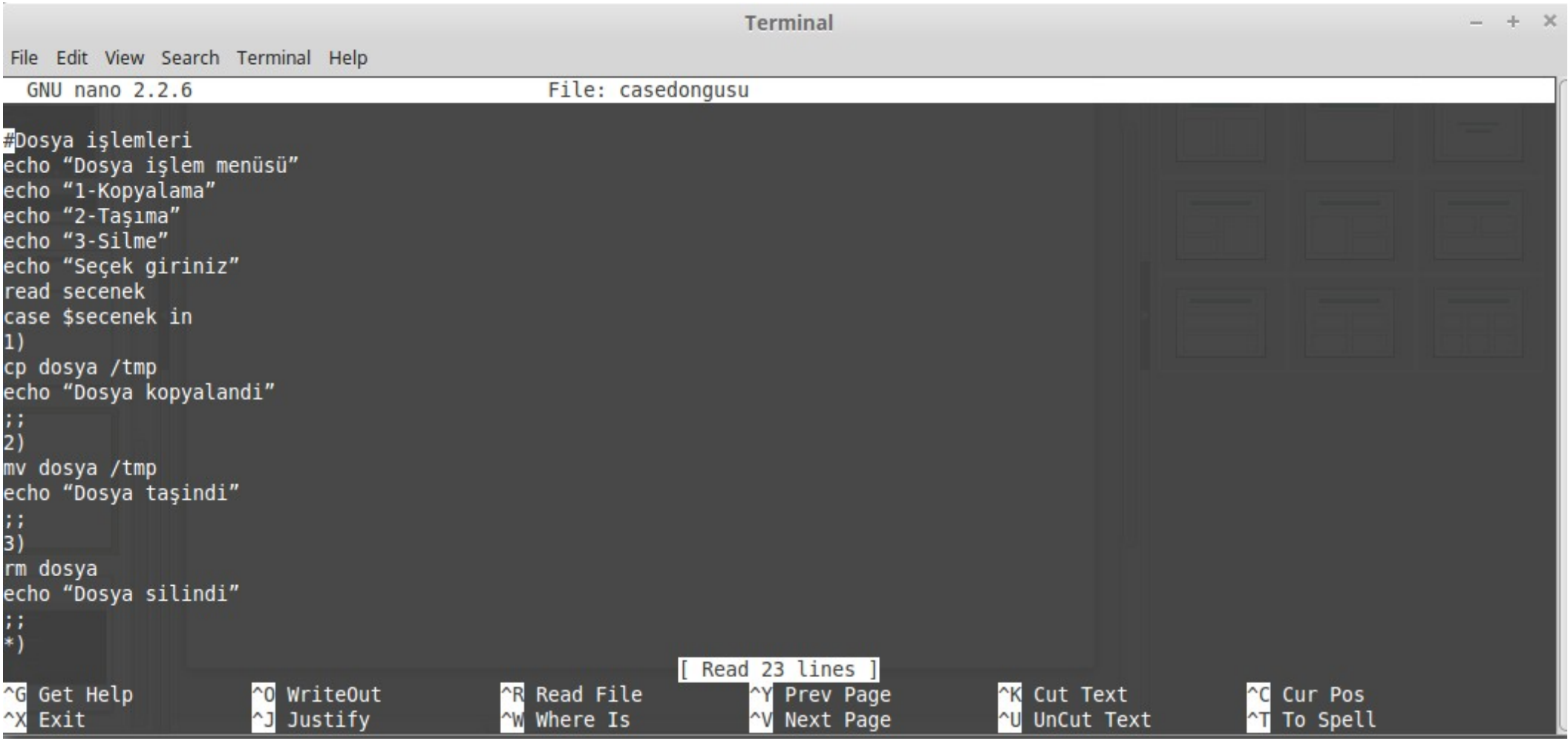
MİMAR SİNAN FINE ARTS UNIVERSITY

Case Döngüsü

Birkaç alternatif arasında seçim yapılması gerekiyorsa case kalıbı kullanılır. Komutun genel kullanımı şu biçimdedir:

```
case aranacak_kelime in
seçenek1)
komutlar
;;
seçenek2)
komutlar
;;
...
*)
komutlar
;;
esac
```

MİMAR SİNAN
FINE ARTS
UNIVERSITY



MİMAR SİNAN
FINE ARTS
UNIVERSITY

Terminal

File Edit View Search Terminal Help

```
ozok@ozok-ThinkPad-X200 ~/Desktop/Bilgi_islem_dersi/bashscript $ ls
calistir  casedongusu  dosya  ifkalibi
ozok@ozok-ThinkPad-X200 ~/Desktop/Bilgi_islem_dersi/bashscript $ ./casedongusu
"Dosya işlem menüsü"
"1-Kopyalama"
"2-Taşıma"
"3-Silme"
"Seçek giriniz"
5
"Yanliş bilgi girişi"
ozok@ozok-ThinkPad-X200 ~/Desktop/Bilgi_islem_dersi/bashscript $ ./casedongusu
"Dosya işlem menüsü"
"1-Kopyalama"
"2-Taşıma"
"3-Silme"
"Seçek giriniz"
1
"Dosya kopyalandi"
ozok@ozok-ThinkPad-X200 ~/Desktop/Bilgi_islem_dersi/bashscript $ ls /tmp/
dosya                       icedteaplugin-root-VJ52vi  MathLink
hsperfdata_mdm              kde-ozok                   mintUpdate
hsperfdata_ozok             ksocket-ozok               OSL_PIPE_1000_SingleOfficeIPC_8dc3b8df23cb7a490fd786e34aba479
hsperfdata_root             lu212847c8ji4.tmp          pulse-PKdhtXMmr18n
icedteaplugin-mdm-fpBgvf    m000002191301              ssh-IslOOuHqxlD7
icedteaplugin-root-NrVdA9   m00000285851
ozok@ozok-ThinkPad-X200 ~/Desktop/Bilgi_islem_dersi/bashscript $
```

MİMAR SİNAN FINE ARTS UNIVERSITY

Döngüler

Bash kabuk programında aşağıdaki döngü türleri kullanılabilir:

while kontrol ifadesi
do
komutlar
Done
ya da;

for değişken in değerler
do
komutlar
done

MİMAR SİNAN
FINE ARTS
UNIVERSITY

Terminal

File Edit View Search Terminal Help

GNU nano 2.2.6 File: dowhile

```
toplam=0
i=1
while [ $i -le 10 ]
do
 (( toplam = $toplam +$i))
 (( i = $i + 1 ))
  done

echo $toplam
```

[Read 9 lines]

^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell

```
ozok@ozok-ThinkPad-X200 ~/Desktop/Bilgi_islem_dersi/bashscript $ ./dowhile
55
ozok@ozok-ThinkPad-X200 ~/Desktop/Bilgi_islem_dersi/bashscript $
```

MİMAR SİNAN
FINE ARTS
UNIVERSITY

Terminal

File Edit View Search Terminal Help

GNU nano 2.2.6 File: fortest

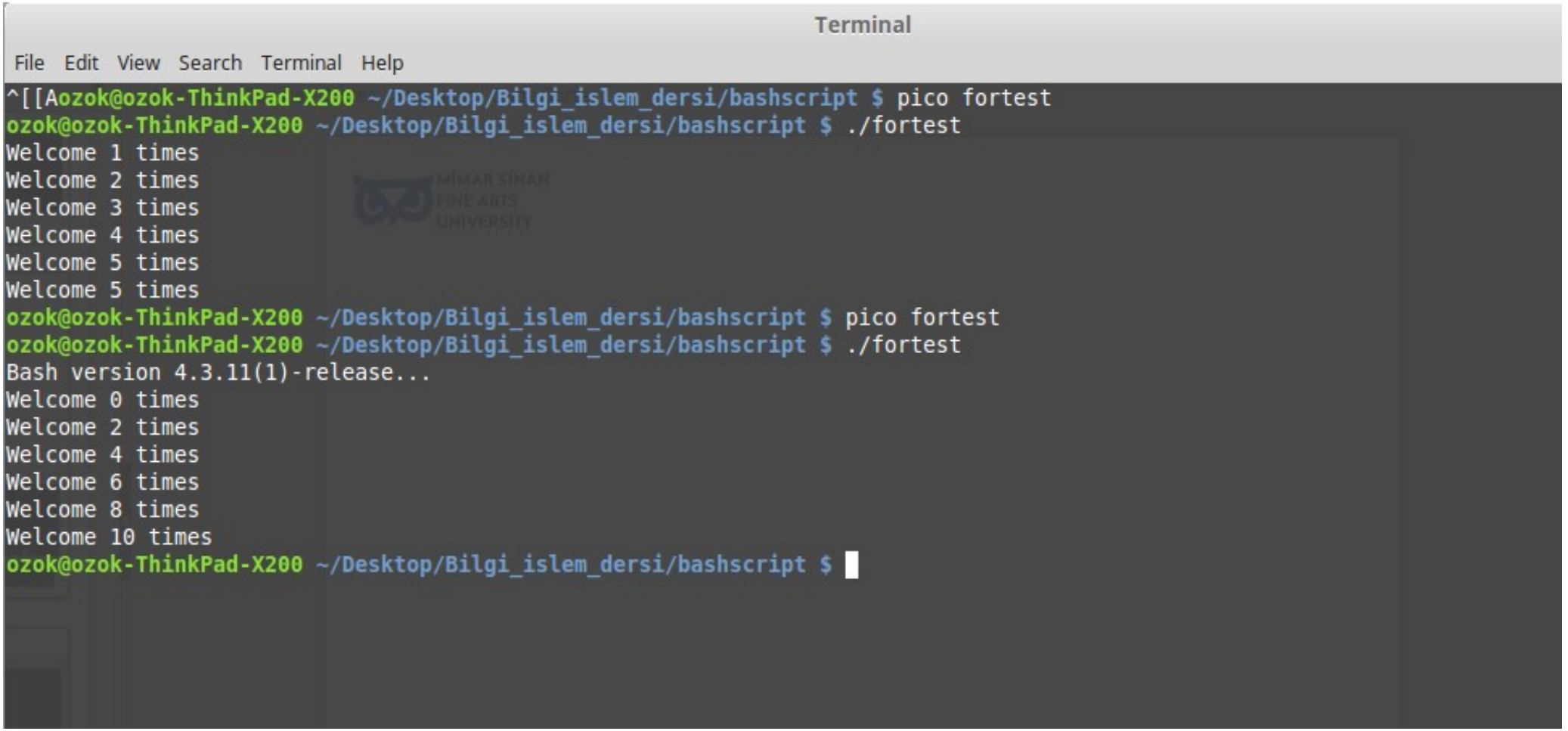
```
#!/bin/bash
for i in {1..5}
do
   echo "Welcome $i times"
done


#echo "Bash version ${BASH_VERSION}..."
#for i in {0..10..2}
#  do
     echo "Welcome $i times"
# done
#for i in 1 2 3 4 5
#do
#   echo "Welcome $i times"
#done
```

[Read 17 lines]

^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell

MİMAR SİNAN
FINE ARTS
UNIVERSITY



MİMAR SİNAN FINE ARTS UNIVERSITY

Kabuk programlarında kullanıcı tarafından tanımlanmış fonksiyonlar kullanılabilir.

Terminal

File Edit View Search Terminal Help

GNU nano 2.2.6 File: islemornek

```
#!/bin/bash
# Fonksiyon kullanim örnek programi
islem(){
sayi1=$1
sayi2=$2
operator=$3
case $operator in
+)
echo "Toplama Islemi"
(( sonuc= $sayi1 + $sayi2 ))
;;
-)
echo "Cikarma Islemi"
(( sonuc= $sayi1 - $sayi2 ))
;;
x)
echo "Carpma Islemi"
(( sonuc= $sayi1 * $sayi2 ))
;;
/)
echo "Bolme Islemi"
(( sonuc= $sayi1 / $sayi2 ))
;;
*)
sonuc="Hatali Islem"
;;
esac
echo $sonuc
}
echo -n "Degerleri giriniz (sayi1 sayi2 operator) : "
read s1 s2 op
echo "Girilen Degerler"
echo "1. sayi : $s1"
echo "2. sayi : $s2"
echo "Operator : $op"
islem $s1 $s2 $op
echo "Islem Tamam"
```

^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell

MİMAR SİNAN FINE ARTS UNIVERSITY

Kabuk programlarında kullanıcı tarafından tanımlanmış fonksiyonlar kullanılabilir.

Terminal

File Edit View Search Terminal Help

```
ozok@ozok-ThinkPad-X200 ~/Desktop/Bilgi_islem_dersi/bashscript $ ./islemornek
Degerleri giriniz (sayi1 sayi2 operator) : 9 3 /
Girilen Degerler
1. sayi : 9
2. sayi : 3
Operator : /
Bolme Islemi
3
Islem Tamam
ozok@ozok-ThinkPad-X200 ~/Desktop/Bilgi_islem_dersi/bashscript $ ./islemornek
Degerleri giriniz (sayi1 sayi2 operator) : 6 4 -
Girilen Degerler
1. sayi : 6
2. sayi : 4
Operator : -
Cikarma Islemi
2
Islem Tamam
ozok@ozok-ThinkPad-X200 ~/Desktop/Bilgi_islem_dersi/bashscript $ ./islemornek
Degerleri giriniz (sayi1 sayi2 operator) : 8 4 x
Girilen Degerler
1. sayi : 8
2. sayi : 4
Operator : x
Carpma Islemi
32
Islem Tamam
ozok@ozok-ThinkPad-X200 ~/Desktop/Bilgi_islem_dersi/bashscript $ ./islemornek
Degerleri giriniz (sayi1 sayi2 operator) : 4 7 :
Girilen Degerler
1. sayi : 4
2. sayi : 7
Operator : :
Hatali Islem
Islem Tamam
ozok@ozok-ThinkPad-X200 ~/Desktop/Bilgi_islem_dersi/bashscript $
```